

**UNIVERSITAS DIPONEGORO**

**PEMBUATAN APLIKASI ANDROID “DRIVER CONTROL” SEBAGAI SARANA MEMONITOR ANAK BERKENDARA SECARA WAKTU-NYATA**

**TUGAS AKHIR**

**LIA LIDYA ROZA**

**21120112120011**

**FAKULTAS TEKNIK**

**DEPARTEMEN SISTEM KOMPUTER**

**SEMARANG**

**DESEMBER 2016**



**UNIVERSITAS DIPONEGORO**

**PEMBUATAN APLIKASI ANDROID “DRIVER CONTROL” SEBAGAI SARANA MEMONITOR ANAK BERKENDARA SECARA WAKTU-NYATA**

**TUGAS AKHIR**

**Diajukan sebagai salah satu syarat untuk memperoleh gelar Sarjana Teknik**

**Lia Lidya Roza**

**21120112120011**

**FAKULTAS TEKNIK**

**DEPARTEMEN SISTEM KOMPUTER**

**SEMARANG**

**DESEMBER 2016**

# HALAMAN PENGESAHAN

Tugas Akhir ini diajukan oleh :

Nama : Lia Lidya Roza

NIM : 21120112120011

Departemen : Sistem Komputer

Judul Tugas Akhir : Pembuatan Aplikasi Android “Driver Control” Sebagai Sarana Memonitor Anak Berkendara Secara Waktu-Nyata

Telah berhasil dipertahankan di hadapan Tim Penguji dan diterima sebagai bagian persyaratan yang diperlukan untuk memperoleh gelar Sarjana Teknik pada Departemen Sistem Komputer, Fakultas Teknik, Universitas Diponegoro.

**TIM PENGUJI**

Pembimbing I : Dr. R. Rizal Isnanto, S.T., M.M., M.T. ( )

Pembimbing II : Eko Didik Widianto, S.T., M.T. ( )

Ketua Penguji : Rinta Kridalukmana, S.Kom, M.T. ( )

Sekretaris Penguji : Ike Pertiwi Windasari, S.T., M.T. ( )

Anggota Penguji : Dania Eridani, S.T., M.Eng. ( )

Semarang, 14 Desember 2016

Ketua Departemen Sistem Komputer

Dr. R. Rizal Isnanto, S.T., M.M., M.T.

NIP. 197007272000121001

# HALAMAN PERNYATAAN ORISINALITAS

**Tugas Akhir ini adalah hasil karya saya sendiri,**

**dan semua sumber baik yang dikutip maupun yang dirujuk telah saya nyatakan dengan benar.**

NAMA : LIA LIDYA ROZA

NIM : 21120112120011

Tanda Tangan :

Tanggal : Semarang, 1 November 2016

# HALAMAN PERNYATAAN PERSETUJUAN PUBLIKASI TUGAS AKHIR UNTUK KEPENTINGAN AKADEMIS

Sebagai sivitas akademika Universitas Diponegoro, saya yang bertanda tangan di bawah ini :

Nama : LIA LIDYA ROZA

NIM : 21120112120011

Departemen : SISTEM KOMPUTER

Fakultas : TEKNIK

Jenis Karya : TUGAS AKHIR

demi pengembangan ilmu pengetahuan, menyetujui untuk memberikan kepada Universitas Diponegoro **Hak Bebas Royalti Noneksklusif** (*Non-exclusive Royalty Free Right*) atas karya ilmiah saya yang berjudul :

Pembuatan Aplikasi Android “Driver Control” Sebagai Sarana Memonitor Anak Berkendara Secara Waktu-Nyata.

beserta perangkat yang ada (jika diperlukan). Dengan Hak Bebas Royalti/Noneksklusif ini Universitas Diponegoro berhak menyimpan, mengalihmedia/formatkan, mengelola dalam bentuk pangkalan data (basisdata), merawat dan memublikasikan tugas akhir saya selama tetap mencantumkan nama saya sebagai penulis/pencipta dan sebagai pemilik Hak Cipta.

Demikian pernyataan ini saya buat dengan sebenarnya.

Dibuat di : Semarang

Pada Tanggal : 1 November 2016

Yang menyatakan

(Lia Lidya Roza)

# ABSTRAK

*Zaman sekarang motor merupakan salah satu kebutuhan primer di Indonesia, hampir setiap keluarga setidaknya memiliki minimal 1 kendaraan bermotor, tak terkecuali anak-anak yang juga sudah diizinkan orang tua untuk mengendarai sepeda motor agar lebih mandiri. Namun tanpa disadari bahaya kecelakaan dan kejahatan seperti pencurian sepeda motor terus mengintai sang anak. Untuk itu diperlukan sebuah aplikasi yang dapat memonitor anak berkendara secara waktu-nyata sebagai upaya mengurangi angka kecelakaan lalu lintas dan pencurian sepeda motor.*

*Metodologi penelitian pada aplikasi ini dibuat menggunakan model waterfall. Pembuatan aplikasi ini menggunakan perangkat lunak Android Studio dengan bahasa pemrograman Java, basisdata MySQL, dan berbasis Web Service.*

*Hasil penelitian ini adalah aplikasi Android Driver Control yang mampu memberikan informasi lokasi kendaraan, kecepatan, notifikasi motor jatuh, dan rekaman data pengemudi. Pengguna dapat mematikan atau menghidupkan mesin motor secara otomatis melalui aplikasi ini. Aplikasi ini juga memiliki fasilitas login untuk admin yang digunakan untuk mengaktifkan atau menonaktifkan alat Driver Control. Aplikasi ini dapat digunakan lebih dari satu pengguna untuk memantau satu sepeda motor yang sudah terpasang alat Driver Control.*

***Kata Kunci :*** *Motor, Driver Control, Android Studio*

# ABSTRACT

*Nowadays, motorcycle is one of the primary needs in Indonesia. Almost every family has at least one motorcycle, includes the children who is allowed by their parents to ride a motorcycle independently. However, without any realization, accidents and crimes such as motorcycle robbery are lurking children continuously. Because of that, a real time application is needed to monitor the children while riding motorcycle. This application is made as an effort to reduce the number of traffic accidents and motorcycle robbery.*

*Waterfall model is used as research's method on developing the application. The application is developed on Android Studio, using Java programming language, database MySQL, and a web-based service.*

*This research's results is an Android Driver Control Application which can provide motorcycle location information, speed, fallen motorcycle notifications, and driver record data. Users can turn off or turn on the motorcycle's engine automatically through this application. It also has login facility for admin to enable or disable the Driver Control tool and can be used by one or more user to monitor their motorcycle with Driver Control tool installed on it.*

***Keywords :*** *motorcycle, Driver Control, Android Studio*

# KATA PENGANTAR

Puji syukur kehadirat Tuhan Yang Maha Esa yang telah melimpahkan rahmat dan anugerah-Nya sehingga penulis dapat menyelesaikan laporan Tugas Akhir yang berjudul “PEMBUATAN APLIKASI ANDROID *DRIVER CONTROL* SEBAGAI SARANA MEMONITOR ANAK BERKENDARA SECARA WAKTU-NYATA”.

Dalam penyusunan laporan Tugas Akhir ini penulis banyak mendapatkan bimbingan, dorongan dan bantuan dari berbagai pihak. Oleh karena itu, melalui kesempatan ini penulis menyampaikan ucapan terima kasih kepada :

1. Bapak Dr. R. Rizal Isnanto, S.T., M.M., M.T. selaku Ketua Departemen Sistem Komputer Fakultas Teknik Universitas Diponegoro.
2. Bapak Dr. R. Rizal Isnanto, S.T., M.M., M.T. selaku dosen wali, dan dosen pembimbing I, yang telah memberikan petunjuk serta bimbingan dalam pembuatan Tugas Akhir.
3. Bapak Eko Didik Widianto, S.T., M.T. selaku dosen pembimbing II yang telah memberikan petunjuk serta bimbingan dalam pembuatan Tugas Akhir.
4. Ibu Ike Pertiwi Windasari, S.T., M.T. selaku Koordinator Tugas Akhir, yang telah membantu kelancaran Tugas Akhir ini.
5. Bapak dan Ibu dosen Jurusan Teknik Sistem Komputer Universitas Diponegoro.
6. Mas Andi, Mbak Okta, Mbak Melati, Mas Sidik, Mas Erwan atas bantuannya dalam kelengkapan administrasi.
7. Orang tua dan keluarga yang sangat penulis sayangi, yang telah memberikan doa restu, semangat, dorongan dan bantuan kepada penulis selama kuliah, dan mengerjakan Tugas Akhir ini.
8. Khoirunissa Waskitaningrum, dan Mas Budi Cahyo S.P. yang sudah sangat membantu dalam pembuatan alat *Driver Control*.
9. Mas Eko, Mas Adik, Dhewa, Febri, Alif, Syafira, Dhea, Bokir, dan Mba Linda yang sudah membantu dan memberi semangat dalam pembuatan Tugas Akhir ini.
10. Teman-teman Program Studi Sistem Komputer, khususnya angkatan 2012 yang sudah membantu saya dalam kegiatan akademis maupun organisasi selama 4 tahun di Undip.
11. Ibu Istijabah, Pak Tohir, Mas Marsono, dan teman-teman kos Griya Sakinah yang sudah seperti keluarga kandung penulis selama 4 tahun lebih di Semarang.
12. Seluruh pengurus HIMASKOM tahun 2013 sampai 2016, khususnya Departemen Infokom dan Departemen Sosial.
13. Teman-teman Nafiri, PMK-FT, Call Center, dan KSR UNDIP yang sudah berbagi pengalaman dengan penulis.
14. Semua pihak yang tidak dapat penulis sebutkan satu persatu yang telah membantu selama ini.

Penulis menyadari bahwa masih banyak kekurangan di dalam penyusunan laporan ini, maka penulis mengharapkan kritik dan saran yang membangun untuk menyempurnakan laporan ini. Semoga laporan ini dapat memberikan manfaat bagi penulis maupun bagi rekan-rekan mahasiswa.

Semarang, 1 November 2016

Penulis

# DAFTAR ISI

Halaman Cover i

Halaman Pengesahan ii

Halaman Pernyataan Orisinalitas iii

Halaman Pernyataan Persetujuan Publikasi iv

Abstrak v

Abstract vi

Kata Pengantar vii

Daftar Isi ix

Daftar Gambar xi

Daftar Tabel xv

Daftar Istilah xvi

BAB I PENDAHULUAN

* 1. Latar Belakang 1
  2. Rumusan Masalah 2
  3. Batasan Masalah 2
  4. Tujuan Penelitian 3
  5. Manfaat Penelitian 3
  6. Metode Penelitian 4
  7. Sistematika Penulisan 5

BAB II TINJAUAN PUSTAKA

* 1. Penelitian Terdahulu 7
  2. Android Studio 8
  3. Bahasa Pemrograman Java 9
  4. GPS 10
  5. Basisdata MySQL 12
  6. Unified Modelling Language (UML) 13
  7. Metode Waterfall 18
  8. Web Service 19
  9. Pengujian Kotak Hitam 19

BAB III PERANCANGAN SISTEM

* 1. Analisis Kebutuhan 20

3.1.1 Analisis Kebutuhan Fungsional 20

3.1.2 Analisis Kebutuhan Non Fungsional 21

* 1. Perancangan Sistem Aplikasi 21

3.2.1 Diagram Use Case 21

3.2.2 Diagram Aktivitas 22

3.2.3 Diagram Sekuensial 34

3.2.4 Perancangan Basisdata 46

* 1. Desain Tampilan 50

BAB IV IMPLEMENTASI DAN PENGUJIAN

* 1. Tahap Pembuatan 62

4.1.1 Tahap Pembuatan Basisdata 62

4.1.2 Tahap Pembuatan Web Service 65

4.1.3 Tahap Pembuatan Aplikasi 75

* 1. Analisis dan Pengujian 102

BAB V PENUTUP

* 1. Kesimpulan 114
  2. Saran 114

DAFTAR PUSTAKA 115

LAMPIRAN 1 BIODATA MAHASISWA

LAMPIRAN 2 MAKALAH TUGAS AKHIR

# DAFTAR GAMBAR

Gambar 2.1 Metode Waterfall 18

Gambar 3.1 Use case diagram sistem 22

Gambar 3.2 Diagram aktivitas pendaftaran 24

Gambar 3.3 Diagram aktivitas login 24

Gambar 3.4 Diagram aktivitas konfirmasi Kode User 25

Gambar 3.5 Diagram aktivitas menampilkan peta lokasi kendaraan 26

Gambar 3.6 Diagram aktivitas menampilkan grafik kecepatan 27

Gambar 3.7 Diagram aktivitas menampilkan notifikasi motor jatuh 27

Gambar 3.8 Diagram aktivitas ubah profil 28

Gambar 3.9 Diagram aktivitas ubah password 29

Gambar 3.10 Diagram aktivitas fitur lock 30

Gambar 3.11 Diagram aktivitas menu how to use 31

Gambar 3.12 Diagram aktivitas menampilkan data record 31

Gambar 3.13 Diagram aktivitas menu about 32

Gambar 3.14 Diagram aktivitas logout 33

Gambar 3.15 Diagram aktivitas admin 33

Gambar 3.16 Diagram aktivitas alat 34

Gambar 3.17 Diagram sekuensial pendaftaran 35

Gambar 3.18 Diagram sekuensial login 36

Gambar 3.19 Diagram sekuensial konfirmasi Kode User 36

Gambar 3.20 Diagram sekuensial maps 37

Gambar 3.21 Diagram sekuensial kecepatan 38

Gambar 3.22 Diagram sekuensial notifikasi motor jatuh 38

Gambar 3.23 Diagram sekuensial lock 39

Gambar 3.24 Diagram sekuensial ubah profil 40

Gambar 3.25 Diagram sekuensial ubah password 41

Gambar 3.26 Diagram sekuensial petunjuk aplikasi 42

Gambar 3.27 Diagram sekuensial rekaman data 43

Gambar 3.28 Diagram sekuensial tentang aplikasi 43

Gambar 3.29 Diagram sekuensial logout 44

Gambar 3.30 Diagram sekuensial admin 45

Gambar 3.31 Diagram sekuensial alat 45

Gambar 3.32 Desain ERD aplikasi 46

Gambar 3.33 Desain basisdata relasi antar tabel 47

Gambar 3.34 Tampilan halaman login 50

Gambar 3.35 Tampilan Halaman Register 51

Gambar 3.36 Tampilan halaman utama aplikasi 52

Gambar 3.37 Tampilan halaman maps 53

Gambar 3.38 Tampilan halaman kecepatan 53

Gambar 3.39 Tampilan halaman lock ketika motor hidup 54

Gambar 3.40 Tampilan halaman lock ketika motor mati 55

Gambar 3.41 Tampilan halaman edit profil 55

Gambar 3.42 Tampilan halaman ubah password 56

Gambar 3.43 Tampilan slidebar menu 57

Gambar 3.44 Tampilan halaman petunjuk aplikasi 57

Gambar 3.45 Tampilan menu Data record 58

Gambar 3.46 Tampilan halaman tentang aplikasi 59

Gambar 3.47 Tampilan notifikasi motor jatuh 60

Gambar 3.48 Tampilan konfirmasi keluar aplikasi 60

Gambar 3.49 Tampilan halaman admin 61

Gambar 4.1 Layanan login 66

Gambar 4.2 Layanan register\_user 67

Gambar 4.3 Layanan read\_user 67

Gambar 4.4 Layanan a.php 68

Gambar 4.5 Layanan read\_a 68

Gambar 4.6 Layanan read\_location 69

Gambar 4.7 Layanan read\_speed 69

Gambar 4.8 Layanan read\_record 69

Gambar 4.9 Layanan relay 70

Gambar 4.10 Layanan read\_status 70

Gambar 4.11 Layanan update\_profil 71

Gambar 4.12 Layanan update\_password 72

Gambar 4.13 Layanan update\_status 72

Gambar 4.14 Layanan register\_device 73

Gambar 4.15 Layanan send\_notification 73

Gambar 4.16 Layanan read\_alat 74

Gambar 4.17 Layanan update\_alat 75

Gambar 4.18 Halaman register perangkat Android Lenovo 76

Gambar 4.19 Halaman register perangkat Android Samsung 77

Gambar 4.20 Halaman login perangkat Android Lenovo 78

Gambar 4.21 Halaman login perangkat Android Samsung 78

Gambar 4.22 Verifikasi pengguna 1 perangkat Android Lenovo 79

Gambar 4.23 Verifikasi pengguna 2 perangkat Android Samsung 79

Gambar 4.24 Halaman utama perangkat Android Lenovo 80

Gambar 4.25 Halaman utama perangkat Android Samsung 81

Gambar 4.26 Perintah untuk mengisi Kode User 82

Gambar 4.27 Profile pengguna 1 pada perangkat Android Lenovo 82

Gambar 4.28 Profile pengguna 2 pada perangkat Android Samsung 83

Gambar 4.29 Map pada perangkat Android Lenovo 84

Gambar 4.30 Map pada perangkat Android Samsung 85

Gambar 4.31 Kecepatan pada perangkat Android 1 hari 1 86

Gambar 4.32 Kecepatan pada perangkat Android 2 hari 2 87

Gambar 4.33 Lock ketika mesin hidup pengguna 1 88

Gambar 4.34 Lock ketika mesin hidup pengguna 2 89

Gambar 4.35 Lock ketika mesin mati pengguna 1 89

Gambar 4.36 Lock ketika mesin mati pengguna 2 90

Gambar 4.37 Ubah Password perangkat Android Lenovo 91

Gambar 4.38 Ubah Password perangkat Android Samsung 92

Gambar 4.39 Petunjuk pada perangkat Android Lenovo 93

Gambar 4.40 Petunjuk pada perangkat Android Samsung 93

Gambar 4.41 Data record pada perangkat Android Lenovo 94

Gambar 4.42 Data record pada perangkat Android Samsung 95

Gambar 4.43 Tentang aplikasi pada perangkat Android Samsung 96

Gambar 4.44 Tentang aplikasi pada perangkat Android Lenovo 96

Gambar 4.45 Notifikasi pada perangkat Android Lenovo 97

Gambar 4.46 Notifikasi pada perangkat Android Samsung 98

Gambar 4.47 Logout pada perangkat Android Lenovo 99

Gambar 4.48 Logout pada perangkat Android Samsung 99

Gambar 4.49 Status alat tidak aktif 100

Gambar 4.50 Konfirmasi untuk mengaktifkan alat 101

Gambar 4.51 Status alat aktif 101

# DAFTAR TABEL

Tabel 2.1 Notasi diagram use case 14

Tabel 2.2 Notasi class diagram 15

Tabel 2.3 Notasi statechart diagram 16

Tabel 2.4 Notasi diagram sequence 17

Tabel 3.1 Kamus data user 48

Tabel 3.2 Kamus data ditrol 48

Tabel 3.3 Kamus data bacarelay 49

Tabel 3.4 Kamus data devices 49

Tabel 3.5 Kamus data alat 50

Tabel 4.1 Pengujian Login 102

Tabel 4.2 Pengujian Register 104

Tabel 4.3 Pengujian Halaman Utama 105

Tabel 4.4 Pengujian Slidebar Menu 106

Tabel 4.5 Pengujian menu Lock Saat Motor Hidup 106

Tabel 4.6 Pengujian menu Lock Saat Motor Mati 107

Tabel 4.7 Pengujian menu Profile 108

Tabel 4.8 Pengujian UBAH PASSWORD 109

Tabel 4.9 Pengujian menu Data record 110

Tabel 4.10 Pengujian Notifikasi Motor Jatuh 111

Tabel 4.11 Pengujian Speed 112

Tabel 4.12 Pengujian Logout 112

Tabel 4.13 Pengujian Admin 113

# DAFTAR ISTILAH

WHO : *World Heatlh Organization*

GPS : *Global Positioning System*

SDLC : *Software Development Life Cycle*

UML : *Unified Modelling Language*

SIG : Sistem Informasi Geografis

SMS : *Short Message Service*

GPRS : *General Packet Radio Service*

APK : *Application Package File*

DBMS : *Database Management System*

RDBMS : *Relationship Database Management System*

SQL : *Structured Query Language*

SP : *Stored Prosedured*

P2P : *Program-to-Program*

B2C : *Business-to-Consume*r

REST : *Representational State Transfer*

S3 : *Simple Storage Service*

HTTP : *Hypertext Transfer Protocol*

JDK : *Java Development Kit*

ERD : *Entity Relationship Diagram*

GUI : *Graphical User Interface*

**BAB I  
PENDAHULUAN**

* 1. **Latar Belakang**

Kepadatan lalu lintas bisa terjadi karena mobilitas yang tinggi. Mobilitas yang tinggi menuntut masyarakat agar tidak tertinggal oleh kemajuan zaman, salah satunya yaitu dengan memiliki kendaraan. Tetapi tidak sedikit orang terbunuh di jalan akibat kepadatan lalu lintas dan kesalahan manusia. Organisasi Kesehatan Dunia (WHO) memperkirakan bahwa di tahun 2020 penyebab terbesar ketiga kematian adalah kecelakaan jalan raya, tepat di bawah penyakit jantung dan depresi. WHO mencatat bahwa 1 juta orang di seluruh dunia meninggal setiap tahunnya di jalan raya akibat kecelakaan, dimana 40% di antaranya berusia dibawah 25 tahun. [1]

Di Indonesia angka pencurian sepeda motor cukup tinggi, pada tahun 2012 di Jakarta Timur terdapat 422 kasus pencurian sepeda motor sedangkan di kota Malang, Jawa Timur terdapat 1.200 kasus pencurian sepeda motor dan pada tahun 2013 mencapai 1.188 kasus.[2] Hal ini membuat masyarakat semakin khawatir untuk bepergian dan saat parkir kendaraan sepeda motor ditempat umum. Keinginan utama para orang tua tentu saja melindungi keluarga terutama anak-anak dari bahaya, namun hal ini sulit untuk dilakukan oleh orang tua yang sibuk bekerja karena mereka tidak memiliki waktu yang optimal untuk mengawasi anak mereka secara langsung terutama saat anak-anak mereka berkendara.

Berdasarkan latar belakang diatas diperlukan sebuah aplikasi yang mampu untuk membantu orang tua untuk mengawasi anak berkendara dan mendeteksi pencurian sepeda motor.Aplikasi *Driver Control* akan memberi informasi kepada orang tua berupa posisi atau lokasi sepeda motor, kecepatan rata-rata saat anak berkendara, dan memberikan notifikasi ketika motor jatuh, dengan ide ini diharapkan dapat membantu orang tua untuk mengawasi anak mereka saat berkendara, dan membantu pemerintah dalam membuat inovasi baru sebagai upaya untuk mengurangi jumlah kecelakaan lalu lintas.

* 1. **Rumusan Masalah**

Perumusan masalah dalam pembuatan aplikasi Android “*Driver Control*” sebagai sarana memonitor anak berkendara secara waktu-nyata adalah bagaimana membuat aplikasi pada teknologi canggih seperti telepon pintar Android yang mampu menyediakan informasi lokasi kendaraan sepeda motor, kecepatan rata-rata mengemudi, dan memberikan notifikasi ketika motor jatuh.

* 1. **Batasan Masalah**

Untuk menghindari pembahasan yang meluas dalam pembuatan aplikasi ini maka dalam Tugas Akhir ini ditetapkan batasan-batasan masalah dengan hal-hal sebagai berikut:

1. Aplikasi ini dikembangkan berbasis *mobile* untuk sistem operasi Android.
2. Perangkat lunakpengembangan aplikasi yang digunakan adalah Android Studio dengan menggunakan bahasa pemrograman Java dan basisdata MySQL melalui *Web Service*.
3. Aplikasi ini dapat digunakan oleh beberapa pengguna untuk memantau satu motor yang sudah terpasang alat *Driver Control*.
4. Aplikasi hanya dapat digunakan oleh pengguna yang memiliki kode alat dan kode *user*.
5. Aplikasi mampu menampilkan lokasi motor terakhir pada Google Maps.
6. Aplikasi mampu menampilkan grafik kecepatan rata-rata pengemudi sepeda motor dan rekaman data pengemudi dalam waktu 1 minggu terakhir.
7. Pengguna dapat mematikan atau menghidupkan mesin motor secara otomatis melalui aplikasi.
8. Aplikasi dapat memberi notifikasi motor jatuh.
9. Aplikasi ini membutuhkan koneksi Internet dan GPS sehingga perangkat *Driver Control* hanya dapat mengirim data pada ruang terbuka.
10. Aplikasi memperoleh data dari perangkat *Driver Control* yang merupakan bagian dari Tugas Akhir yang dibuat oleh saudari Khoirunisa Waskitaningrum.
11. Penelitian ini bermaksud merancang dan membuat aplikasi untuk memonitor anak berkendara secara waktu-nyata tanpa membahas kecurangan-kecurangan yang dilakukan oleh anak.
    1. **Tujuan Penelitian**

Tujuan dari Tugas Akhir ini adalah untuk membuat aplikasi Android yang dapat membantu orang tua untuk memonitor anak berkendara secara waktu-nyata dengan menampilkan lokasi kendaraan, kecepatan berkendara, rekaman data saat mengemudi, notifikasi ketika motor jatuh, dan pengguna dapat mematikan mesin motor secara otomatis melalui aplikasi yang dirancang menggunakan perangkat lunak Android Studio, dan basisdata MySQL melalui *Web Service*.

* 1. **Manfaat Penelitian**

Manfaat yang diperoleh dari Tugas Akhir ini adalah sebagai berikut :

1. Manfaat bagi penulis
2. Memiliki pengalaman dalam merancang dan mengimplementasikan rancangan perangkat lunak ke dalam suatu sistem.
3. Mengimplementasikan ilmu yang didapat untuk membuat aplikasi Android yang memudahkan orang tua dalam memantau lokasi kendaraan dan pengemudi sepeda motor.
4. Manfaat bagi Universitas Diponegoro

Mengimplementasikan Tri Dharma Perguruan Tinggi untuk menyelesaikan permasalahan aktual di masyarakat, sebagai contoh tingginya angka kecelakaan lalu lintas dan banyaknya kasus pencurian sepeda motor.

1. Manfaat bagi pengguna perangkat lunak
2. Orang tua dapat memantau anak berkendara.
3. Orang tua atau pengguna dapat mengetahui informasi lokasi kendaraan, kecepatan rata-rata pengemudi sepeda motor, dan mendapatkan notifikasi ketika motor jatuh.
4. Manfaat bagi pemerintah

Menciptakan inovasi baru untuk mengurangi angka kecelakaan lalu lintas dan pencurian sepeda motor.

* 1. **Metode Penelitian**

Metode yang digunakan untuk penyusunan laporan Tugas Akhir ini adalah sebagai berikut :

1. Studi Pustaka dan Bimbingan

Tahap ini merupakan tahapan untuk mendapatkan landasan teori, data atau informasi sebagai bahan acuan dalam melakukan perencanaan, percobaan, pembuatan, dan penyusunan Tugas Akhir. Bimbingan bertujuan untuk mendapatkan pengetahuan tambahan, masukan dari dosen pembimbing, dan koreksi kesalahan dalam pembuatan laporan.

1. Analisis dan Spesifikasi Kebutuhan

Tahap ini merupakan tahapan untuk melakukan analisis dan menjabarkan kebutuhan yang diperlukan dalam melakukan penelitian. Kebutuhan yang diperlukan tersebut akan dianalisis tujuan dan manfaatnya dalam melakukan penelitian, kemudian kebutuhan tersebut akan digunakan pada tahap perancangan.

1. Perancangan

Tahap ini merupakan tahapan untuk membuat rancangan sistem aplikasi Android *Driver Control* sebagai sarana memonitor anak berkendara secara waktu-nyata, sehingga akan diperoleh rancangan sistem yang baik dan memenuhi tujuan pembuatan aplikasi. Perancangan aplikasi ini menggunakan *Software Development Life Cycle* (SDLC). SDLC merupakan proses mengembangkan atau mengubah suatu sistem perangkat lunak dengan menggunakan atau mengubah suatu sistem perangkat lunak dengan menggunakan model-model dan metodologi yang digunakan orang untuk mengembangkan sistem-sistem perangkat lunak sebelumnya. Salah satu model yang dimiliki oleh SDLC adalah metode *waterfall.* Metode *waterfall* menyediakan pendekatan alur hidup perangkat lunak secara terurut dimulai dari analisis, desain, pengodean, dan pengujian.

1. Implementasi

Tahap ini membahas tentang implementasi hasil perancangan sistem yang kemudian dibangun kedalam sebuah perangkat bergerak Android. Implementasinya yaitu bagaimana mewujudkan konsep yang telah dirancang, sehingga menghasilkan suatu aplikasi Android yang dapat memonitor anak berkendara secara waktu-nyata dan semua fungsi fitur yang ada pada aplikasi dapat berfungsi dengan baik.

1. Pengujian

Tahap pengujian ini akan menguji sistem secara keseluruhan apakah perangkat lunak yang dibuat telah dapat berjalan dengan baik dan sesuai dengan tujuan yang ingin dicapai. Pengujian dilakukan terhadap aplikasi untuk mengidentifikasikan dan memperbaiki kesalahan yang ada. Hasil dari tahapan ini adalah aplikasi yang sudah terbebas dari kesalahan-kesalahan dan siap digunakan.

* 1. **Sistematika Penulisan**

Laporan Tugas Akhir ini disusun menurut sistematika penulisan yang terdiri atas lima bab dengan susunan sebagai berikut:

**BAB I PENDAHULUAN**

Bab ini berisi tentang latar belakang, rumusan masalah, pembatasan masalah, tujuan Tugas Akhir, manfaat penelitian, metode penelitian, dan sistematika penulisan.

**BAB II TINJAUAN PUSTAKA**

Bab ini berisi uraian pengertian dan konsep-konsep dasar yang menjadi pedoman dalam penelitian dan pembuatan aplikasi Android *Driver Control* sebagai sarana memonitor anak berkendara secara waktu-nyata. Uraian yang ada dalam landasan teori di antaranya mengenai Android Studio, Bahasa Pemrograman Java, *Global Positioning System* (GPS), Basisdata MySQL, *Unified Modelling Language* (UML), metode *Waterfall, Web Service,* danPengujian Kotak Hitam.

**BAB III PERANCANGAN SISTEM**

Bab ini berisi tentang perancangan aplikasi Android *Driver Control* sebagai sarana memonitor anak berkendara secara waktu-nyata yang berisi mengenai kebutuhan dalam membuat aplikasi, desain sistem dan desain tampilan dari aplikasi yang akan dibuat.

**BAB IV IMPLEMENTASI DAN PENGUJIAN SISTEM**

Bab ini berisi tentang implementasi dan pengujian aplikasi Android *Driver Control* sebagai sarana memonitor anak berkendara secara waktu-nyata*.* Pengujiannya meliputi apakah fungsi setiap fitur yang ada dalam aplikasi telah berjalan dengan baik atau tidak.

**BAB V PENUTUP**

Bab ini berisi tentang kesimpulan mengenai aplikasi yang telah dibangun apakah telah sesuai dengan tujuan perancangannya serta saran untuk perbaikan maupun pengembangan aplikasi di masa yang akan datang.

**BAB II  
TINJAUAN PUSTAKA**

2. 1. **Penelitian Terdahulu**

Penulis melakukan penelitian ini didorong oleh hasil penelitian lain yang pernah dilakukan sebelumnya berkaitan dengan aplikasi pencarian lokasi dan aplikasi yang mengimplementasikan *Web Service* sebagai berikut:

1. Raidah Hanifah[3] melakukan penelitian mengenai Simulasi Sistem Informasi Geografis (SIG) Pemantauan Posisi Kendaraan *Via* SMS *Gateway*. Penelitian ini bertujuan untuk membangun simulasi sistem pemantauan posisi kendaraan yang dapat memvisualisasikan posisi kendaraan berdasarkan koordinat bujur dan lintang ke dalam peta digital dan dapat menampilkan jalur perjalanan kendaraan. Sistem dibuat menggunakan perangkat lunak Gammu 1.25 untuk mengirimkan SMS *Gateway*, MapObject 2.0 untuk menampilkan peta, dan bahasa pemrograman Delphi. Hasil keluaran penelitian ini adalah Sistem Informasi Geografis yang dapat menampilkan peta lokasi, jalur yang dilewati kendaraan dan dikontrol melalui SMS *Gateway*.
2. Rusnandar dkk[4] melakukan penelitian mengenai Sistem Pelacak Kendaraan Berbasis OpenGTS. Penelitian ini bertujuan untuk membangun sistem pelacak kendaraan dalam upaya meningkatkan keamanan dan mutu komoditas industri pengiriman layanan barang dan jasa. Sistem dibuat menggunakan GPS, GPRS, dan perangkat lunak OpenGTS untuk mengirim SMS kepada pengguna dan alat pelacak. Hasil keluaran penelitian ini adalah Sistem Informasi yang dapat menampilkan posisi, kecepatan beberapa kendaraan yang disajikan dalam Google Maps berbasis *Web* dan dikontrol melalui SMS*.*
3. Hendra Pria Utama[5] melakukan penelitian mengenai Pembuatan Aplikasi Memantau Lokasi Anak Berbasis Android Menggunakan *Location Based Service*.

Penelitian ini bertujuan untuk membuat aplikasi *Location Based Service* untuk menampilkan lokasi anak yang dapat diakses melalui perangkat bergerak. Aplikasi dibuat menggunakan basisdata MySQL, perangkat lunak Android Studio, bahasa pemrograman Java, dan menggunakan *Web Service*. Hasil keluaran penelitian ini adalah aplikasi *Location Based Service* untuk menampilkan lokasi anak yang dapat diakses melalui perangkat bergerak telepon pintar Android dan mengimplentasikan *Web Service*.

Perbedaan ketiga penelitian tersebut dengan penelitian yang dilakukan oleh penulis adalah penulis membahas bagaimana cara membuat aplikasi Android *Driver* Control untuk memonitor anak berkendara secara waktu-nyata yang terhubung oleh basisdata *server* melalui *Web Service*. Hasil keluaran yang diperoleh adalah aplikasi Android yang mampu menujukkan lokasi kendaraan, grafik kecepatan pengemudi, status motor hidup atau mati, memberikan notifikasi ketika motor jatuh, memberikan rekaman data pengemudi, dan pengguna dapat mematikan atau menghidupkan mesin motor secara otomatis melalui aplikasi.

* 1. **Android Studio**

Android Studio adalah sebuah perangkat lunak baru untuk pengembangan aplikasi berbasis Android yang berbasis IntelliJ IDEA. Android Studio menyediakan banyak fitur dan peningkatan dari Eclipse ADT dan akan menjadi Android IDE yang resmi. Beberapa fitur yang ada pada Android Studio adalah sebagai berikut:

1. Sistem *build* yang fleksibel berbasis Gradle,
2. Mempunyai banyak macam generasi APK dan variasi *build*,
3. Mendukung *template* untuk layanan Google dan berbagai macam tipe perangkat,
4. Tampilan editor yang kaya dan juga mendukung untuk perubahan tema,
5. Menyediakan peralatan Lint untuk untuk merekam kinerja, penggunaan, kompabilitas versi, dan permasalahan lainnya,
6. Mempunyai fasilitas ProGuard dan *app-signing* untuk aplikasi,
7. Mendukung Google Cloud Platform yang memudahkan untuk melakukan integrasi Google Cloud Messaging dan App Engine.

Berikut penjelasan fungsi dari bagian-bagian yang berada di perangkat lunak Android Studio :

1. *Title Bar* adalah baris yang menunjukkan nama projek yang berada di layar utama.
2. *Menu Bar* adalah kumpulan menu yang berisi daftar perintah untuk pengolahan data yang akan atau sedang dibuat.
3. *Tool Bar* adalah kumpulan menu yang memiliki fungsi-fungsi spesifik untuk mengolah projek yang sedang dikerjakan.
4. *Navigation Bar* adalah baris yang menunjukkan navigasi direktori dari jendela yang sedang dibuka.
5. *Project Bar* menunjukkan ikhtisar dari semua yang ada di dalam *project*.
6. *Structure Bar* menujukkan struktur isi dari file yang sedang dibuka.
7. *Commander Bar* menunjukkan direktori folder pada projek beserta strukturnya dalam dua tampilan sekaligus.
8. *Gradle Bar* untuk menunjukkan *task* yang akan dibangun.
9. *Android Bar* menunjukkan perangkat yang akan digunakan untuk menampilkan hasil program setelah dibangun serta terdapat tampilan log program. [6]
   1. **Bahasa Pemrograman Java**

Bahasa pemrograman Java adalah sebuah bahasa pemrograman umum berorientasi objek yang pertama kali diluncurkan pada tahun 1990-an oleh James Gosling dan Sun Microsystem. Java memiliki beberapa keunggulan bila dibandingkan dengan bahasa pemrograman lainnya yaitu sebagai berikut:

1. Java bersifat lebih sederhana dan relatif mudah dimodelkan sebagian dari bahasa C++, namun dengan memperbaiki beberapa karakteristik C++, seperti mengurangi kompleksitas beberapa fitur, penambahan fungsionalitas, serta penghilangan beberapa aspek pemicu ketidakstabilan system C++.
2. Java adalah suatu konsep pemrograman yang memecahkan masalah dengan cara memilah program menjadi objek-objek yang saling berinteraksi satu sama lain.
3. Java bersifat *multiplatform* karena dapat diterjemahkan oleh Java *interpreter* pada berbagai sistem operasi.
4. Java bersifat *multithread* yang dapat mengerjakan beberapa proses dalam waktu yang hampir bersamaan.[7]
   1. **GPS**

GPS adalah singkatan dari *Global Positioning System*, yang merupakan sistem navigasi dengan menggunakan teknologi satelit yang dapat menerima sinyal dari satelit. Sistem ini menggunakan 24 satelit yang mengirimkan sinyal gelombang mikro ke bumi. Sinyal ini diterima oleh alat penerima di permukaan, dimana GPS penerima ini akan mengumpulkan informasi dari satelit GPS, seperti:

1. Waktu. GPS penerima menerima informasi waktu dari jam atom yang mempunyai keakurasian sangat tinggi.
2. Lokasi. GPS memberikan informasi lokasi dalam tiga dimensi:

1) Latitude (garis lintang)

2) Longitude (garis bujur)

3) Elevasi (ketinggian)

1. Kecepatan. Ketika berpindah tempat, GPS dapat menunjukkan informasi kecepatan berpindah tersebut.
2. Arah perjalanan. GPS dapat menunjukkan arah tujuan.
3. Simpan lokasi. Tempat-tempat yang sudah pernah atau ingin dikunjungi bisa disimpan oleh GPS penerima.
4. Komulasi data. GPS penerima dapat menyimpan informasi jejak, seperti total perjalanan yang sudah pernah dilakukan, kecepatan rata-rata, kecepatan paling tinggi, kecepatan paling rendah, waktu atau jam sampai tujuan, dan sebagainya.

Posisi yang ditunjukkan oleh suatu GPS mempunyai faktor kesalahan atau juga disebut tingkat akurasi. Sebagai contoh suatu alat GPS menunjukkan titik koordinat dengan tingkat akurasi 5 meter, berarti posisi pengguna bisa berada dalam kisaranradius 5 meter dari titik yang ditunjukkan tersebut. Ada beberapa hal yang mempengaruhi tingkat akurasi tersebut, antara lain:

1. Kesalahan Ephemeris. Terjadi jika satelit tidak dapat mentransmisikan posisinya di orbit dengan tepat.
2. Keadaan Ionosfer. Ionosfer berada pada jarak sekitar 43-50 mil di atas permukaan bumi. Satelit yang melewati Ionosfer akan menjadi lambat dikarenakan adanya plasma atau gas dengan tingkat kepadatan rendah. Walaupun GPS penerima berusaha untuk mengkoreksi atau memperbaiki faktor keterlambatan yang terjadi tetap saja aktivitas tertentu dari plasma bisa menyebabkan kesalahan perhitungan.
3. Keadaan Troposfer. Troposfer adalah bagian terendah dari atmosfer sampai dengan ketinggian sekitar 11 mil dari permukaan tanah. Variasi pada temperatur, tekanan, dan kelembaban bisa menyebabkan perbedaan kecepatan penerimaan gelombang radio.
4. Kesalahan Waktu. Karena penempatan jam atom pada setiap GPS penerima tidak berjalan sebagaimana mestinya. Kesalahan waktu dari GPS penerima yang tidak presisi dapat menimbulkan ketidakakurasian.
5. Kesalahan Multipath. Terjadi karena sinyal satelit membentur permukaan keras seperti bangunan atau tebing sebelum mencapai GPS penerima. Hal tersebut bisa menyebabkan terjadinya waktu-tunda sehingga perhitungan jarak menjadi tidak akurat.
6. Buruknya Sinyal Satelit. Keadaan langit yang terhalang akan menyebabkan GPS sulit menerima data satelit. Sebuah sinyal satelit yang pada hari tertentu diterima dengan sangat bagus belum tentu pada hari lain bisa diterima dengan kualitas yang sama walaupun pengguna berdiri pada tempat yang sama. Hal tersebut dikarenakan posisi dari satelit yang terus bergerak atau bisa juga disebabkan faktor penghalang lain seperti pohon, gedung bertingkat, dan sebagainya. [8]
   1. **Basisdata MySQL**

Basisdata adalah kumpulan informasi yang disimpan di dalam komputer secara sistematik sehingga dapat diperiksa menggunakan suatu program komputer untuk memperoleh informasi dari basisdata tersebut.

*Database Management System* (DBMS) merupakan suatu sistem perangkat lunak yang memungkinkan pengguna untuk membuat, memelihara, mengontrol, dan mengakses basisdatasecara praktis dan efisien. Sedangkan *Relationship Database Management System* (RDBMS) merupakan salah satu jenis DBMS yang mendukung adanya hubungan antar tabel.

MySQL adalah sebuah perangkat lunak sistem manajemen basisdata SQL yang multithread, multi-user, dengan sekitar 6 juta instalasi di seluruh dunia. Fitur- fitur MySQL antara lain:

1. *Relational Database System*. Seperti halnya perangkat lunak basisdata lain yang ada di pasaran, MySQL termasuk RDBMS.
2. Arsitektur *Client-Server*. MySQL memiliki arsitektur *client-server* dimana *server* basisdata MySQL terinstal di *server*. *Client* MySQL dapat berada di komputer yang sama dengan *server*, dan dapat juga di komputer lain yang berkomunikasi dengan *server* melalui jaringan bahkan Internet.
3. Mengenal perintah SQL standar. *Structured Query Language* (SQL) merupakan suatu bahasa standar yang berlaku di hampir semuaperangkat lunak basisdata. MySQL mendukung SQL versi SQL 2003.
4. Mendukung *Sub Select*. Mulai versi 4.1 MySQL telah mendukung *select* dalam *select* atau *sub select*.
5. Mendukung *Views*. MySQL mendukung *views* sejak versi 5.0.
6. Mendukung *Stored Prosedured* (SP). MySQL mendukung SP sejak versi 5.0.
7. Mendukung *Triggers*. MySQL mendukung *trigger* pada versi 5.0 namun masih terbatas. Pengembang MySQL berjanji akan meningkatkan kemampuan *trigger* pada versi 5.1.
8. Mendukung replikasi.
9. Mendukung transaksi.
10. Mendukung *foreign key*.
11. Tersedia fungsi GIS.
12. Gratis, stabil, dan tangguh.
13. Fleksibel dengan berbagai pemrograman.
14. Keamanan yang baik dan terdapat dukungan dari banyak komunitas.
15. Perkembangan perangkat lunak yang cukup cepat. [9]
    1. ***Unified Modelling Language* (UML)**

*Unified Modelling Language* (UML) merupakan metode pemodelan secara visual sebagai sarana untuk merancang atau membuat perangkat lunak berorientasi objek. Karena UML ini merupakan bahasa visual untuk pemodelan bahasa berorientasi objek, maka semua elemen dan diagram berbasiskan pada paradigma berbasis objek. Dengan menggunakan UML pengguna dapat membuat model untuk semua jenis aplikasi perangkat lunak, dimana aplikasi tersebut dapat berjalan pada perangkat keras, sistem operasi dan jaringan apapun, serta ditulis dalam bahasa pemrograman apapun. Tetapi karena UML juga menggunakan kelasdan operasidalam konsep dasarnya, maka lebih cocok untuk penulisan perangkat lunak dalam bahasa-bahasa berorientasi objek seperti C++, Java, C# atau VB.NET. Walaupun demikian, UML tetap dapat digunakan untuk model aplikasi prosedural dalam VB atau C. UML sendiri terdiri atas pengelompokkan diagram-diagram sistem menurut aspek atau sudut pandang tertentu. Diagram adalah yang menggambarkan permasalahan maupun solusi dari permasalahan suatu model. [10] Berikut ini merupakan contoh model diagram UML :

1. *Diagram Use Case*

Diagram *use case* menggambarkan apa saja aktifitas yang dilakukan oleh suatu sistem dari sudut pandang pengamatan luar. Diagram *use case* dekat kaitannya dengan kejadian-kejadian. Kejadian atau *scenario* merupakan contoh apa yang terjadi ketika seseorang atau alat berinteraksi dengan sistem. Berikut adalah notasi-notasi pada diagram *use case,* yang ditunjukan pada Tabel 2.1.

Tabel 2.1 Notasi diagram *use case*

|  |  |  |
| --- | --- | --- |
| **Notasi** | **Keterangan** | **Simbol** |
| *Actor* | *Actor* merupakan pengguna sistem, sistem yang berhubungan dengan sistem lain, atau waktu. *Actor* tidak terbatas hanya manusia saja, jika sebuah sistem berkomunikasi dengan aplikasi lain dan membutuhkan input atau mengeluarkan output, maka aplikasi tersebut juga bisa dianggap sebagai *actor*. |  |
| *Use Case* | *Use case* merupakan sebuah pekerjaan tertentu, misalnya masuk ke dalam sistem, membuat sebuah daftar belanja, dan sebagainya yang disediakan sistem. |  |
| Asosiasi | Menghubungkan antara *actor* dengan *use case* |  |
| Sistem | Suatu kesatuan yang terdiri [komponen](https://id.wikipedia.org/w/index.php?title=Komponen&action=edit&redlink=1) atau [elemen](https://id.wikipedia.org/wiki/Elemen) yang dihubungkan bersama untuk memudahkan aliran [informasi](https://id.wikipedia.org/wiki/Informasi), [materi](https://id.wikipedia.org/wiki/Materi) atau [energi](https://id.wikipedia.org/wiki/Energi) untuk mencapai suatu tujuan. Sistem menggambarkan suatu set entitas yang berinteraksi. |  |

Tipe Relasi/Asosiasi pada *Use Case* :

1. Include

Memungkinkan 1 *use case* menggunakan fungsionalitas yang disediakan oleh *use case* lain.

1. Extends

Memungkinkan 1 *use case* secara optional menggunakan fungsionalitas yang disediakan oleh *use case* lain.

1. Generalization/inheritance antara *use case*

Dibuat ketika ada sebuah keadaan yang lain atau perlakuan khusus *Inheriting use case* dibawah *base* atau *parent use case*.

1. *Class Diagram*

*Class* adalah sebuah spesifikasi yang jika diinstansiasi akan menghasilkan sebuah objek dan merupakan inti dari pengembangan dan desain berorientasi objek. *Class* menggambarkan keadaan atribut atau properti suatu sistem, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut yang disebut metode atau fungsi. *Class diagram* menggambarkan struktur dan deskripsi *class, package* dan objek beserta hubungan satu sama lain seperti *containment*, pewarisan, asosiasi, dan lain-lain. Berikut adalah notasi-notasi pada *class diagram,* yang ditunjukan pada Tabel 2.2.

Tabel 2.2 Notasi *class diagram*

|  |  |  |
| --- | --- | --- |
| **Notasi** | **Keterangan** | **Simbol** |
| *Class* | *Class* adalah blok-blok pembangun pada pemrograman berorientasi objek. |  |
| Asosiasi | Asosiasi, yaitu hubungan statis antar *class*. Umumnya menggambarkan *class* yang memiliki atribut berupa *class* lain, atau *class* yang harus mengetahui eksistensi *class* lain. |  |
| Agregasi | Agregasi, yaitu hubungan yang menyatakan bagian (“terdiri atas..”) atau biasa disebut relasi mempunyai sebuah. |  |
| *Composition* | *Composition*, yaitu sebuah *class* tidak bisa berdiri sendiri dan harus merupakan bagian dari *class* yang lain, maka *class* tersebut memiliki relasi *composition*. |  |
| *Dependency* | *Dependency*, yaitu hubungan yang menunjukan operasi pada suatu *class* yang menggunakan *class* yang lain. |  |

Tabel 2.2 Notasi *class diagram* (lanjutan)

|  |  |  |
| --- | --- | --- |
| Generalisasi | Generalisasi atau pewarisan, yaitu hubungan hirarki antar *class*. *Class* dapat diturunkan dari *class* lain dan mewarisi semua atribut dan metode *class* asalnya dan menambahkan fungsionalitas baru, sehingga disebut anak dari *class* yang diwarisinya. |  |

1. *Statechart Diagram*

*Statechart* atau biasa disebut dengan *state diagram* digunakan untuk mendokumentasikan beragam kondisi atau keadaan yang terjadi terhadap sebuah *class* dan kegiatan apa saja yang dapat merubah keadaan atau kondisi tersebut. Berikut adalah notasi-notasi pada *Statechart Diagram,* yang ditunjukan pada Tabel 2.3.

Tabel 2.3 Notasi *statechart diagram*

|  |  |  |
| --- | --- | --- |
| **Notasi** | **Keterangan** | **Simbol** |
| *State* | Notasi *state* menggambarkan kondisi sebuah entitas, dan digambarkan dengan segi empat yang keempat sudutnya tumpul berisi nama *state* di dalamnya. |  |
| *Transition* | *Transition* menggambarkan sebuah perubahan kondisi objek yang disebabkan oleh sebuah *event*. *Transition* digambarkan dengan sebuah anak panah dengan nama *event* yang ditulis di atasnya, atau di bawahnya, atau di sepanjang anak panah. |  |
| *Initial state* | *Initial state* adalah sebuah kondisi awal sebuah objek sebelum ada perubahan keadaan. *Initial state* digambarkan dengan sebuah lingkaran solid. |  |

Tabel 2.3 Notasi *statechart diagram* (lanjutan)

|  |  |  |
| --- | --- | --- |
| *Final state* | *Final state* menggambarkan ketika objek berhenti memberi respon terhadap sebuah *event. Final state* digambarkan dengan lingkaran solid didalam sebuah lingkaran kosong. |  |

1. *Diagram Sequence*

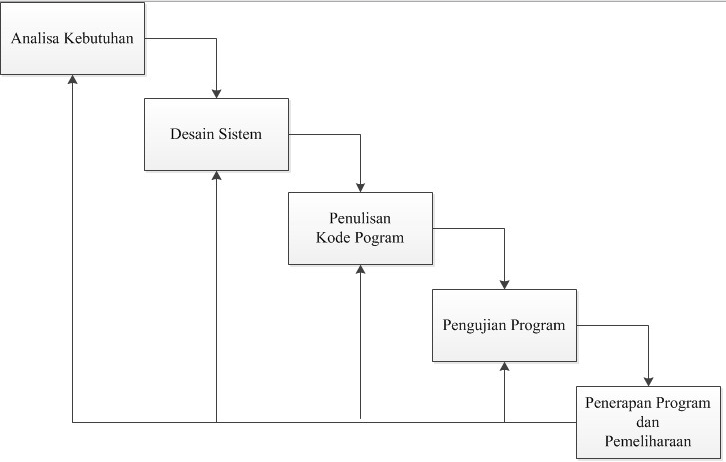
*Diagram sequence* merupakan salah satu diagram interaksi yang menjelaskan bagaimana suatu operasi itu dilakukan, berupa pesan apa yang dikirim dan kapan pelaksanaannya. Diagram ini diatur berdasarkan waktu. Objek-objek yang berkaitan dengan proses berjalannya operasi diurutkan dari kiri ke kanan berdasarkan waktu terjadinya dalam pesan yang terurut. Berikut adalah notasi-notasi pada *diagram sequence,* yang ditunjukan pada Tabel 2.4.

Tabel 2.4 Notasi diagram sequence

|  |  |  |
| --- | --- | --- |
| **Notasi** | **Keterangan** | **Simbol** |
| *Object* | Merupakan *instance class* dan dituliskan tersusun secara horizontal. |  |
| *Actor* | *Actor* juga dapat berkomunikasi dengan *object*. |  |
| *Lifeline* | Mengindikasikan keberadaan sebuah objek dalam basis waktu. |  |
| *Activation* | Mengindikasikan sebuah objek yang melakukan sebuah aksi. |  |
| *Message* | Mengindikasikan komunikasi antar objek. |  |

3. 3. **Metode *Waterfall***

*Waterfall* *Model* adalah sebuah metode pengembangan perangkat lunak yang bersifat sekuensial. Metode ini dikenalkan oleh Royce pada tahun 1970 dan pada saat itu disebut sebagai isi klus klasik dan sekarang ini lebih dikenal dengan sekuensial linier. Selain itu model ini merupakan model yang paling banyak dipakai oleh para pengembang perangkat lunak. Inti dari metode *waterfall* adalah pengerjaan dari suatu sistem dilakukan secara berurutan. Jadi jika langkah satu belum dikerjakan maka tidak akan bisa melakukan pengerjaan langkah 2, 3 dan seterusnya.

Keterkaitan dan pengaruh antar tahap ini ada karena hasil luaran sebuah tahap dalam *Waterfall Model* merupakan input bagi tahap berikutnya, dengan demikian ketidak sempurnaan hasil pelaksanaan tahap sebelumnya adalah awal ketidak sempurnaan tahap berikutnya. Memperhatikan karakteristik ini, sangat penting bagi tim pengembang dan perusahaan untuk secara bersama-sama melakukan analisa kebutuhan dan desain sistem sesempurna mungkin sebelum masuk kedalam tahap penulisan kode program. [11] Metode *waterfall* ditunjukkan pada Gambar 2.1.

Gambar 2.1 Metode Waterfall

2. 8. ***Web Service***

*Web service* adalah teknologi yang mengubah kemampuan Internet dengan menambahkan kemampuan Web transaksional, yaitu kemampuan Web untuk saling berkomunikasi dengan pola *program-to-program* (P2P). Fokus Web selama ini didominasi oleh komunikasi *program-to-user* dengan interaksi *business-to-consume*r (B2C), sedangkan Web transaksional akan didominasi oleh *program-to-program* dengan interaksi *business-to-business*.[12]

Sampai dengan saat ini teknologi *Web Service* terus berkembang. Salah satu teknologi yang populer saat ini adalah REST *(Representational State Transfer)* atau terkadang disebut dengan *RESTful*. Beberapa contoh *RESTful Web Service* adalah *Amazon’s Simple Storage Service* (S3), *Atom Publishing Protocol*, dan Google Maps. Pada prinsipnya permintaan data ke suatu *RESTful* *Web Service* sebenarnya adalah suatu *HTTP Request*. [13]

1. 9. **Pengujian Kotak Hitam**

Pada tahap pengujian ini penulis menggunakan pengujian kotak hitam, karena pengujian kotak hitam dapat mengetahui apakah perangkat lunak yang dibuat dapat berfungsi dengan benar dan telah sesuai dengan rancangan yang diharapkan. Pengujian menggunakan sekumpulan aktifitas validasi dengan pengujian kotak hitam. Pengujian kotak hitam merupakan pengujian perangkat lunak dari segi spesifikasi fungsional tanpa menguji desain dan kode program. Pengujian dimaksudkan untuk mengetahui apakah fungsi-fungsi, masukkan, dan keluaran dari perangkat lunak sesuai dengan spesifikasi yang dibutuhkan. Pengujian kotak hitamdilakukan dengan membuat kasus uji yang bersifat mencoba semua fungsi dengan memakai perangkat lunak kemudian melakukan pengecekan apakah perangkat lunak sudah sesuai dengan spesifikasi yang dibutuhkan.[14]

**BAB III**

**PERANCANGAN SISTEM**

Dalam sebuah pengembangan sistem perangkat lunak yang baik diperlukan perancangan yang matang. Dalam perancangan aplikasi Android *Driver Control* akan digambarkan dengan diagram *use case,* diagram aktivitas, dan diagram sekuensial.

1. **Analisis Kebutuhan**
2. **Analisis Kebutuhan Fungsional**

Analisis kebutuhan merupakan tahapan yang sangat penting karena pada tahap ini dilakukan perencanaan terhadap sistem, baik perilakunya, fungsi-fungsi yang diinginkan dan pengelompokkan terhadap fitur-fitur yang harus ada.

Kebutuhan fungsional juga berisi informasi-informasi yang harus ada pada sistem, di antaranya adalah sebagai berikut :

1. Aplikasi ini dapat digunakan oleh beberapa pengguna untuk memantau satu motor yang sudah terpasang alat *Driver Control*.
2. Pengguna mendapatkan notifikasi *email* setelah registrasi akun baru untuk verifikasi.
3. Pengguna dapat mengakses menu **Maps**, **Speed**, **Lock**, dan **Data record** setelah memasukkan kode *user*.
4. Aplikasi dapat menampilkan lokasi motor.
5. Aplikasi mampu menampilkan grafik kecepatan rata-rata berkendara dan rekaman data pengemudi selama 1 minggu terakhir.
6. Aplikasi dapat memberi notifikasi motor jatuh.
7. Pengguna mampu mematikan dan menghidupkan mesin motor melalui aplikasi.
8. Pengguna mampu mengubah data diri dan kata sandi melalui aplikasi.
9. Admin mampu mengaktifkan atau menonaktifkan alat *Driver Control*.
10. **Analisis Kebutuhan Non Fungsional**

Kebutuhan non fungsional merupakan kebutuhan yang mengacu pada kinerja sistem sehingga sistem dapat berjalan secara maksimal. Kebutuhan non fungsional aplikasi *Driver Control* adalah sebagai berikut :

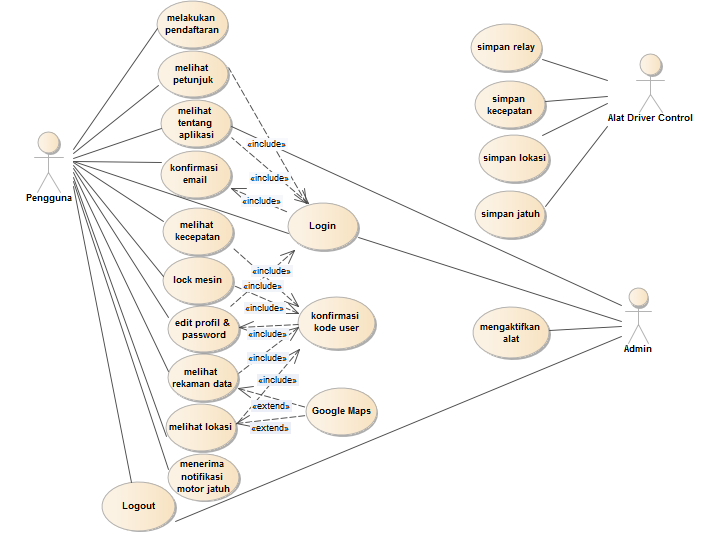
1. **Spesifikasi Perangkat Keras**

Perangkat keras yang digunakan untuk membuat aplikasi Android *Driver Control* untuk memonitor anak berkendara secara waktu-nyata adalah :

1. Laptop Asus A455L, Core i5-5200U, 2.7GHz, RAM 8GB, dengan sistem operasi Windows 8.1 64-bit. Perangkat ini digunakan dalam proses pembuatan perangkat lunak.
2. Lenovo Tablet 2A7-10F untuk pengujian pada perangkat Android oleh pengguna 1.
3. Samsung Galaxy Tablet GT-N5100 untuk pengujian pada perangkat Android oleh pengguna 2.
4. Redmi 2s untuk pengujian pada perangkat Android oleh admin.
5. **Spesifikasi Perangkat Lunak**

Perangkat lunak yang digunakan untuk membuat aplikasi Android *Driver Control* untuk memonitor anak berkendara secara waktu-nyataadalah:

1. Android Studio, digunakan untuk membuat aplikasi Android.
2. *Java Development Kit* (JDK), digunakan agar komputer dapat membaca bahasa pemrograman Java.
3. Enterprise Architect, digunakan untuk perancangan sistem.
4. **Perancangan Sistem Aplikasi**
5. **Diagram Use Case**

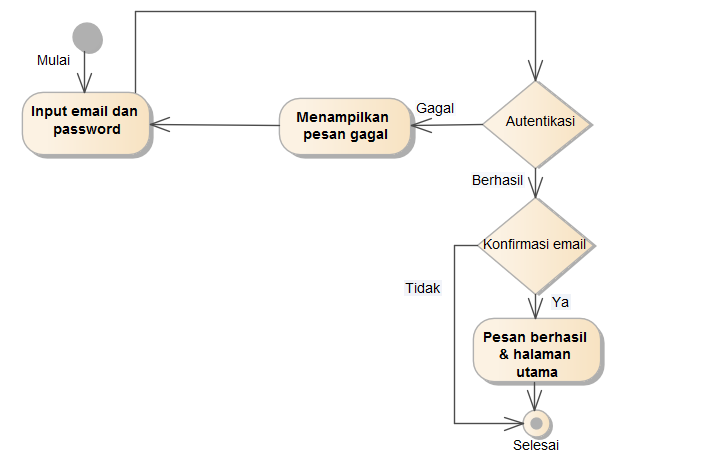
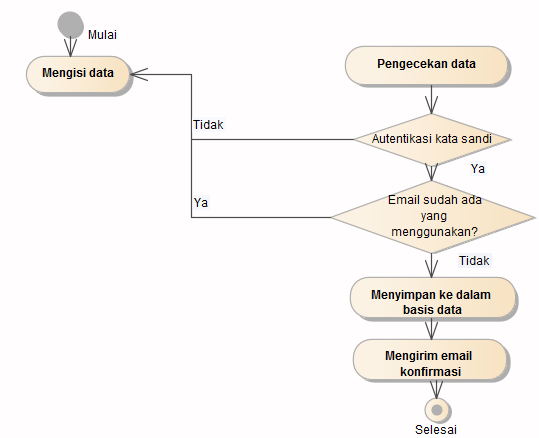
Dari perencanaan kebutuhan sebelumnya dapat digambarkan secara umum sistem yang akan dibuat dengan menggunakan UML, UML yang digunakan adalah UML versi 1.5 dimana terdapat delapan diagram yaitu *Class Diagram, Component Diagram, Deployment Diagram, Use Case Diagram, Activity Diagram, Statechart Diagram, Sequence Diagram,* dan *Collaboration Diagram*. Dalam menggambarkan kebutuhan sistem digunakan *Use Case* *Diagram*, untuk kebutuhan dari aplikasi *Driver* *****Control* dapat dilihat pada Gambar 3.1.

Gambar 3.1 *Use case* diagram sistem

Pada Gambar 3.1 dapat dilihat terdapat 3 aktor pada sistem yaitu **Pengguna***,* **Admin**, dan **Alat Driver Control**, serta terdapat 19 *usecase* yaitu **melakukan pendaftaran**, **login***,* **melihat lokasi***,* **melihat kecepatan***,* **menerima notifikasi motor jatuh**, **konfirmasi email**, **konfirmasi kode user**, **melihat** **petunjuk**, **edit profil dan password**, **lock mesin**, **melihat** **rekaman data, Google Maps, melihat tentang aplikasi, simpan relay, simpan kecepatan, simpan lokasi, simpan jatuh, mengaktifkan alat,** dan **Logout**. Dari *use case* pada Gambar 3.1 dapat disimpulkan bahwa **Pengguna**dapat membuka aplikasi jika sudah **melakukan pendaftaran** dan **konfirmasi email** sebelum **Login**. Setelah berhasil **Login**, pengguna dapat **melihat petunjuk** **penggunaan aplikasi**, **melihat tentang pengembang aplikasi**, dan menu **Profile**. Untuk **melihat lokasi kendaraan***,* **melihat kecepatan kendaraan**, **lock mesin**, dan **melihat rekaman data** dapat dilakukan jika pengguna sudah **konfirmasi kode user** pada menu **Profile**. Saat motor jatuh, pengguna dapat menerima **notifikasi motor jatuh** selama Internet aktif. **Google Maps** digunakan untuk **melihat lokasi kendaraan** jika pengguna mengakses menu **Maps** dan halaman **Rekaman data**. Untuk keluar dari aplikasi pengguna dapat menggunakan menu **Logout**pada aplikasi. **Alat Driver Control** berfungsi untuk menyimpan data **lokasi**, **kecepatan**, **jatuh**, dan **relay** pada basisdata. Sebelum data ditampilkan pada aplikasi, **Admin** harus **Login** untuk mengaktifkan alat terlebih dahulu dengan mengecek **kode alat** dan **status alat Driver Control**.

1. **Diagram Aktivitas**

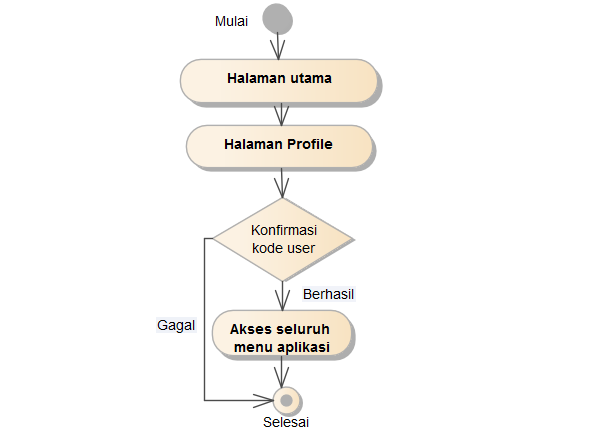
Setelah perencanaan kebutuhan dilakukan, selanjutnya mendesain mengenai perilaku sistem dan aktivitas yang terjadi ketika aplikasi dijalankan. Aktivitas serta perilaku aplikasi ini digambarkan dalam beberapa diagram aktivitas.

Diagram aktivitas **pendaftaran** yang terdapat pada Gambar 3.2 menggambarkan apabila seorang pengguna membuat akun baru. Hal pertama yang harus dilakukan oleh pengguna yaitu mengisi data. Setelah semua data di isi, maka sistem melakukan pengecekan data terlebih dahulu sebelum disimpan ke dalam basisdata. Kemudian autentikasi kata sandi apakah sudah benar atau belum benar, jika belum benar maka akan muncul **pesan kesalahan**, jika kata sandi sudah benar maka selanjutnya sistem akan melakukan autentikasi *email* apakah sudah ada yang menggunakan atau belum. Jika *email* sudah ada yang menggunakan maka akan muncul pesan kesalahan, jika belum ada yang menggunakan maka data tersebut akan disimpan ke dalam basisdata dan aplikasi akan mengirimkan *email* untuk dikonfirmasi oleh pengguna agar dapat **Login**.

Gambar 3.3 Diagram aktivitas login

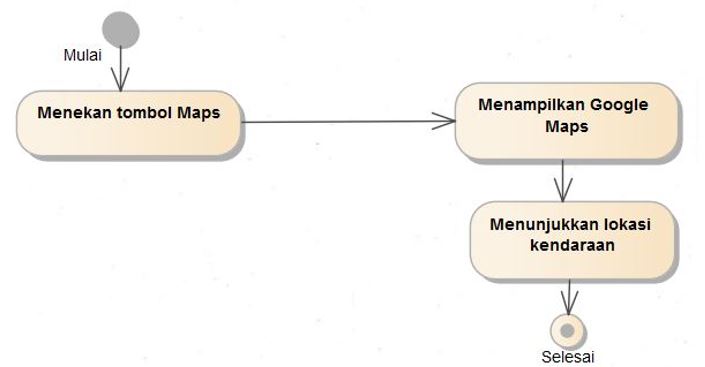
Gambar 3.2 Diagram aktivitas pendaftaran

Diagram aktivitas **Login**yang terdapat pada Gambar 3.3 menggambarkan tentang bagaimana proses yang akan terjadi ketika pengguna masuk ke dalam aplikasi *Driver Control*. Proses awal yang dilakukan adalah memasukkan *email* dan *password,* dan menekan tombol **Login**yang bertujuan melakukan identifikasi terhadap pengguna aplikasi, sehingga aplikasi atau sistem dapat mengenali orang yang menggunakannya. Dalam proses identifikasi juga dibutuhkan autentikasi, proses autentikasi pada sistem merupakan rangkaian proses ketika seorang pengguna memberikan identitas berupa *email* kemudian sistem melakukan pengecekan apakah sudah ada yang menggunakan atau belum, kemudian pengecekan pada kata sandi yang hanya diketahui oleh pemilik akun. Setelah itu sistem mengecek apakah pengguna sudah **konfirmasi email** atau belum, jika sudah **konfirmasi email** setelah melakukan pendaftaran pengguna akan menuju halaman **utama** aplikasi saat menekan tombol **Login**.

****

Gambar 3.4 Diagram aktivitas konfirmasi Kode User

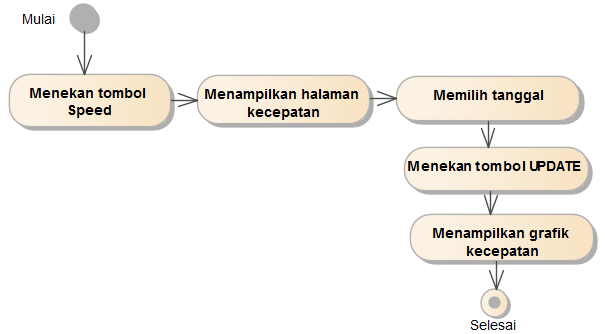
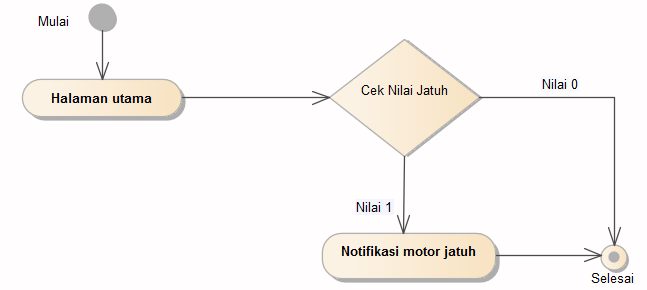
Diagram aktivitas **konfirmasi Kode User**yang terdapat pada Gambar 3.4 menggambarkan tentang bagaimana proses yang akan terjadi ketika pengguna sudah masuk ke halaman **utama** aplikasi dan ingin mengakses menu aplikasi. Untuk dapat mengakses seluruh menu aplikasi, pengguna harus mengisi **Kode User** yang terdapat pada menu **Profile** dan **konfirmasi Kode User**harus berhasil sehingga pengguna bisa membuka semua menu aplikasi. Jika pengguna salah memasukkan **Kode User,** maka pengguna tidak dapat mengakses seluruh menu pada aplikasi.

****

Gambar 3.5 Diagram aktivitas menampilkan peta lokasi kendaraan

Gambar 3.5 menunjukkan bagaimana proses yang terjadi ketika pengguna menekan tombol **Maps** pada aplikasi, aplikasi akan meminta data ke *server* untuk selanjutnya menampilkan lokasi kendaraan motor pada Google Maps.

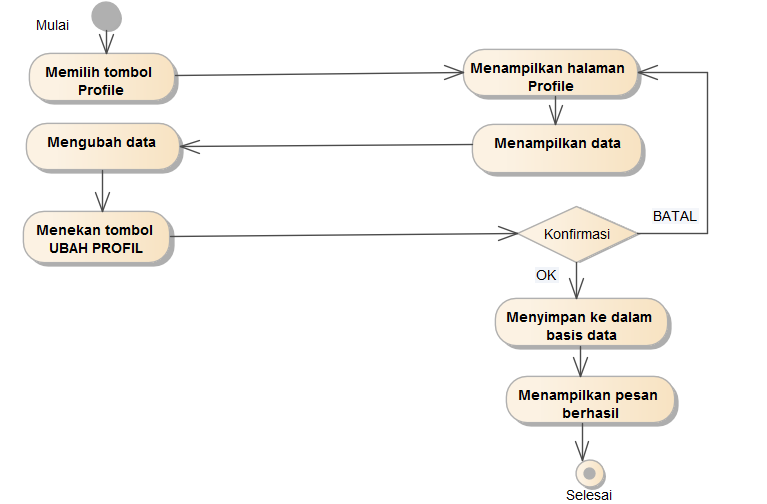
Gambar 3.6 menunjukkan proses yang terjadi ketika pengguna menekan tombol **Speed**pada aplikasi, kemudian aplikasi akan menampilkan halaman **Kecepatan.** Untuk melihat grafik kecepatan, pengguna harus memilih **tanggal** dan menekan tombol **UPDATE** terlebih dahulu.



Gambar 3.7 Diagram aktivitas menampilkan notifikasi motor jatuh

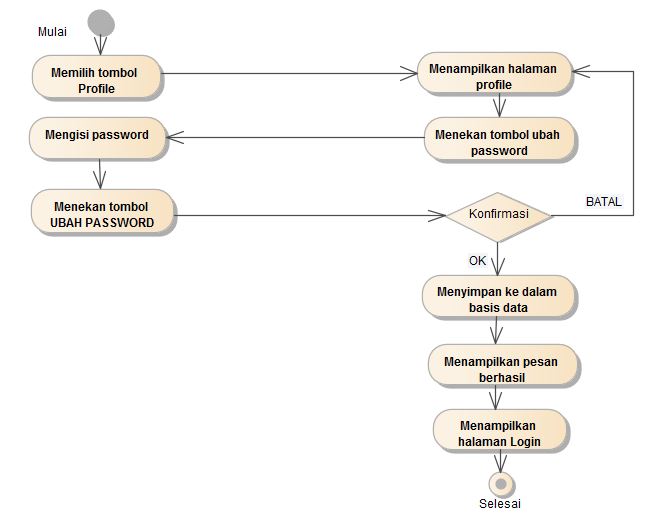
Gambar 3.6 Diagram aktivitas menampilkan grafik kecepatan

Gambar 3.7 merupakan proses yang terjadi ketika motor jatuh, sistem akan mendeteksi status jatuh kendaraan pada basisdata. Jika nilai jatuh satu, maka sistem akan mengirimkan **notifikasi motor jatuh** kepada pengguna saat pengguna mengaktifkan Internet.



Gambar 3.8 Diagram aktivitas ubah profil

Gambar 3.8 merupakan proses yang terjadi ketika pengguna ingin mengubah data profil yang sudah dibuat sebelumnya saat **pendaftaran**, hal pertama yang dilakukan yaitu menekan tombol **Profile**, kemudian aplikasi akan menuju halaman **Profile**, aplikasi menampilkan data pengguna, kemudian pengguna dapat mengubah data dengan menekan tombol **UBAH PROFIL** dan pengguna melakukan **konfirmasi** untuk mengubah data profil. Jika pengguna memilih **OK** sistem akan menyimpan data ke dalam basisdata dan menampilkan pesan berhasil. Jika pengguna memilih **BATAL** maka tidak terjadi perubahan data dan aplikasi tetap pada halaman **Profile**.

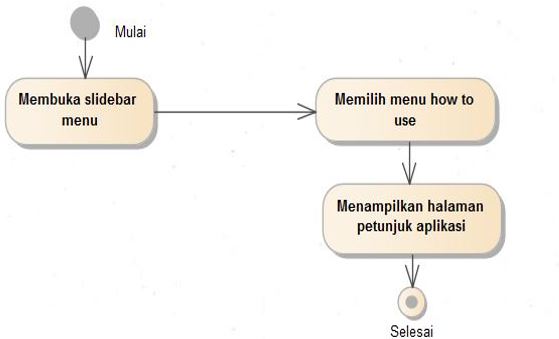
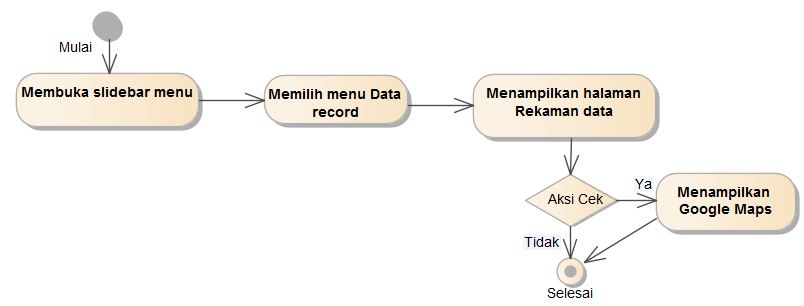
Gambar 3.9 merupakan proses yang terjadi ketika pengguna ingin mengubah *password*, hal pertama yang dilakukan yaitu menekan tombol **Profile** dan aplikasi akan menuju halaman **Profile**, kemudian menekan tombol **UBAH PASSWORD** aplikasi akan menuju halaman **Ubah Password**, pengguna mengisi data **Password Lama**, **Password Baru**, dan **Ulangi Password Baru**. Kemudian pengguna menekan tombol **UBAH PASSWORD** maka akan ada **konfirmasi** untuk mengubah *password*. Jika pengguna memilih **OK** sistem akan menyimpan *password* ke dalam basisdata dan menampilkan pesan berhasil dan pengguna harus **Login** kembali. Jika pengguna memilih **BATAL** maka tidak terjadi perubahan *password* dan aplikasi tetap pada halaman **Ubah Password**.

Gambar 3.9 Diagram aktivitas ubah password

Gambar 3.10 merupakan proses yang terjadi ketika pengguna aplikasi menekan tombol **Lock***,* fungsi tombol **Lock** pada aplikasi *Driver Control* adalah untuk mematikan dan menghidupkan mesin motor secara otomatis. Saat menekan tombol **Lock**sistem akan mengecek status mesin motor hidup atau mati. Jika motor hidup akan muncul halaman yang menampilkan status “Mesin Motor Hidup” dan **konfirmasi** untuk mematikan mesin motor. Jika pengguna memilih **YA**, maka aplikasi akan mematikan mesin motor. Jika pengguna memilih **TIDAK**, mesin motor tetap hidup. Sebaliknya jika motor mati, aplikasi akan menampilkan status “Mesin Motor Mati” dan **konfirmasi** untuk menghidupkan mesin motor.

Gambar 3.10 Diagram aktivitas fitur lock

Gambar 3.11 merupakan proses yang terjadi ketika pengguna aplikasi ingin menuju halaman **Petunjuk**, pertama pengguna membuka **slidebar menu***,* kemudian menekan tombol **How to use***,* dan pengguna akan diarahkan menuju halaman **Petunjuk** agar mengetahui cara penggunaan aplikasi *Driver Control.*



Gambar 3.12 Diagram aktivitas menampilkan data record

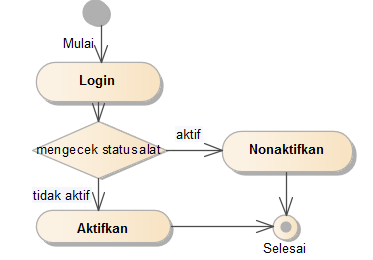
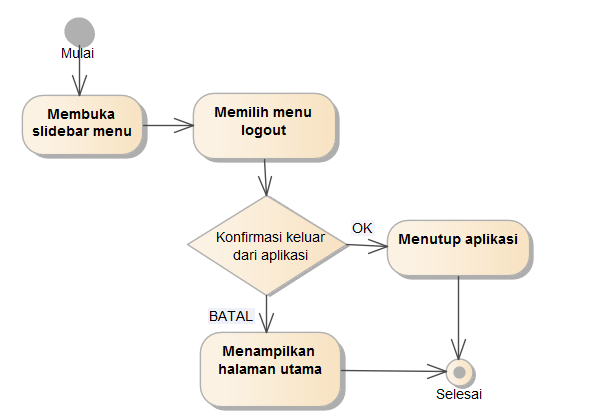
Gambar 3.11 Diagram aktivitas menu how to use

Gambar 3.12 merupakan proses yang terjadi ketika pengguna aplikasi ingin melihat rekaman data pengemudi, pertama pengguna membuka **slidebar menu***,* kemudian menekan tombol **Data record***,* kemudian pengguna akan diarahkan menuju halaman **Rekaman data***,* jika ingin melihat lokasi pengguna dapat menekan tautan **Cek** agar bisa mengakses peta lokasi kendaraan motor.

Gambar 3.13 Diagram aktivitas menu about

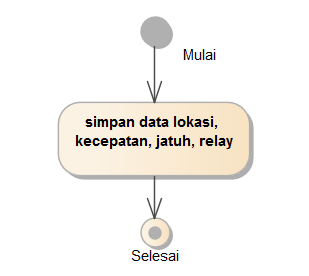
Gambar 3.13 merupakan proses yang terjadi ketika pengguna aplikasi ingin melihat tentang pengembang aplikasi, pertama pengguna membuka **slidebar menu**kemudian menekan tombol **About***,* dan pengguna akan diarahkan menuju halaman **Tentang aplikasi** *Driver Control.*

Gambar 3.14 merupakan proses yang terjadi ketika pengguna ingin keluar dari aplikasi, pertama pengguna membuka **slidebar menu**kemudian menekan tombol **Logout***,* kemudian akan muncul **konfirmasi** keluar dari aplikasi. Jika pengguna memilih **OK** maka pengguna akan keluar dari menu aplikasi, jika pengguna memilih **BATAL**, pengguna akan diarahkan menuju halaman **utama**.

****

Gambar 3.15 Diagram aktivitas admin

Gambar 3.14 Diagram aktivitas logout

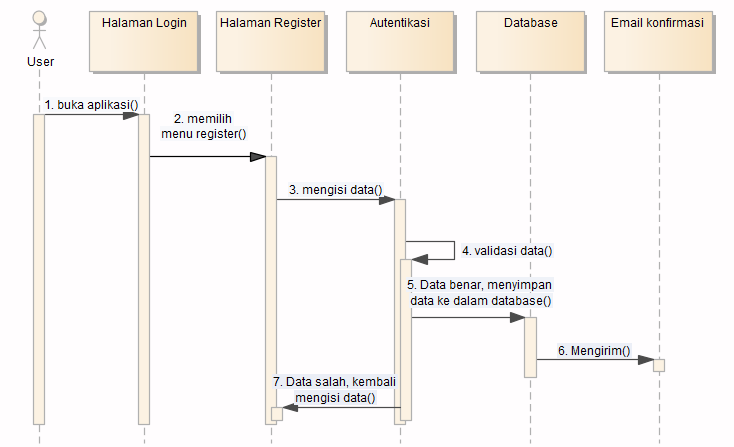
Gambar 3.15 merupakan proses yang terjadi ketika admin menyaring data dari alat untuk ditampilkan ke aplikasi pengguna, pertama admin **Login**kemudian mengecek status alat aktif atau tidak. Jika kondisinya tidak aktif, admin dapat mengaktifkan alat. Sebaliknya jika kondisi alat aktif, admin dapat menonaktifkan alat untuk mengganti alat yang rusak atau karena hal lainnya.

Gambar 3.16 Diagram aktivitas alat

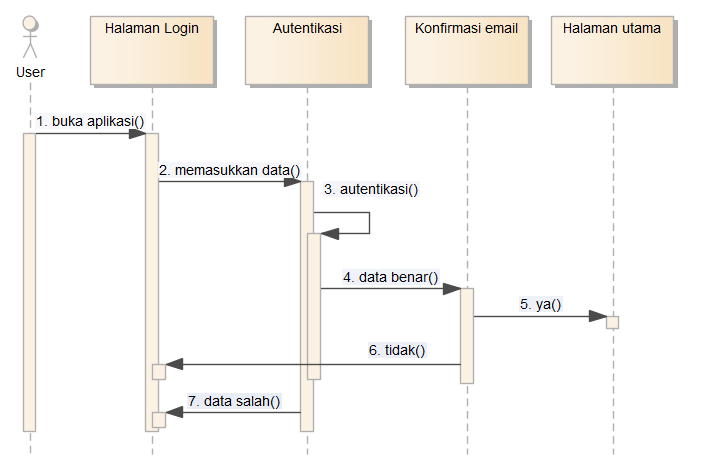
Gambar 3.16 merupakan proses yang terjadi pada bagian **alat Driver Control**, alat berfungsi untuk menyimpan data lokasi, kecepatan, jatuh, dan relay pada basisdata.

1. **Diagram Sekuensial**

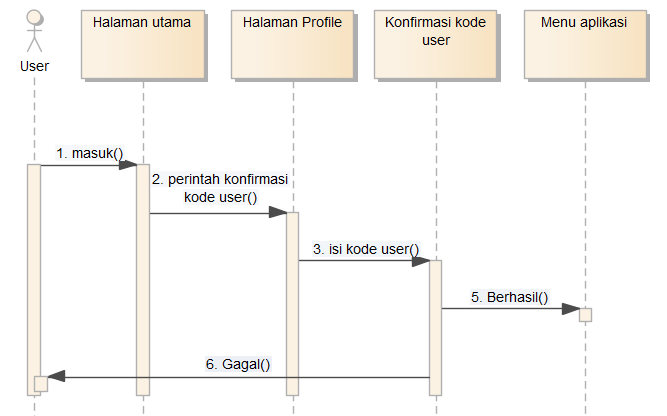
Tahap berikutnya untuk memberikan gambaran yang lebih jauh mengenai proses yang dilakukan oleh aplikasi akan digambarkan proses-proses yang terjadi dengan diagram sekuensial.

Pada Gambar 3.17 menunjukkan proses yang terjadi pada saat pengguna membuat akun baru. Proses pertama yang terjadi adalah pengguna membuka aplikasi dan memilih tombol **Register**, kemudian sistem akan mengarahkan pengguna menuju halaman **Register**, selanjutnya pengguna diminta untuk mengisi data, setelah semua data terisi sistem akan melakukan validasi, apakah *email* yang digunakan sudah ada atau belum, jika sudah maka akan muncul pesan kesalahan. Jika tidak maka sistem akan melakukan pengecekan kata sandi jika belum benar akan muncul pesan kesalahan*,* jika sudah benar data akan disimpan ke dalam basisdata dan aplikasi akan mengirim *email* untuk dikonfirmasi oleh pengguna.

Gambar 3.17 Diagram sekuensial pendaftaran

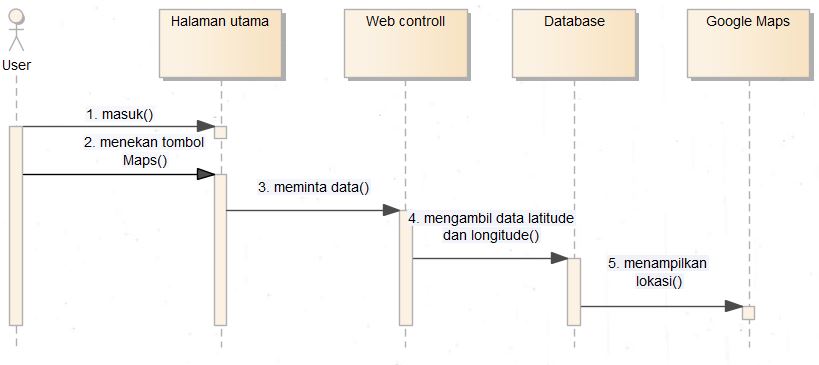
Pada Gambar 3.18 menggambarkan proses yang terjadi pada saat **Login** untuk masuk ke halaman **utama**. Proses pertama yang terjadi adalah pengguna membuka aplikasi dan sistem menampilkan halaman **Login** selanjutnya pengguna memasukan *email* dan *password*, setelah itu sistem akan melakukan autentikasi apakah data yang diisi sudah benar atau masih salah, jika salah maka sistem akan menampilkan pesan kesalahan, jika benar selanjutnya sistem mengecek apakah pengguna sudah **konfirmasi email** atau belum, jika sudah **konfirmasi email** setelah melakukan pendaftaran pengguna akan menuju halaman **utama** aplikasi saat menekan tombol **Login**.

Gambar 3.18 Diagram sekuensial login



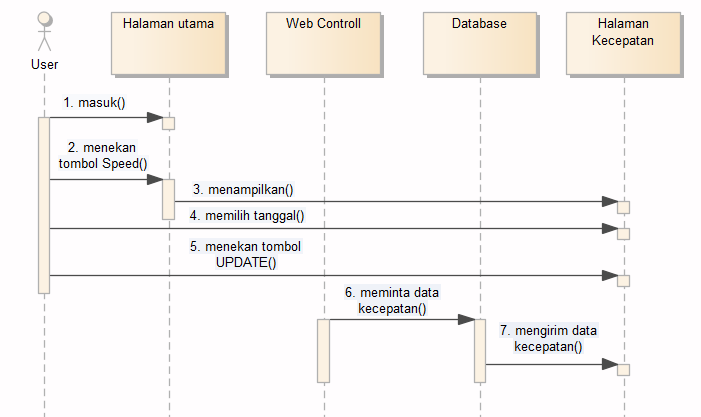
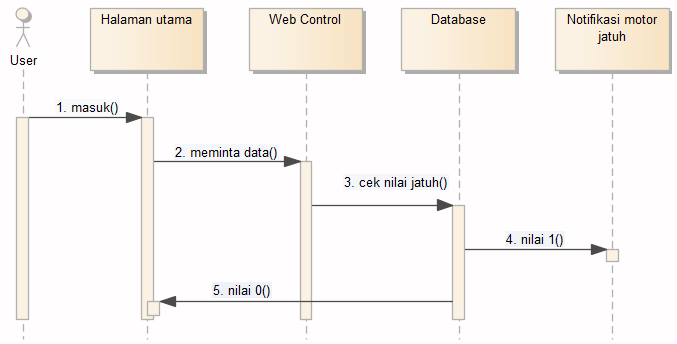
Gambar 3.15 Diagram sekuensial pendaftaran

Gambar 3.19 Diagram sekuensial konfirmasi Kode User

Gambar 3.19 menggambarkan proses yang dilakukan pengguna untuk dapat mengakses seluruh menu aplikasi *Driver Control*, yang dilakukan setelah pengguna berhasil melakukan **Login** dan masuk ke halaman **utama** aplikasi. Jika pengguna mencoba menekan salah satu menu aplikasi pada halaman **utama** maka akan tampil perintah untuk mengisi **Kode User** pada menu **Profile**. Seluruh menu aplikasi dapat di akses pengguna setelah mengisikan **Kode User** dengan benar. Jika pengguna salah memasukkan **Kode User,** maka pengguna tidak dapat mengakses seluruh menu pada aplikasi.

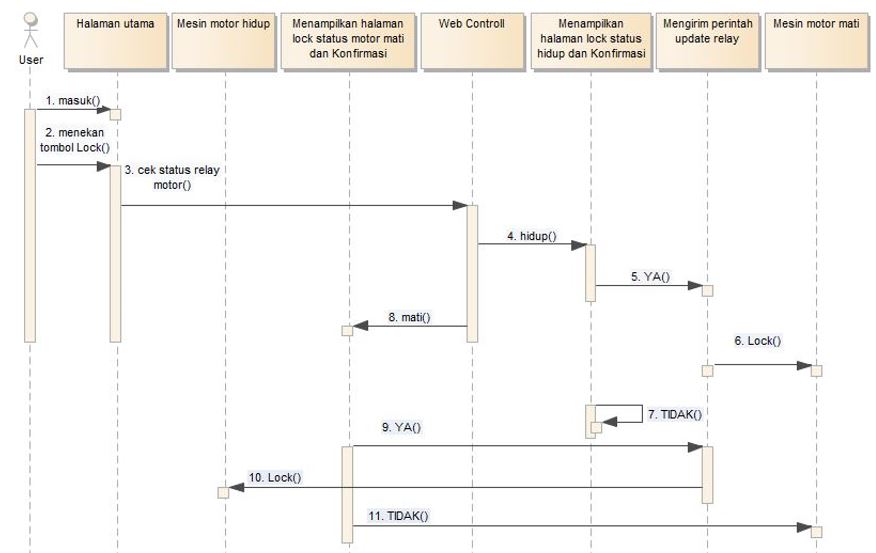
Gambar 3.20 Diagram sekuensial maps

Pada Gambar 3.20 menggambarkan proses yang terjadi pada saat pengguna ingin melihat lokasi kendaraan. Proses yang pertama yaitu pengguna masuk ke halaman **utama** kemudian menekan tombol **Maps**, pengguna akan diarahkan menuju Google Maps agar dapat melihat peta lokasi kendaraan.

Pada Gambar 3.21 menggambarkan proses yang terjadi pada saat pengguna ingin melihat kecepatan rata-rata berkendara. Proses yang pertama yaitu pengguna masuk ke halaman **utama** kemudian menekan tombol **Speed**. Aplikasi akan menuju halaman **Kecepatan**,setelah pengguna memilih **tanggal** dan menekan tombol **UPDATE** sistem akan meminta data kecepatan ke basisdata untuk ditampilkan dalam ****bentuk grafik pada halaman **Kecepatan**.

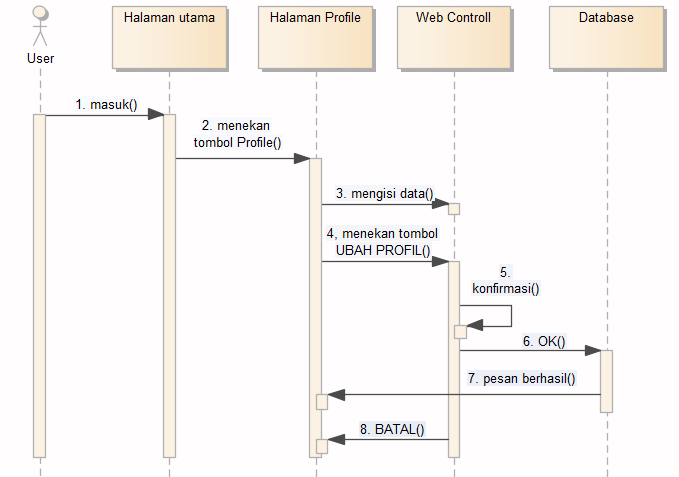
Gambar 3.22 Diagram sekuensial notifikasi motor jatuh

Gambar 3.21 Diagram sekuensial kecepatan

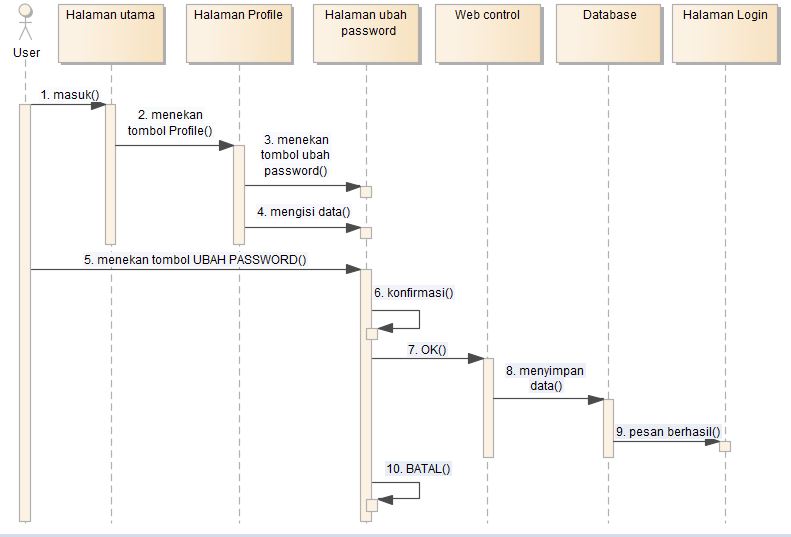
****Gambar 3.22 menunjukkan proses yang terjadi pada saat motor jatuh, sistem akan mendeteksi nilai jatuh kendaraan pada basisdata. Jika nilai jatuh satu, maka sistem akan mengirimkan notifikasi motor jatuh kepada pengguna saat pengguna mengaktifkan Internet.

Gambar 3.23 Diagram sekuensial lock

Pada Gambar 3.23 menggambarkan proses yang terjadi saat pengguna ingin mematikan atau menghidupkan mesin motor melalui aplikasi. Pertama masuk ke dalam aplikasi, kemudian menekan tombol **Lock**,kemudian *web control* akan mengecek status mesin motor hidup atau mati. Jika motor hidup akan muncul halaman yang menampilkan status “Mesin Motor Hidup” dan **konfirmasi** untuk mematikan mesin motor. Jika pengguna memilih **YA**, maka aplikasi akan mematikan mesin motor. Jika memilih **TIDAK**, mesin motor tetap hidup. Sebaliknya jika motor mati, aplikasi akan menampilkan status “Mesin Motor Mati” dan **konfirmasi** untuk menghidupkan mesin motor.

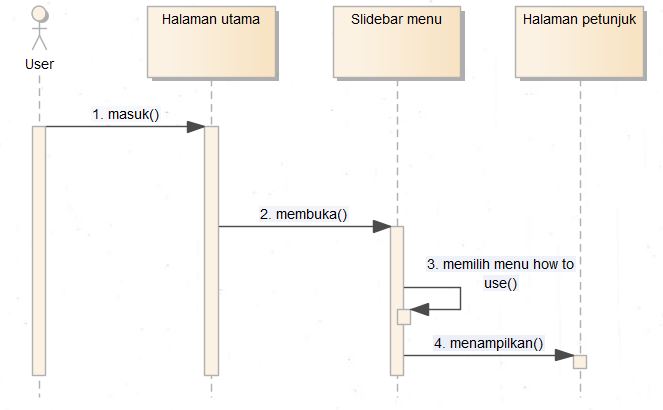
****Pada Gambar 3.24 menggambarkan proses yang terjadi pada saat pengguna ingin mengedit data profil. Setelah masuk halaman utama pengguna menekan tombol **Profile**, kemudian aplikasi akan menuju halaman **Profile**, menampilkan data pengguna aplikasi, pengguna dapat mengubah data dengan menekan tombol **UBAH PROFIL** maka akan ada **konfirmasi** untuk mengubah data profil. Jika pengguna memilih **OK** sistem akan menyimpan data ke dalam basisdata dan menampilkan pesan berhasil. Jika pengguna memilih **BATAL** maka tidak terjadi perubahan data dan aplikasi tetap pada halaman **Profile**.

Gambar 3.24 Diagram sekuensial ubah profil

Pada Gambar 3.25 menunjukkan proses yang terjadi pada saat pengguna ingin mengubah *password*, hal pertama yang dilakukan yaitu menekan tombol **Profile** dan aplikasi akan menuju halaman **Profile**, kemudian menekan tombol **UBAH PASSWORD** aplikasi akan menuju halaman **Ubah Password**, pengguna akan mengisi data **Password Lama**, **Password Baru**, dan **Ulangi Password Baru**. Kemudian pengguna menekan tombol **UBAH PASSWORD** maka akan ada **konfirmasi** untuk mengubah *password*. Jika pengguna memilih **OK** sistem akan menyimpan *password* ke dalam basisdata dan menampilkan **pesan berhasil** dan pengguna harus **Login** kembali. Jika pengguna memilih **BATAL** maka tidak terjadi perubahan *password* dan aplikasi tetap pada halaman **Ubah Password**.

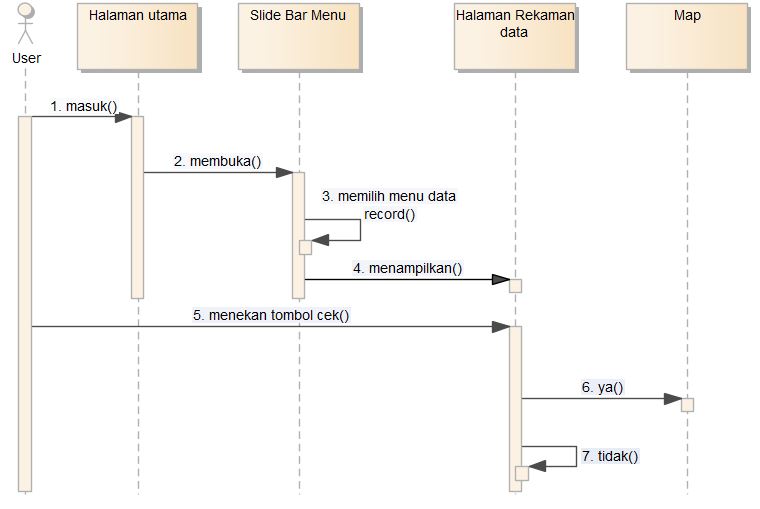
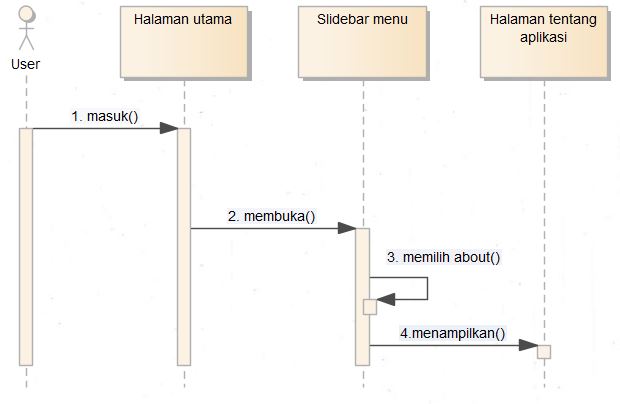
Gambar 3.25 Diagram sekuensial ubah password

Pada Gambar 3.26 menunjukkan proses yang terjadi pada saat pengguna ingin melihat petunjuk penggunaan aplikasi. Proses yang pertama yaitu penggunamasuk ke halaman **utama** kemudian membuka **slidebar menu***,* menekan tombol **How to use**, kemudian pengguna akan dialihkan menuju halaman **Petunjuk** aplikasi.

****

Gambar 3.26 Diagram sekuensial petunjuk aplikasi

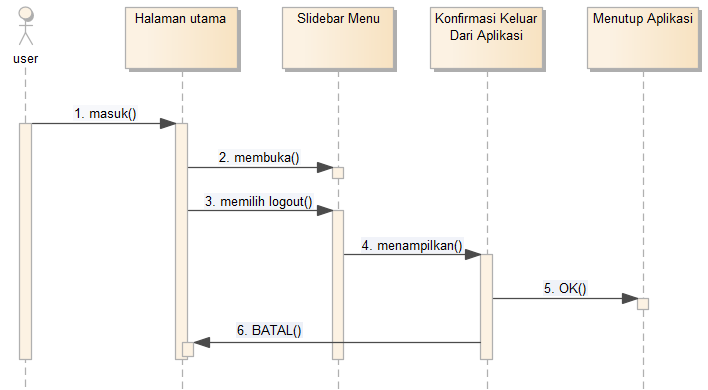
Gambar 3.27 menunjukkan proses yang terjadi pada saat pengguna ingin melihat rekaman data pengemudi. Proses yang pertama yaitu pengguna masuk ke halaman **utama** kemudian membuka **slidebar menu**, menekan tombol **Data record**, kemudian pengguna akan dialihkan menuju halaman **Rekaman data**. Pada tabel rekaman data terdapat tautan **Cek** jika ditekan akan menuju Google Maps. Jika tidak ditekan akan tetap berada pada halaman **Rekaman data**.

****

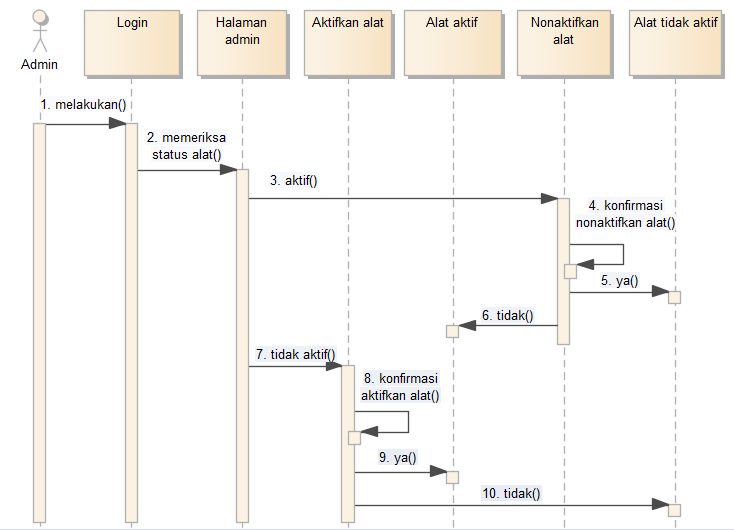
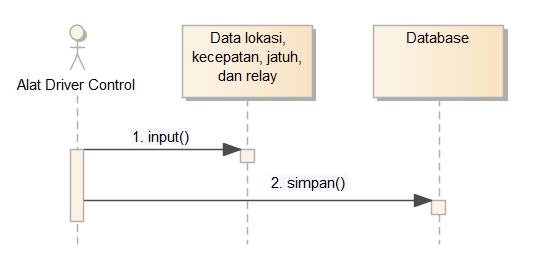
Gambar 3.28 Diagram sekuensial tentang aplikasi

Gambar 3.27 Diagram sekuensial rekaman data

Pada Gambar 3.28 menunjukkan proses yang terjadi pada saat pengguna ingin melihat tentang pengembang aplikasi. Proses yang pertama yaitu pengguna masuk ke halaman **utama** kemudian membuka **slidebar menu** dan menekan tombol **About**, setelah itu sistem akan mengalihkan pengguna menuju halaman **Tentang aplikasi**.

Gambar 3.29 menunjukkan proses yang terjadi pada saat pengguna ingin keluar dari aplikasi. Proses yang pertama yaitu pengguna masuk ke halaman **utama** kemudian membuka **slidebar menu**, dan memilih menu **Logout**, akan muncul **konfirmasi**, jika pengguna memilih **OK** maka pengguna akan keluar dari menu aplikasi. Jika memilih **BATAL** maka akan diarahkan ke halaman **utama**.

Gambar 3.29 Diagram sekuensial logout

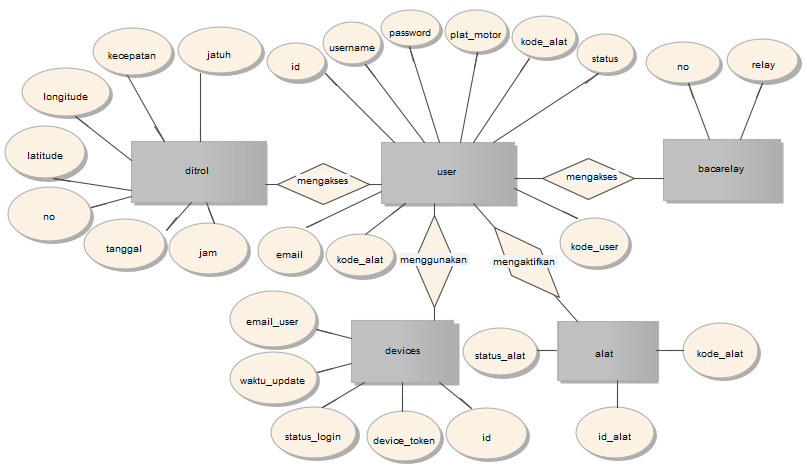
****

Gambar 3.31 Diagram sekuensial alat

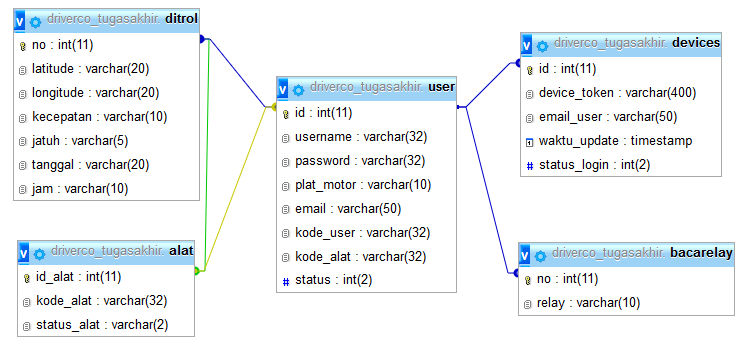
Gambar 3.30 Diagram sekuensial admin

Pada Gambar 3.30 menunjukkan proses yang terjadi pada saat ketika admin menyaring data dari alat untuk ditampilkan ke aplikasi pengguna, pertama admin **Login**kemudian mengecek status alat aktif atau tidak. Jika kondisinya tidak aktif admin dapat memilih **aktifkan** kemudian menyetujui **konfirmasi** maka alat akan aktif. Sebaliknya jika alat ingin dinonaktifkan, admin juga harus melakukan konfirmasi terlebih dahulu. Gambar 3.31 merupakan proses yang terjadi pada **alat Driver Control**, pertama alat mengirim data berupa lokasi, kecepatan, jatuh, dan relay ke dalam basisdata kemudian disimpan pada basisdata.

1. **Perancangan Basisdata**

Pada perancangan perangkat lunak, desain basisdata merupakan suatu tahap yang penting, pada tahap ini membuat desain dari skema penyimpanan data perangkat lunak. Pembuatan desain basisdata dilakukan dengan membuat gambaran mengenai kebutuhan serta keterkaitan antar data. Gambaran yang digunakan yaitu *Entity Relationship Diagram* (ERD). Untuk menggambarkan basisdata pada aplikasi *Driver Control* dapat dilihat pada Gambar 3.32 dan Gambar 3.33.

Gambar 3.32 Desain ERD aplikasi



Gambar 3.33 Desain basisdata relasi antar tabel

Gambar 3.33 merupakan desain ERD aplikasi Android *Driver Control,* dalam ERD terdapat empat entitas yaitu entitas user, ditrol, bacarelay, dan devices. Pada entitas user terdapat delapan atribut yaitu atribut id, username, password, plat\_motor, email, status, kode\_alat, dan kode\_user. Pada entitas ditrol terdapat tujuh atribut yang terdiri dari atribut no, latitude, longitude, kecepatan, jatuh, tanggal, dan jam. Pada entitas bacarelay terdapat dua atribut yaitu no dan relay. Pada entitas devices terdapat lima atribut yaitu id, email\_user, waktu\_update, device\_token, dan status\_login. Pada entitas alat terdapat tiga atribut yang terdiri dari atribut id\_alat, kode\_alat, dan status\_alat. Gambar 3.33 menunjukkan keterkaitan antara tabel user, tabel ditrol, tabel bacarelay, tabel devices, dan tabel alat. Keterkaitan antara tabel user, tabel ditrol, tabel bacarelay, tabel devices, dan tabel alat digambarkan dengan menggunakan *Foreign Key* id yang terdapat pada tabel user.

Dari desain yang telah dibuat dihasilkan kamus data yang akan diterapkan pada basisdata. Seperti pada Tabel 3.1, Tabel 3.2, Tabel 3.3, Tabel 3.4, dan Tabel 3.5.

Tabel 3.1 Kamus data user

|  |  |
| --- | --- |
| Kolom | Tipe Data |
| id | int(11) |
| username | varchar(32) |
| password | varchar(32) |
| plat\_motor | varchar(10) |
| email | varchar(50) |
| kode\_user | varchar(32) |
| kode\_alat | varchar(32) |
| status | int(2) |

Seperti yang terlihat pada Tabel 3.1, dimana pada tabel user terdapat 6 atributyaitu id, username, password, plat\_motor, email, dan no\_hp. Pada basisdata id sebagai *primary key* memiliki tipe data integer dengan panjang 11, username memiliki tipe data varchar dengan panjang 32, password memiliki tipe data varchar dengan panjang 32, plat\_motor memiliki tipe data varchar dengan panjang 10, email memiliki tipe data varchar dengan panjang 50, kode\_user memiliki tipe data varchar dengan panjang 32, kode\_alat memiliki tipe data varchar dengan panjang 32, dan status memiliki tipe data integer dengan panjang 2.

Tabel 3.2 Kamus data ditrol

|  |  |
| --- | --- |
| Kolom | Tipe Data |
| no | int(11) |
| latitude | varchar(20) |
| longitude | varchar(20) |
| kecepatan | varchar(10) |
| jatuh | varchar(5) |
| tanggal | varchar(20) |
| jam | varchar(10) |

Seperti yang terlihat pada Tabel 3.2, dimana pada tabel ditrol terdapat sembilan atributyaitu no, latitude, longitude, kecepatan, jatuh, tanggal, dan jam. Pada basisdata no sebagai *primary key* memiliki tipe data integer dengan panjang 11, latitude memiliki tipe data varchar dengan panjang 20, longitude memiliki tipe data varchar dengan panjang 20, kecepatan memiliki tipe data varchar dengan panjang 10, jatuh memiliki tipe data varchar dengan panjang 5, tanggal memiliki tipe data varchar dengan panjang 20, dan jam memiliki tipe data varchar dengan panjang 10.

Tabel 3.3 Kamus data bacarelay

|  |  |
| --- | --- |
| Kolom | Tipe Data |
| no | int(11) |
| relay | varchar(10) |

Seperti yang terlihat pada Tabel 3.3, dimana pada tabel ditrol terdapat dua atributyaitu no, dan relay. Pada basisdata no sebagai *primary key* memiliki tipe data integer dengan panjang 11, dan relay memiliki tipe data varchar dengan panjang 10.

Tabel 3.4 Kamus data devices

|  |  |
| --- | --- |
| Kolom | Tipe Data |
| id | int(11) |
| device\_token | varchar(400) |
| email\_user | varchar(50) |
| waktu\_update | timestamp |
| status\_login | int(2) |

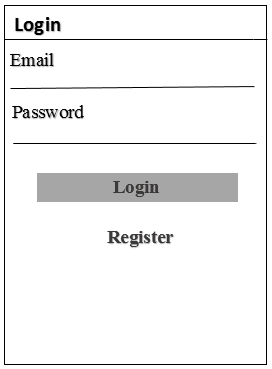
Seperti yang terlihat pada Tabel 3.4, dimana pada tabel devices terdapat empat atributyaitu id, device\_token, email\_user dan waktu\_update. Pada basisdata id sebagai *primary key* memiliki tipe data integer dengan panjang 11, device\_token memiliki tipe data varchar dengan panjang 400, email\_user memiliki tipe data varchar dengan panjang 50, waktu\_update memiliki tipe data timestamp, dan status\_login memiliki tipe data integer dengan panjang 2.

Tabel 3.5 Kamus data alat

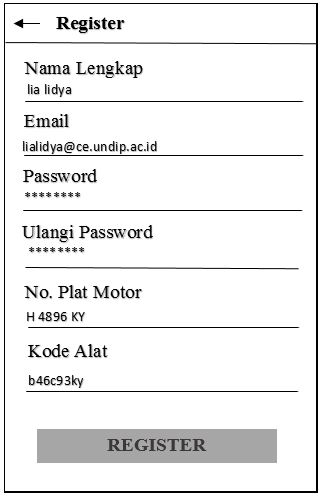
|  |  |
| --- | --- |
| Kolom | Tipe Data |
| id\_alat | int(11) |
| kode\_alat | varchar(32) |
| status\_alat | varchar(2) |

Seperti yang terlihat pada Tabel 3.5, dimana pada tabel alat terdapat tiga atributyaitu id\_alat, kode\_alat, dan status\_alat. Pada basisdata id\_alat sebagai *primary key* memiliki tipe data integer dengan panjang 11, kode\_alat memiliki tipe data varchar dengan panjang 32, dan status\_alat memiliki tipe data varchar dengan panjang 2.

1. **Desain Tampilan**

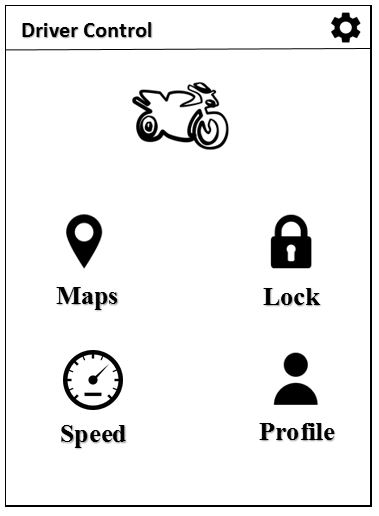
Desain tampilan merupakan tahapan yang dilakukan berikutnya, desain tampilan bertujuan memberikan gambaran secara umum dari aplikasi yang akan dibuat. Desain tampilan merupakan salah satu tahapan yang penting, karena tampilan akan berpengaruh terhadap interaksi antara pengguna dan aplikasi, desain tampilan yang mudah digunakan akan membuat pengguna merasa nyaman ketika menggunakan aplikasi.

Gambar 3.34 Tampilan halaman login

Pada Gambar 3.34 merupakan desain tampilan halaman **Login**. Halaman **Login** adalah halaman **awal** ketika aplikasi pertama kali dijalankan oleh pengguna. Pada halaman **Login** terdapat kolom yang berisi data apa saja yang harus diisi oleh pengguna. Kemudian terdapat tombol **Login** untuk masuk ke dalam aplikasi dan tombol **Register** yang berfungsi untuk melakukan pendaftaran akun yang baru. Jika pengguna menekan tombol **Login** maka akan terjadi proses autentikasi data, ketika autentikasi data berhasil maka pengguna akan dialihkan menuju halaman **utama**.

Gambar 3.35 Tampilan Halaman Register

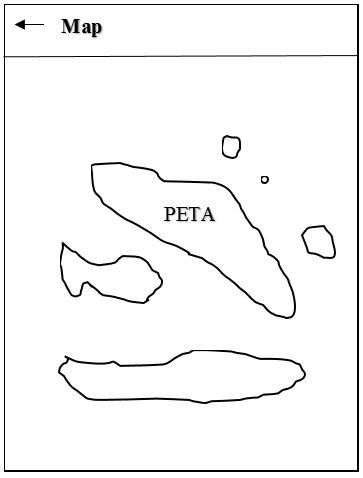
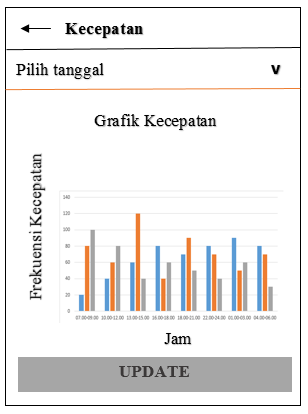
Gambar 3.30 Tampilan halaman login

Gambar 3.35 merupakan tampilan halaman pembuatan akun baru dari aplikasi *Driver Control*. Pada halaman **Register** terdapat teks yang memberikan informasi kepada pengguna mengenai data apa saja yang harus diisi oleh pengguna pada kolom yang masih kosong. Selain itu terdapat dua tombol yaitu **Register** dan . Tombol **Register** berfungsi untuk menyimpan data yang telah diisi oleh pengguna ke dalam basisdata sehingga pengguna dapat masuk ke dalam aplikasi, sedangkan tombol  berfungsi untuk kembali ke halaman **Login**.

Gambar 3.36 Tampilan halaman utama aplikasi

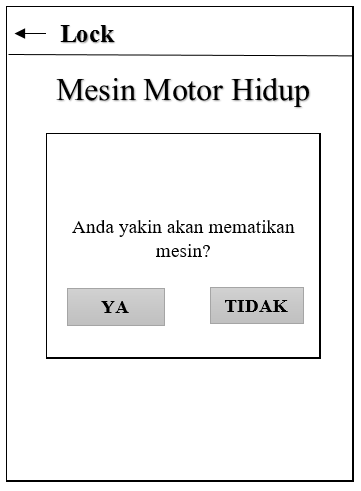
Gambar 3.36 merupakan tampilan halaman **utama** aplikasi *Driver Control*. Pada halaman **utama** terdapat beberapa tombol di antaranya tombol **Maps** untuk melihat peta lokasi kendaraan, tombol **Speed** untuk melihat grafik kecepatan rata-rata berkendara, tombol **Profile** untuk mengubah data profil pengguna, tombol **Lock** berfungsi untuk mematikan atau menghidupkan mesin motor secara otomatis. Pada halaman **utama** terdapat **slidebar menu**yang terdiri dari empat menu yaitu **How to use**, **Data record**, **About**, dan **Logout**.

Gambar 3.37 merupakan tampilan ketika tombol **Maps** ditekan, pengguna dapat melihat posisi kendaraan melalui Google Maps. Pada halaman **Maps** terdapat tombol yang berfungsi untuk kembali ke halaman sebelumnya.



Gambar 3.37 Tampilan halaman maps

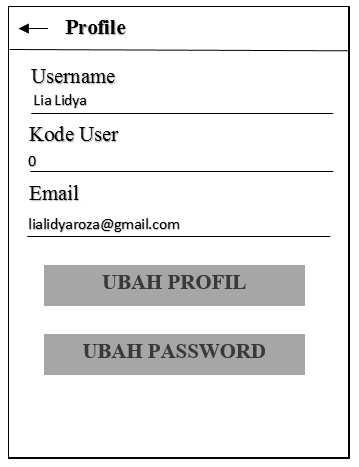
Gambar 3.38 Tampilan halaman kecepatan

Gambar 3.38 merupakan tampilan ketika tombol **Speed** ditekan, pengguna dapat melihat kecepatan pengemudi yang disajikan dalam bentuk grafik setelah memilih **tanggal** dan menekan tombol **UPDATE**. Pada halaman **Kecepatan** terdapat tombol yang berfungsi untuk kembali ke halaman sebelumnya.

Gambar 3.39 Tampilan halaman lock ketika motor hidup

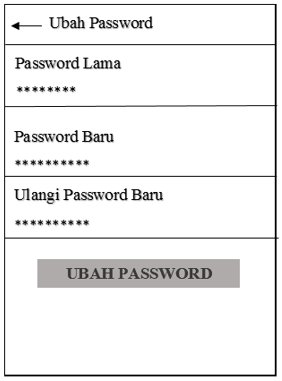
Gambar 3.39 merupakan tampilan ketika tombol **Lock** ditekan dan motor dalam kondisi hidup. Pada halaman **Lock** akan muncul teks “Mesin Motor Hidup” dan **konfirmasi** untuk mematikan mesin motor secara otomatis. Jika pengguna memilih **YA** maka sistem akan mematikan mesin motor. Jika **TIDAK**, pengguna tetap pada halaman **Lock** dan mesin tetap hidup. Pada halaman **Lock** terdapat tombol yang berfungsi untuk kembali ke halaman sebelumnya.

Gambar 3.40 merupakan tampilan ketika tombol **Lock** ditekan dan motor dalam kondisi mati. Pada halaman **Lock** akan menunjukkan status motor mati yang ditampilkan dalam bentuk teks “Mesin Motor Mati” dan **konfirmasi** untuk menghidupkan mesin motor secara otomatis. Jika pengguna memilih **YA** maka sistem akan menghidupkan mesin motor. Jika **TIDAK**, pengguna tetap pada halaman **Lock** dan mesin motor tetap mati. Pada halaman **Lock** terdapat tombol yang berfungsi untuk kembali ke halaman sebelumnya.



Gambar 3.41 Tampilan halaman edit profil

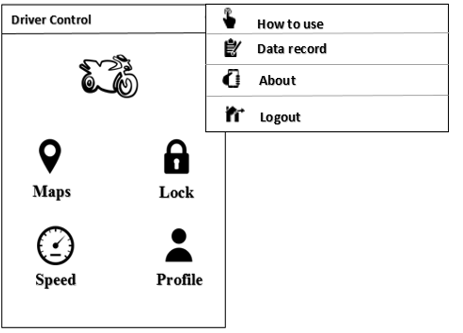
Gambar 3.40 Tampilan halaman lock ketika motor mati

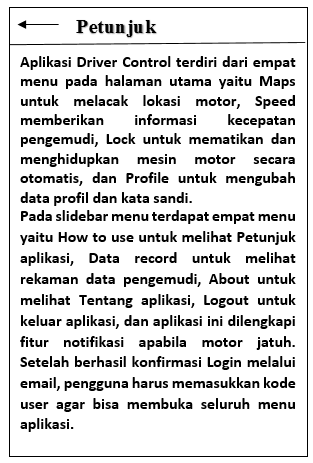
****Gambar 3.41 merupakan tampilan ketika tombol **Profile** ditekan, pengguna dapat mengubah data profil pada aplikasi. Pada halaman ini terdapat tombol **UBAH PROFIL** untuk menyimpan data kemudian tombol **UBAH PASSWORD** yang berfungsi untuk mengganti kata sandi. Pada halaman **Profile** terdapat tombol yang berfungsi untuk kembali ke halaman sebelumnya.

Gambar 3.42 Tampilan halaman ubah password

Gambar 3.42 merupakan tampilan ketika tombol **UBAH PASSWORD** ditekan, pengguna dapat mengubah kata sandi. Pada halaman ini pengguna wajib mengisi **password yang lama**, **password yang baru**, dan **konfirmasi** **password yang baru**. Tombol **UBAH PASSWORD** untuk menyimpan data, dan tombol untuk kembali ke halaman sebelumnya.

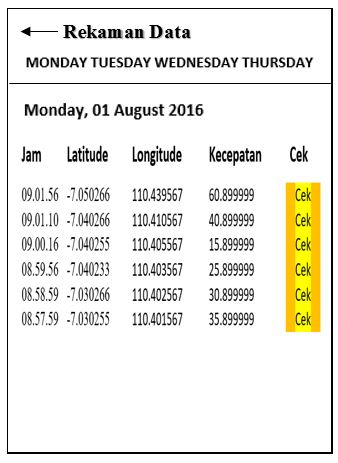
Gambar 3.43 merupakan tampilan **slidebar menu** yang terdiri dari menu **How to use** untuk menuju ke halaman **Petunjuk** aplikasi, menu **Data record** untuk menuju ke halaman **Rekaman data**, menu **About** untuk menuju ke halaman tentang pengembang aplikasi dan menu **Logout** berfungsi untuk keluar dari aplikasi.





Gambar 3.43 Tampilan *slidebar menu*

Gambar 3.44 Tampilan halaman petunjuk aplikasi

Gambar 3.44 merupakan tampilan ketika tombol **How to use** ditekan, pengguna dapat melihat petunjuk cara pemakaian aplikasi di halaman **Petunjuk**. Pada halaman **Petunjuk** terdapat tombol untuk kembali ke halaman sebelumnya.

Gambar 3.45 Tampilan menu Data record

Gambar 3.45 merupakan tampilan menu **Data record** dalam bentuk tabel yang berisi rekaman data pengemudi. Data yang ditampilkan berupa tanggal, jam, *latitude*, *longitude*, dan kecepatan. Pada tabel rekaman data terdapat tautan **Cek** jika ditekan akan menuju Google Maps. Jika tidak ditekan akan tetap berada pada halaman **Rekaman data**.

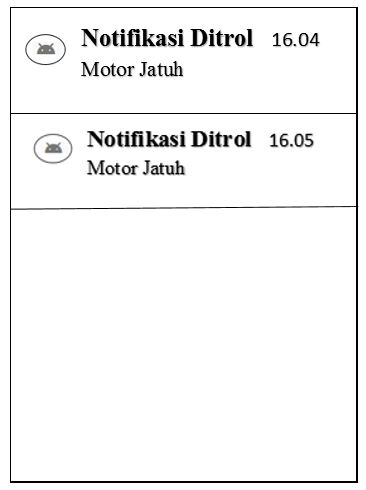
Gambar 3.46 merupakan tampilan ketika tombol **About** ditekan, pengguna dapat melihat seputar aplikasi dan pengembang aplikasi pada halaman ini. Pada halaman **Petunjuk** terdapat tombol yang berfungsi untuk kembali ke halaman sebelumnya.

Gambar 3.46 Tampilan halaman tentang aplikasi

Gambar 3.47 merupakan tampilan **notifikasi** ketika motor jatuh dan ketika diklik oleh pengguna, akan menuju Google Maps untuk menunjukkan lokasi motor jatuh.

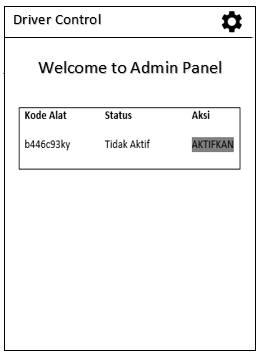
Gambar 3.48 merupakan tampilan ketika tombol **Logout** ditekan, akan muncul **konfirmasi** untuk keluar dari aplikasi. Jika memilih **OK** aplikasi akan ditutup. Jika memilih **BATAL** pengguna akan diarahkan ke halaman **utama**.

Gambar 3.44 Tampilan pesan konfirmasi keluar aplikasi



Gambar 3.48 Tampilan konfirmasi keluar aplikasi

Gambar 3.47 Tampilan notifikasi motor jatuh



Gambar 3.49 Tampilan halaman admin

Gambar 3.49 merupakan tampilan halaman **admin**, admin bertugas untuk mengaktifkan dan menonaktifkan **alat Driver Control**. Pertama admin harus **Login** terlebih dahulu untuk masuk ke halaman admin, kemudian terdapat **kode alat** dimana admin harus memastikan **status alat** tersebut aktif atau tidak aktif. Jika alat aktif, admin dapat menekan tombol **aksi NONAKTIFKAN** untuk menonaktifkan alat. Sedangkan untuk mengaktifkan alat admin dapat menekan tombol **aksi AKTIFKAN**.

**BAB IV**

**IMPLEMENTASI DAN PENGUJIAN**

Berdasarkan pada tahapan yang telah dijelaskan pada bab III, selanjutnya dilakukan tahap pembuatan dan pengujian sistem. Tahap pembuatan merupakan langkah mengubah desain yang telah dibuat sebelumnya menjadi kode-kode program. Sedangkan untuk tahap pengujian dilakukan dengan menggunakan pengujian kotak hitam.

1. **Tahap Pembuatan**

Pada tahap pembuatan aplikasi sesuai dengan kebutuhan dari sistem, tahap pertama yang dilakukan yaitu membuat basisdata dilanjutkan dengan tahap pembuatan *web service* dan tahap pembuatan aplikasi yang digunakan untuk pertukaran data dan dilanjutkan dengan tahap pembuatan aplikasi.

1. **Tahap Pembuatan Basisdata**

Pembuatan basisdata dilakukan menggunakan aplikasi ketiga phpMyAdmin. Dalam phpMyAdmin tersedia layanan pembuatan basisdata secara GUI, dengan memilih beberapa pilihan basisdata sudah dapat dibuat, dan menu SQL yang dapat digunakan untuk mengeksekusi *query* langsung ke basisdata.

#TABEL USER

CREATE TABLE `user` (

`id` int(11) NOT NULL AUTO\_INCREMENT,

`username` varchar(32) NOT NULL,

`password` varchar(32) NOT NULL,

`plat\_motor` varchar(10) NOT NULL,

`email` varchar(50) NOT NULL,

`kode\_user` varchar(32),

`kode\_alat` varchar(32) DEFAULT '0',

`status` int(2) DEFAULT '0',

PRIMARY KEY (`id`);

*Query* di atas menunjukkan bagaimana tabel user dibuat, id dibuat dengan tipe data integer dan sebagai Primary Key yang menujukkan bahwa id merupakan kunci dari tabel yang unik, yang bertujuan agar data yang disimpan tidak boleh sama. Pada kolom username, password, plat\_motor, kode\_user, dan email menggunakan tipe data varchar yang menunjukkan bahwa kolom bertipe data string dan pada kolom-kolom tersebut terdapat *query* NOT NULL yang menunjukkan bahwa kolom harus diisi dan tidak boleh kosong. Pada kolom kode\_alat yang bertipe data varchar dan status yang bertipe data integer diberi nilai DEFAULT '0'. Tabel user digunakan untuk menyimpan data pengguna aplikasi *Driver Control.*

#TABEL DITROL

CREATE TABLE `ditrol` (

`no` int(11) NOT NULL AUTO\_INCREMENT,

`latitude` varchar(20) DEFAULT NULL,

`longitude` varchar(20) DEFAULT NULL,

`kecepatan` varchar(10) DEFAULT NULL,

`jatuh` varchar(5) DEFAULT NULL,

`tanggal` varchar(20) DEFAULT NULL,

`jam` varchar(10) DEFAULT NULL,

PRIMARY KEY (`no`);

*Query* di atas menunjukkan bagaimana tabel ditrol dibuat, kolom no dibuat dengan tipe data integer, sebagai Primary Key*,* yang menujukkan bahwa no merupakan kunci dari tabel yang unik, yang bertujuan agar data yang disimpan tidak boleh sama.Terdapat *query* NOT NULL yang menunjukkan bahwa kolom harus diisi dan tidak boleh kosong, dan *query* AUTO\_INCREMENT berfungsi untuk penambahan data nomor secara otomatis. Pada kolom latitude, longitude, kecepatan, jatuh, tanggal, dan jam menggunakan tipe data varchar yang menunjukkan bahwa kolom bertipe data string. Terdapat *query* DEFAULT NULL yang menunjukkan bahwa kolom tersebut kondisi awalnya adalah kosong. Kolom latitude dan longitude digunakan sebagai koordinat untuk menampilkan lokasi kendaraan pada peta, dan kolom jatuh untuk notifikasi motor jatuh.

#TABEL BACARELAY

CREATE TABLE `bacarelay` (

`no` int(11) NOT NULL AUTO\_INCREMENT,

`relay` varchar(10) DEFAULT NULL,

PRIMARY KEY (`no`);

*Query* di atas menunjukkan bagaimana tabel bacarelay dibuat, kolom no dibuat dengan tipe data integer, sebagai Primary Key*,* yang menujukkan bahwa no merupakan kunci dari tabel yang unik, yang bertujuan agar data yang disimpan tidak boleh sama.Terdapat *query* NOT NULL yang menunjukkan bahwa kolom harus diisi dan tidak boleh kosong, dan *query* AUTO\_INCREMENT berfungsi untuk penambahan data nomor secara otomatis. Pada kolom relay menggunakan tipe data varchar yang menunjukkan bahwa kolom bertipe data string. Terdapat *query* DEFAULT NULL yang menunjukkan bahwa kolom relay kondisi awalnya adalah kosong. Kolom relay dibuat untuk menunjukkan status mesin motor hidup atau mati.

#TABEL DEVICES

CREATE TABLE `devices` (

`id` int(11) NOT NULL AUTO\_INCREMENT,

`device\_token` varchar(400) NOT NULL,

`email\_user` varchar(50) NOT NULL,

`waktu\_update` timestamp NOT NULL DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP,

`status\_login` int(2) NOT NULL DEFAULT '0',

PRIMARY KEY (`id`);

*Query* di atas menunjukkan bagaimana tabel devices dibuat, kolom id dan status\_login dibuat dengan tipe data integer. Kolom id sebagai Primary Key*,* yang menujukkan bahwa id merupakan kunci dari tabel yang unik, yang bertujuan agar data yang disimpan tidak boleh sama.Pada kolom device\_token, email\_user, dan waktu\_update menggunakan tipe data varchar yang menunjukkan bahwa kolom bertipe data string. Terdapat *query* NOT NULL yang menunjukkan bahwa kolom id, device\_token, email\_user, waktu\_update, dan status\_login harus diisi dan tidak boleh kosong. *Query* AUTO\_INCREMENT berfungsi untuk penambahan data id secara otomatis. Terdapat *query* DEFAULT '0' pada kolom status\_login yang menunjukkan bahwa kolom status\_login kondisi awalnya adalah bernilai 0. Tabel devices dibuat untuk memberikan **notifikasi email** saat pengguna melakukan **Login**, dan memberikan **notifikasi motor jatuh** kepada pengguna.

#TABEL ALAT

CREATE TABLE `alat` (

`id\_alat` int(11) NOT NULL AUTO\_INCREMENT,

`kode\_alat` varchar(32) NOT NULL,

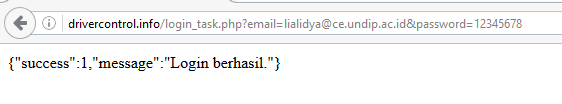
`status\_alat` varchar(2) NOT NULL DEFAULT '1',

PRIMARY KEY (`id\_alat`);

*Query* di atas menunjukkan bagaimana tabel alat dibuat, kolom id\_alat dibuat dengan tipe data integer, sebagai Primary Key*,* yang menujukkan bahwa id\_alat merupakan kunci dari tabel yang unik, yang bertujuan agar data yang disimpan tidak boleh sama.Terdapat *query* NOT NULL yang menunjukkan bahwa kolom id\_alat harus diisi dan tidak boleh kosong, dan *query* AUTO\_INCREMENT berfungsi untuk penambahan data id\_alat secara otomatis. Pada kolom kode\_alat dan status\_alat menggunakan tipe data varchar yang menunjukkan bahwa kolom kode\_alat dan status\_alat bertipe data string. Terdapat *query* NOT NULL yang menunjukkan bahwa kolom kode\_alat dan status\_alat harus diisi dan tidak boleh kosong. Terdapat *query* DEFAULT 1 pada kolom status\_alat yang menunjukkan bahwa kolom status\_alat kondisi awalnya adalah bernilai 1.

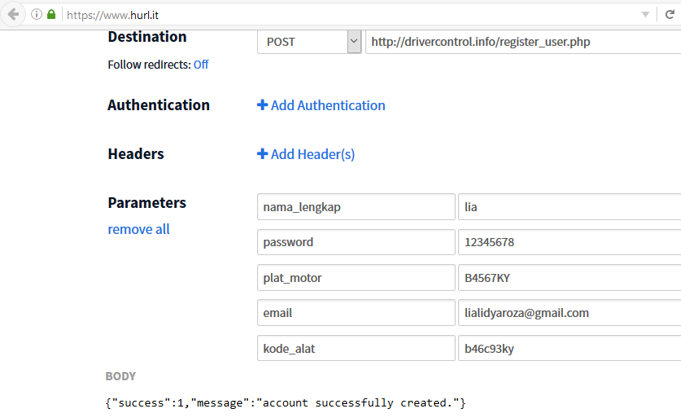
1. **Tahap Pembuatan Web Service**

Pembuatan *web service* ini berfungsi sebagai jembatan antara aplikasi Android dengan *server* basisdata. Cara kerja *web service* yaitu dengan mengambil data dari *server* basisdata kemudian mengubah data tersebut ke dalam format pertukaran data JSON. Pada *web service* terdapat tujuh belashalaman yang bertugas untuk memberikan layanan kepada aplikasi yaitu **login, register, read\_user, a, read\_a, read\_location, read\_speed, read\_record, relay, read\_status, update\_profil, update\_password, update\_status, register\_device, send\_notification\_all, read\_alat,** dan **update\_alat**.

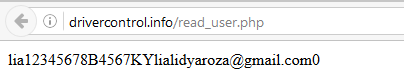
Halaman **Login** digunakan untuk melayani pengguna ketika ingin masuk ke dalam aplikasi *Driver Control*, untuk mengecek fungsi layanan login\_task.php dengan memasukkan [**http://drivercontrol.info/login\_task.php**](http://drivercontrol.info/login_task.php) beserta parameternya secara langsung pada *web browser.* Gambar 4.1 merupakan contoh penggunaan dari fungsi login\_task.php apabila pengguna memasukkan email dan password dengan benar maka akan memberikan nilai 1 dan menampilkan pesan **Login berhasil**.

Gambar 4.1 Layanan login

Pada fungsi register\_user.php untuk mengisi data pengguna pada tabel user, sehingga layanan **register\_user** akan berfungsi untuk membuat akun baru agar pengguna dapat **Login** dan menggunakan aplikasi. Method yang digunakan dalam membuat layanan **register\_user** adalah POST maka untuk mengecek fungsi layanan **register\_user** menggunakan bantuan situs **hurl.it** dengan cara memasukkan alamat [**http://drivercontrol.info/register\_user.php**](http://drivercontrol.info/register_user.php) dan mengisi parameter nama\_lengkap, password, plat\_motor, email, dan kode\_alat. Gambar 4.2 merupakan contoh penggunaan dari fungsi register\_user.php apabila pengguna memasukkan nama\_lengkap, password, plat\_motor, email, dan kode\_alat dengan benar maka akan memberikan **nilai 1** yang menujukkan pendaftaran akun baru berhasil.

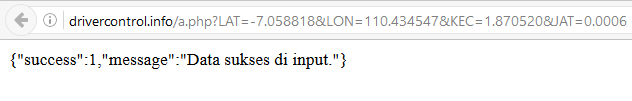


Gambar 4.2 Layanan register\_user

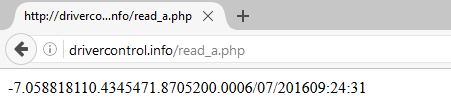
Pada fungsi read\_user.php merupakan layanan untuk membaca data pengguna yang terdapat pada tabel user dalam basisdata*.* Layanan ini berfungsi menampilkan data pengguna berupa username, password, plat\_motor, email dan kode\_user seperti pada Gambar 4.3. Untuk mengecek fungsi layanan **read\_user** dengan memasukkan [**http://drivercontrol.info/read\_user.php**](http://drivercontrol.info/read_user.php) beserta parameternya secara langsung pada *web browser.* Kemudian data pengguna ini akan ditampilkan di halaman **Profile** pada aplikasi.

Gambar 4.3 Layanan read\_user

Pada fungsi a.php untuk mengisi data dari perangkat *Driver Control* berupa latitude, longitude, kecepatan, jatuh, ke dalam tabel ditrol pada basisdata. Untuk mengecek fungsi layanan a.php dengan memasukkan [**http://drivercontrol.info/a.php**](http://drivercontrol.info/a.php) beserta parameternya secara langsung pada *web browser.* Ketika data benar maka akan tampil pesan **data sukses di input** di jendela web seperti Gambar 4.4.



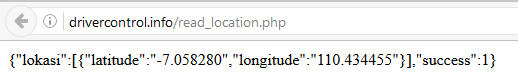
Gambar 4.4 Layanan a.php

Pada fungsi read\_a.php untuk membaca data latitude, longitude, kecepatan, jatuh, tanggal, dan jam dari tabel ditrol yang terdapat di basisdata seperti Gambar 4.5. Untuk mengecek fungsi layanan read\_a.php dengan memasukkan **http://drivercontrol.info/read\_a.php** beserta parameternya secara langsung pada *web browser.* Kemudian data pada tabel ditrol akan ditampilkan aplikasi Android *Driver Control* dalam fitur **Maps, Speed, Profile, Data record,** dan

Gambar 4.5 Layanan read\_a

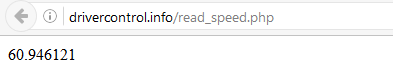
**Notifikasi motor jatuh**.

Pada fungsi read\_location.php untuk membaca data lokasi dari tabel ditrol yang terdapat di basisdata. Data lokasi pada tabel ditrol akan ditampilkan aplikasi Android Driver Control dalam fitur **Maps**. Untuk mengecek apakah fungsi layanan read\_location.php sudah berjalan atau belum, dapat memasukkan alamat **http://drivercontrol.info/read\_location.php** beserta parameternya secara langsung pada *web browser* seperti pada Gambar 4.6.

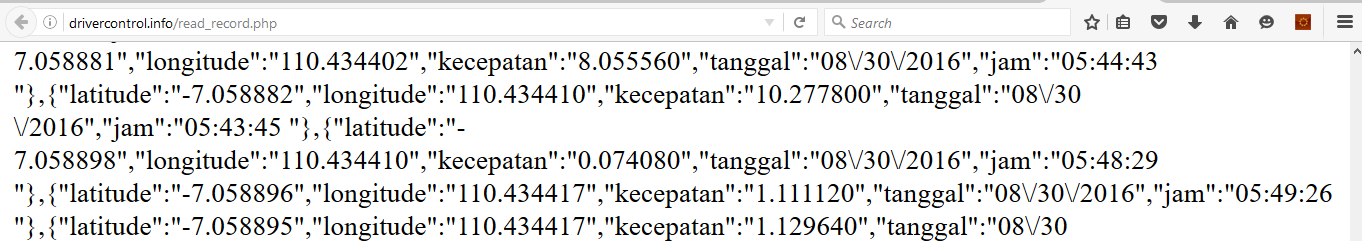


Gambar 4.6 Layanan read\_location

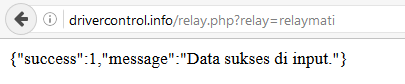
Pada fungsi read\_speed.php untuk membaca data kecepatan dari tabel ditrol yang terdapat di basisdata. Untuk mengecek fungsi layanan read\_speed.php dengan memasukkan **http://drivercontrol.info/read\_speed.php** beserta parameternya secara langsung pada *web browser* seperti pada Gambar 4.7. Data kecepatan pada tabel ditrol akan ditampilkan aplikasi Android Driver Control dalam fitur **Speed**.



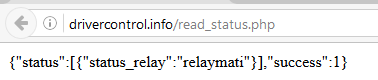
Gambar 4.7 Layanan read\_speed

Pada fungsi read\_record.php untuk membaca rekaman data dari tabel ditrol yang terdapat di basisdata. Untuk mengecek apakah fungsi layanan read\_record.php sudah berjalan baik atau belum dengan memasukkan **http://drivercontrol.info/read\_record.php** beserta parameternya secara langsung pada *web browser* seperti pada Gambar 4.8. Rekaman data pada tabel ditrol akan ditampilkan aplikasi Android Driver Control dalam fitur **Data record**.

Gambar 4.8 Layanan read\_record

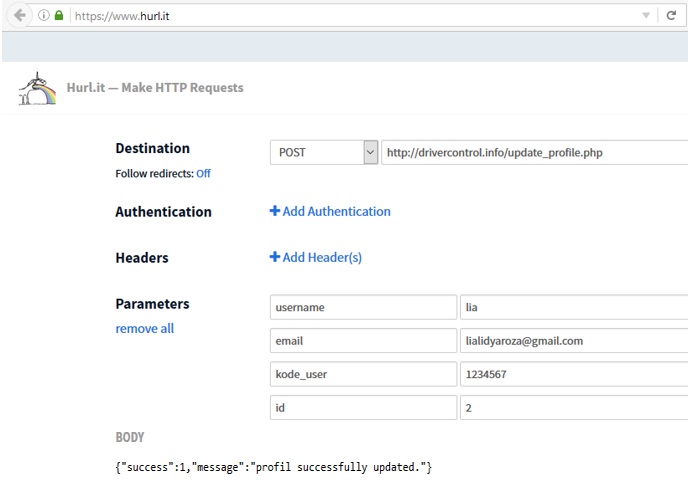
Pada fungsi relay.php untuk mengisi data dari perangkat *Driver Control* berupa relay ke dalam tabel bacarelay pada basisdata. Untuk mengecek fungsi layanan relay.php dengan memasukkan **http://drivercontrol.info/relay.php** beserta parameternya secara langsung pada *web browser.* Ketika data yang diisikan benar maka akan tampil pesan **data sukses di input** di jendela web seperti Gambar 4.9.

Gambar 4.9 Layanan relay

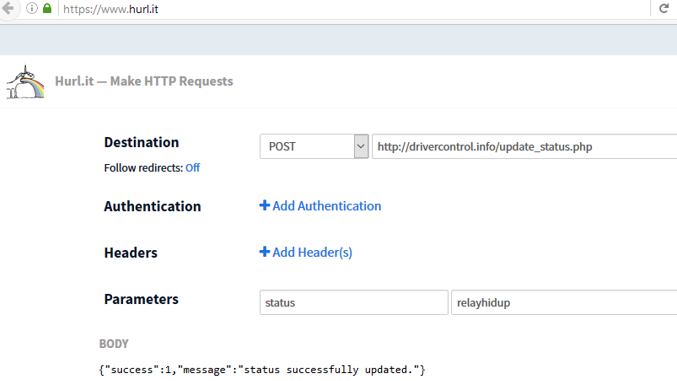
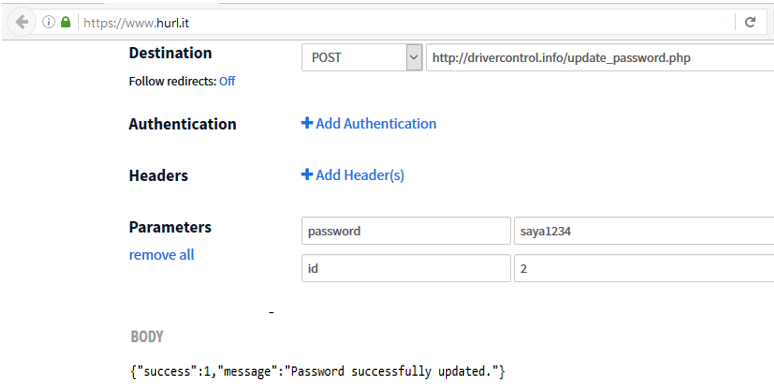
Pada fungsi read\_status.php untuk membaca data relay dari tabel bacarelay yang terdapat di basisdata. Untuk mengecek fungsi layanan read\_status.php dengan memasukkan **http://drivercontrol.info/read\_status.php** beserta parameternya secara langsung pada *web browser* seperti Gambar 4.10. Kemudian data pada tabel bacarelay akan ditampilkan aplikasi Android Driver Control dalam fitur **Lock** untuk menunjukkan status mesin motor hidup atau mati.

Gambar 4.10 Layanan read\_status

Pada fungsi update\_profile.php untuk memperbarui data pengguna dari tabel user yang terdapat di basisdata. Method yang digunakan dalam membuat layanan **update\_profile** adalah POST maka untuk mengecek fungsi layanan **update\_profile** menggunakan bantuan situs **hurl.it** dengan cara memasukkan alamat **http://drivercontrol.info/update\_profile.php** dan mengisi parameter username, email, kode\_user, dan id seperti pada Gambar 4.11.

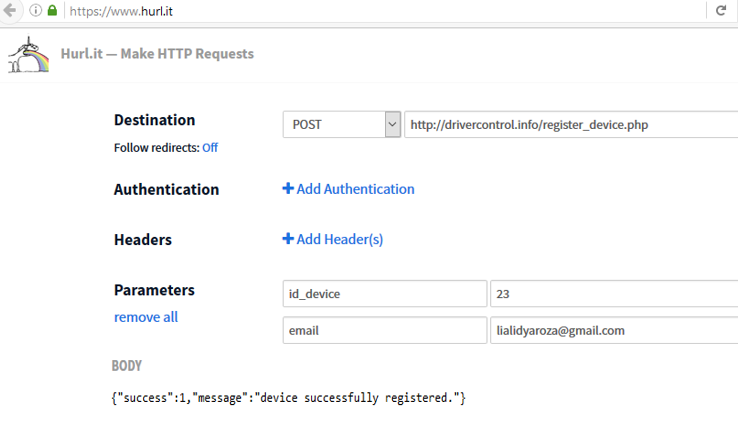
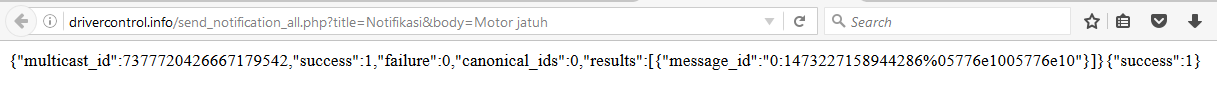
** Gambar 4.11Layanan update\_profil

Pada fungsi update\_password.php untuk memperbarui data pengguna dari tabel user yang terdapat di basisdata. Method yang digunakan dalam membuat layanan **update\_password** adalah POST maka untuk mengecek fungsi layanan **update\_password** menggunakan bantuan situs **hurl.it** dengan cara memasukkan alamat **http://drivercontrol.info/update\_password.php** dan mengisi parameter id, dan password seperti pada Gambar 4.12.

Pada fungsi update\_status.php untuk memperbarui data relay dari tabel bacarelay yang terdapat di basisdata. Method yang digunakan dalam membuat layanan **update\_status** adalah POST maka untuk mengecek fungsi layanan **update\_status** menggunakan bantuan situs **hurl.it** dengan cara memasukkan alamat **http://drivercontrol.info/update\_status.php** dan mengisi parameter username, email, kode\_user dan id seperti pada Gambar 4.13. Kemudian fungsi **update\_status** relay pada tabel bacarelay akan di terapkan pada aplikasi Android Driver Control dalam fitur **Lock** untuk mematikan atau menghidupkan mesin ****kendaraan secara otomatis dengan cara memutus atau menyambung relay mesin.

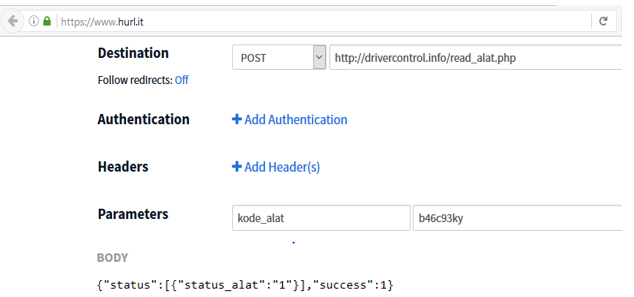
Gambar 4.12 Layanan update\_password

Gambar 4.13 Layanan update\_status

****Pada fungsi register\_device.php untuk mengisi data perangkat Android pada tabel devices, sehingga layanan **register\_device** akan berfungsi untuk mendaftarkan perangkat baru agar pengguna dapat memperoleh notifikasi melalui *email* ketika **Login** dan untuk menerima notifikasi motor jatuh. Method yang digunakan dalam membuat layanan **register\_device** adalah POST maka untuk mengecek fungsi layanan **register\_device** menggunakan bantuan situs **hurl.it** dengan cara memasukkan alamat **http://drivercontrol.info/register\_device.php** dan mengisi parameter id\_device, dan email. Gambar 4.14 merupakan contoh penggunaan dari fungsi register\_device.php apabila pengguna memasukkan id\_device, dan email dengan benar maka akan memberikan **nilai 1** yang menujukkan pendaftaran perangkat baru berhasil.

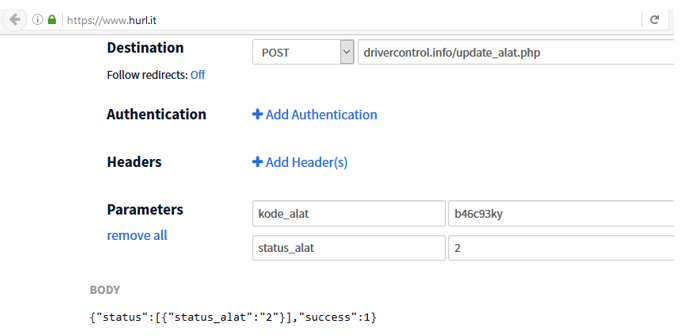
Gambar 4.15 Layanan send\_notification

Gambar 4.14 Layanan register\_device

Pada fungsi send\_notification\_all.php untuk mengirim notifikasi motor jatuh pada perangkat Android. Untuk mengecek fungsi layanan send\_notification\_all.php dengan cara menjatuhkan alat *Driver Control* sehingga secara otomatis akan mengirim notifikasi seperti Gambar 4.15.

Gambar 4.16 Layanan read\_alat

Pada fungsi read\_alat.php untuk melihat status alat dari tabel alat apakah aktif atau tidak aktif. Method yang digunakan dalam membuat layanan **read\_alat** adalah POST maka untuk mengecek fungsi layanan **read\_alat** menggunakan bantuan situs **hurl.it** dengan cara memasukkan alamat **http://drivercontrol.info/read\_alat.php** dan mengisi parameter kode\_alat dengan isinya. Gambar 4.16 merupakan contoh penggunaan dari fungsi register\_device.php apabila pengguna memasukkan kode\_alat, dan value dengan benar maka akan memberikan **nilai 1** yang menujukkan status\_alat 1 yang menandakan alat aktif.

****

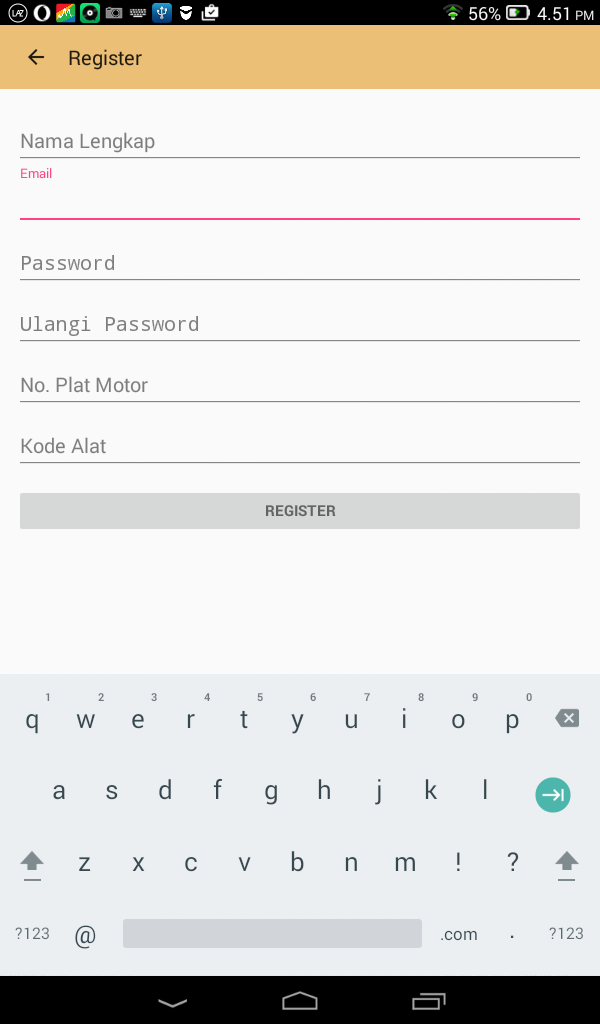
Gambar 4.17 Layanan update\_alat

Pada fungsi update\_alat.php untuk mengubah status alat pada tabel alat dari aktif menjadi tidak aktif dan sebaliknya. Method yang digunakan dalam membuat layanan **update\_alat** adalah POST maka untuk mengecek fungsi layanan **update\_alat** menggunakan bantuan situs **hurl.it** dengan cara memasukkan alamat **http://drivercontrol.info/update\_alat.php** dan mengisi parameter kode\_alat dan status\_alat. Gambar 4.17 merupakan contoh penggunaan dari fungsi update\_alat.php apabila pengguna memasukkan kode\_alat, dan status\_alat dengan benar maka akan status\_alat akan berubah. Nilai 1 pada status\_alat menunjukkan alat aktif, sedangkan nilai 2 menunjukkan alat tidak aktif. Pada Gambar 4.17 status\_alat 2 yang menandakan alat dinonaktifkan.

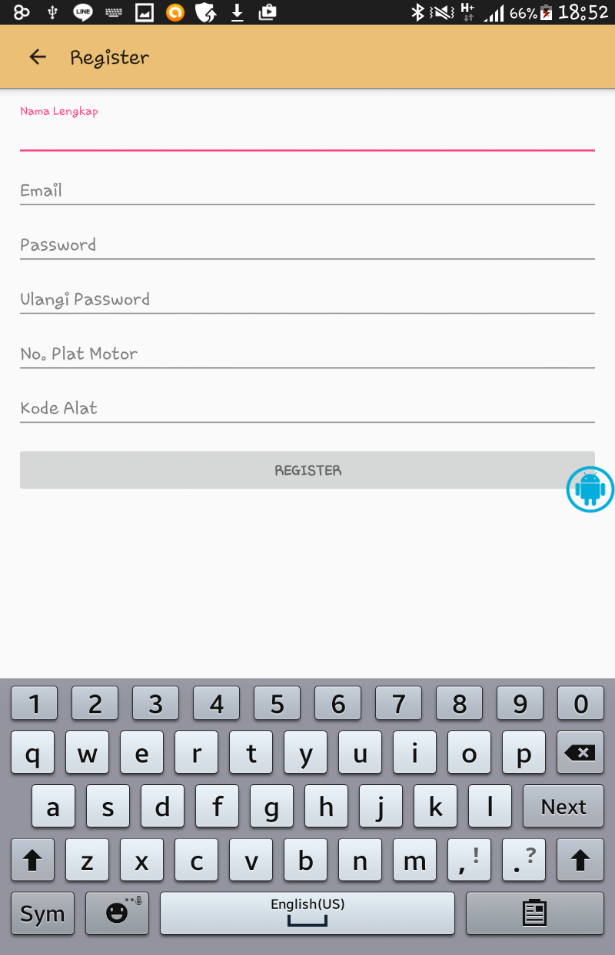
1. **Tahap Pembuatan Aplikasi**

Aplikasi Android *Driver Control* merupakan aplikasi *client* yang ditujukan kepada orang tua agar dapat memonitor anaknya ketika mengendarai sepeda motor, dimana dalam mengelola data kendaraan, aplikasi ini mengakses basis data melalui *web service* yang telah disediakan oleh *server.* Data yang dikirimkan ke basisdata merupakan data dari perangkat *Driver Control* yang merupakan bagian dari Tugas Akhir yang dibuat oleh saudari Khoirunisa Waskitaningrum. Keterhubungan antara Tugas Akhir kami terletak dari kedua sistem kami yang mengakses basis data yang sama dari *server* yang sama. Oleh sebab itu, pada tahap implementasi ini dijelaskan pembuatan aplikasi *client* pada *platform* Android*.*

1. **Register**

Hal pertama yang dilakukan pengguna untuk dapat masuk ke dalam aplikasi adalah membuat akun baru pada menu **Register**. Untuk masuk ke halaman **Register** pengguna harus menekan tombol **Register** yang terdapat pada halaman **Login**. Pada halaman **Register** terdapat enam kolom masukan yang harus diisi pengguna terlebih dahulu agar mempunyai akun baru. Enam Kolom tersebut yaitu kolom untuk **Nama Lengkap, Email, Password, Ulangi Password**, **No. Plat Motor**, dan **Kode Alat**. Tampilan aplikasi ketika membuka halaman **Register** atau pendaftaran akun baru yang terdapat pada Gambar 4.18, dan Gambar 4.19.

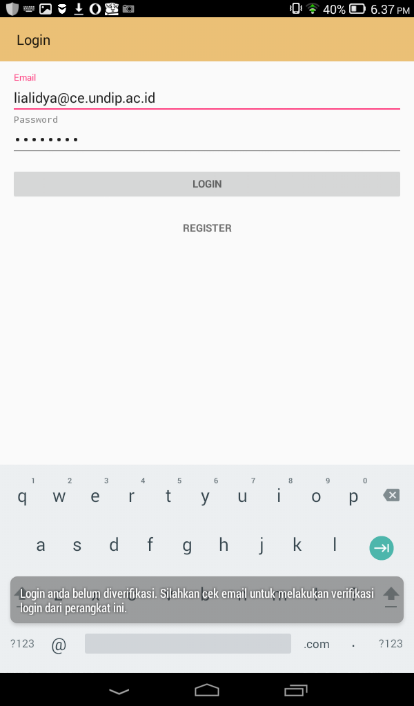
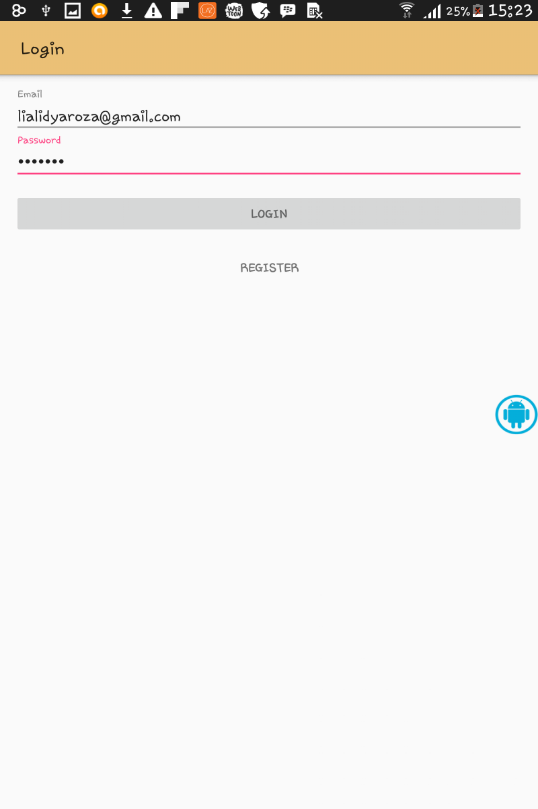
Gambar 4.18 Halaman register perangkat Android Lenovo



Gambar 4.19 Halaman register perangkat Android Samsung

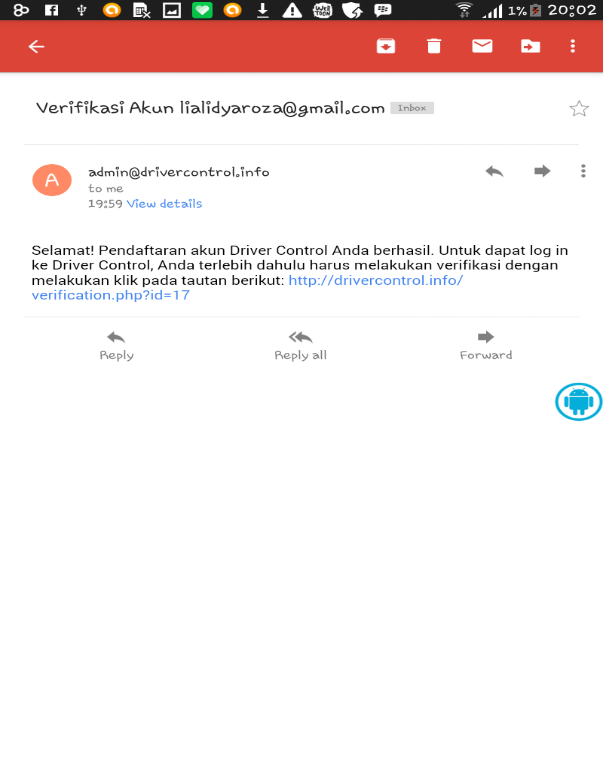
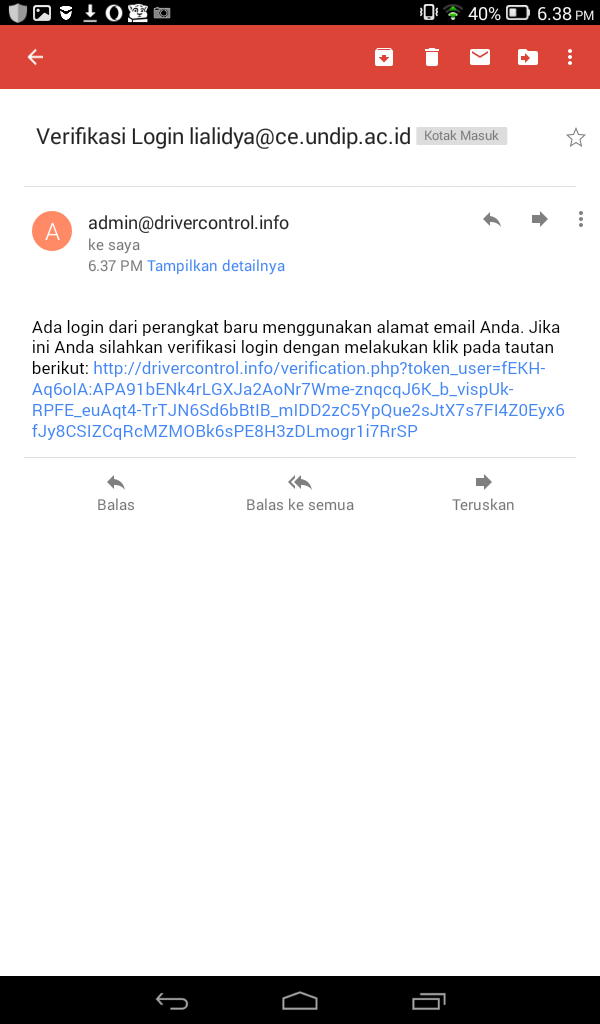
1. **Login**

Pada halaman **Login** terdapat dua kolom masukan yang wajib diisi oleh pengguna, kolom untuk **Email** dan kolom untuk **Password** seperti Gambar 4.20, dan Gambar 4.21. Melalui tombol **Login**, aplikasi akan melakukan autentikasi pengguna. Jika pengguna mengisi data dengan benar, sistem akan mengirim verifikasi *email* untuk masuk ke halaman **utama** aplikasi seperti Gambar 4.22, dan Gambar 4.23. Gambar 4.22 menunjukkan verifikasi melalui pengguna pertama dengan *email* **lialidya@ce.undip.ac.id**, dan Gambar 4.23 menunjukkan verifikasi melalui pengguna kedua dengan *email* **lialidyaroza@gmail.com**. Setelah melakukan verifikasi pengguna dapat masuk ke halaman **utama** aplikasi.



Gambar 4.21 Halaman login perangkat Android Samsung

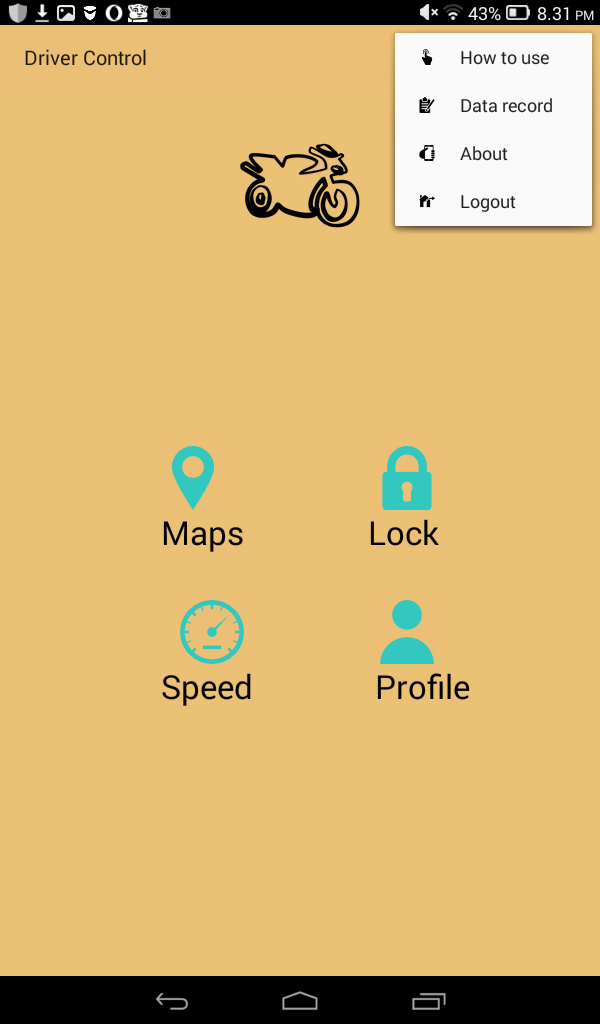
Gambar 4.20 Halaman login perangkat Android Lenovo



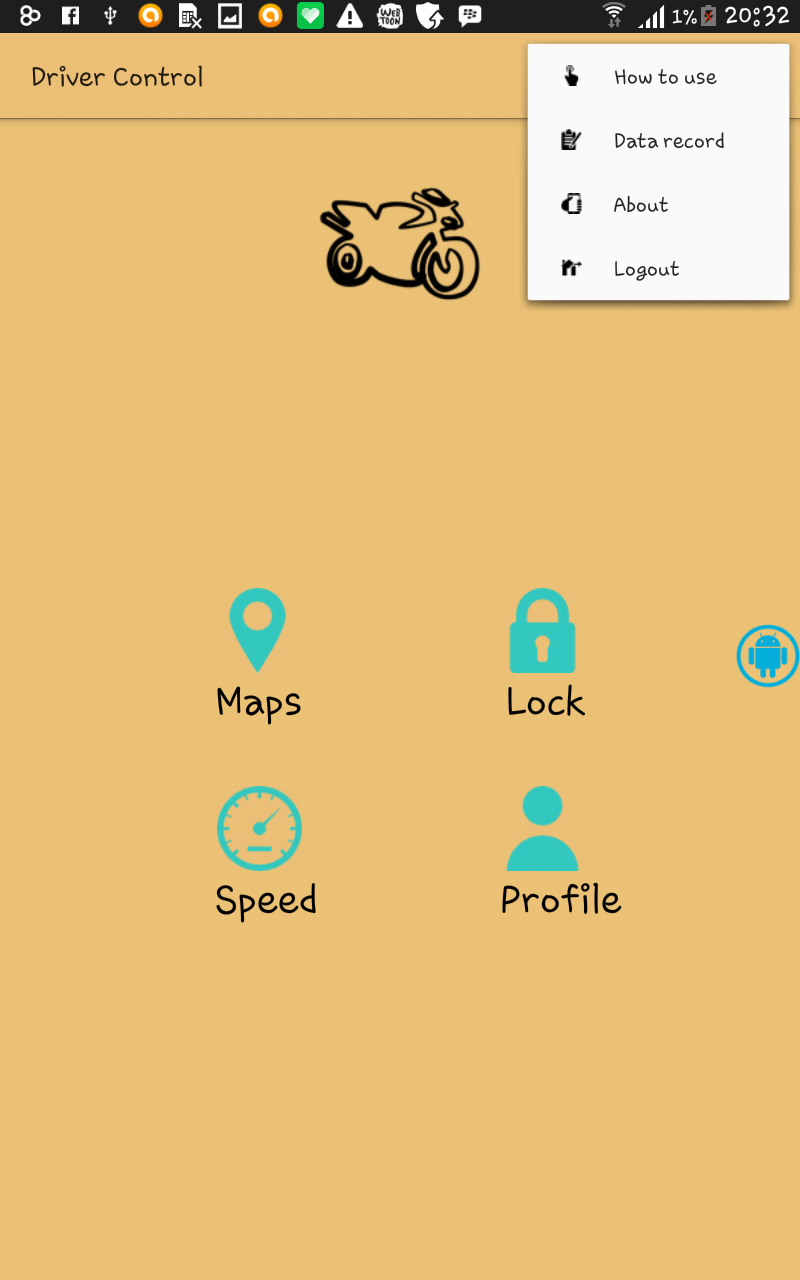
Gambar 4.22 Verifikasi pengguna 1 perangkat Android Lenovo

Gambar 4.23 Verifikasi pengguna 2 perangkat Android Samsung

1. **Halaman utama aplikasi**

****Gambar 4.24, dan Gambar 4.25 merupakan tampilan aplikasi Driver Control, pada halaman **utama** terdapat gambar logo aplikasi Driver Control yaitu motor, dan terdapat empat tombol yaitu **Maps**, **Speed**, **Lock**, dan **Profile**. Pada **slidebar menu** terdapat empat menu **How to use**, **Data record**, **About**, dan **Logout**. Tombol **Maps** untuk melihat peta lokasi kendaraan motor. Tombol **Speed** untuk melihat grafik kecepatan rata-rata pengemudi, tombol **Profile** untuk mengedit profile dan mengubah password, tombol **Lock** untuk mematikan mesin motor secara otomatis dari aplikasi. Sedangkan pada **slidebar menu** terdapat menu **How to use**untuk petunjuk penggunaan aplikasi, menu **Data record** untuk rekaman data pengemudi, menu **About** untuk melihat tentang pengembang aplikasi, dan menu **Logout** untuk keluar dari aplikasi.

Gambar 4.24 Halaman utama perangkat Android Lenovo

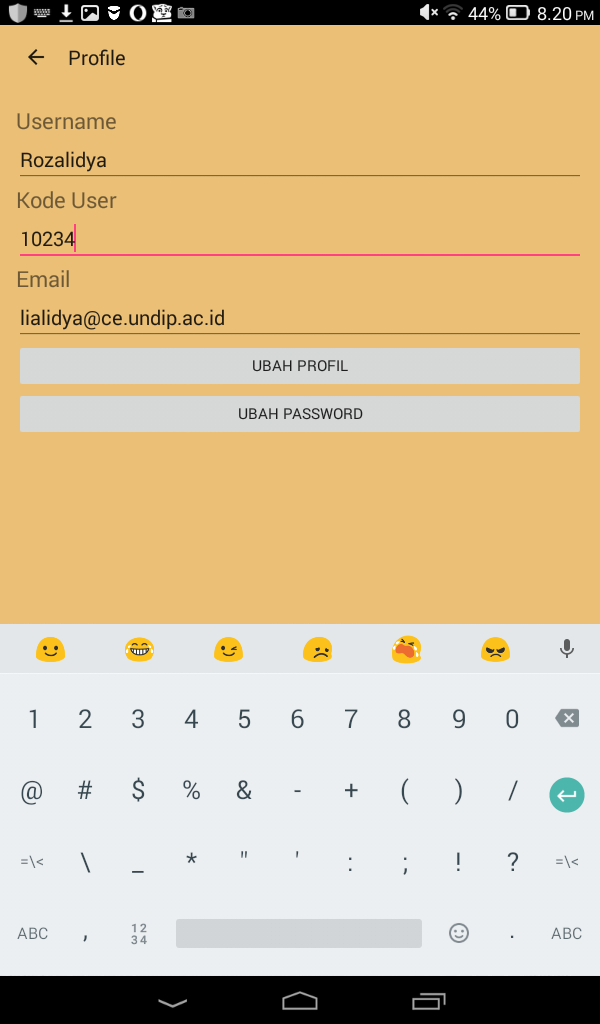
****

Gambar 4.25 Halaman utama perangkat Android Samsung

1. **Profile**

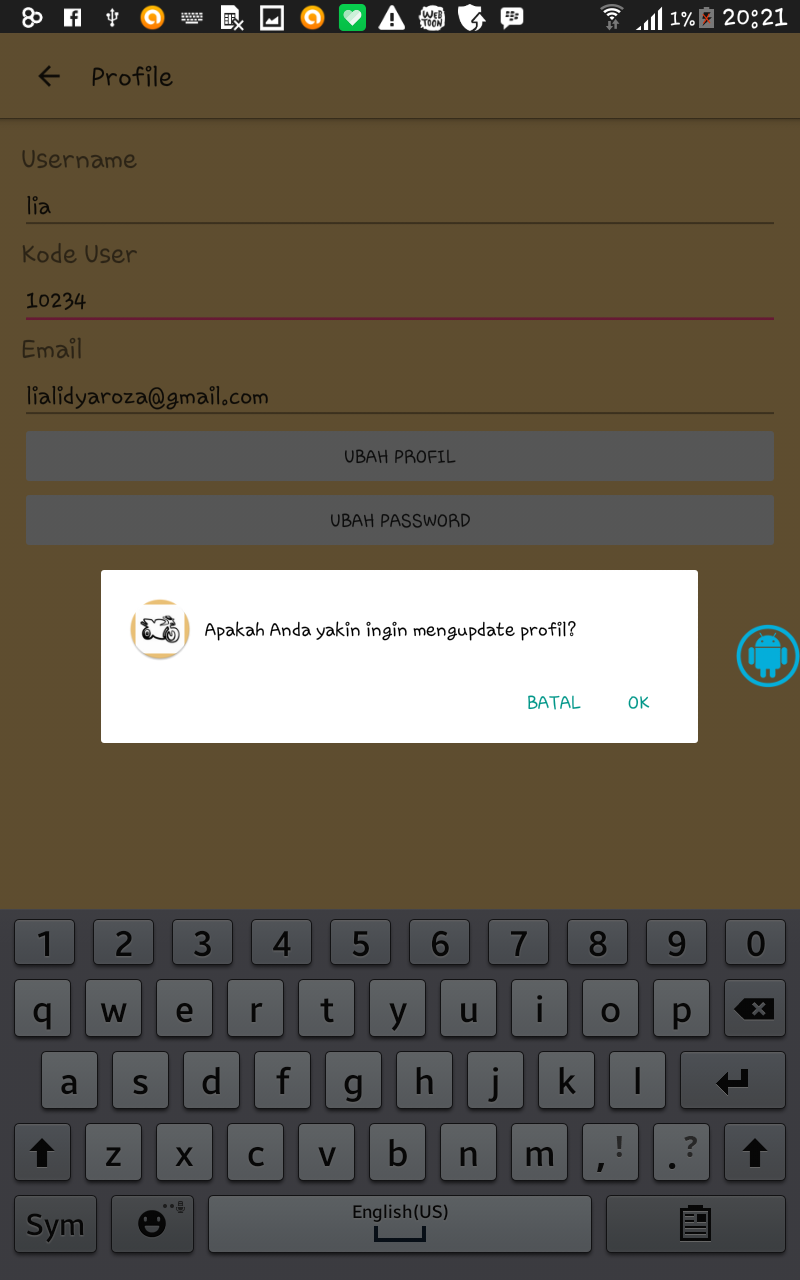
Ketika pengguna sudah berhasil masuk ke halaman **utama** aplikasi, pengguna hanya dapat mengakses menu **Petunjuk**, **Tentang aplikasi**, dan **Profile**. Ketika pengguna mencoba mengakses menu **Maps** akan muncul peringatan untuk mengisi **Kode User** terlebih dahulu pada halaman **Profile**. Gambar 4.26 merupakan tampilan peringatan untuk mengisi **Kode User**.

Gambar 4.27, dan Gambar 4.28 adalah menu **Profile** untuk mengubah data diri pengguna aplikasi dan konfirmasi **Kode User**. Pada halaman **Profile** terdapat empat kolom yang dapat diubah oleh pengguna yaitu **Username, Kode User,** dan **Email**, kemudian untuk menyimpan perubahan data pengguna harus menekan tombol **UBAH PROFIL**. Pada halaman **Profile** terdapat tombol **UBAH PASSWORD** yang berfungsi untuk mengubah kata sandi.



Gambar 4.27 Profile pengguna 1 pada perangkat Android Lenovo

Gambar 4.26 Perintah untuk mengisi Kode User

****

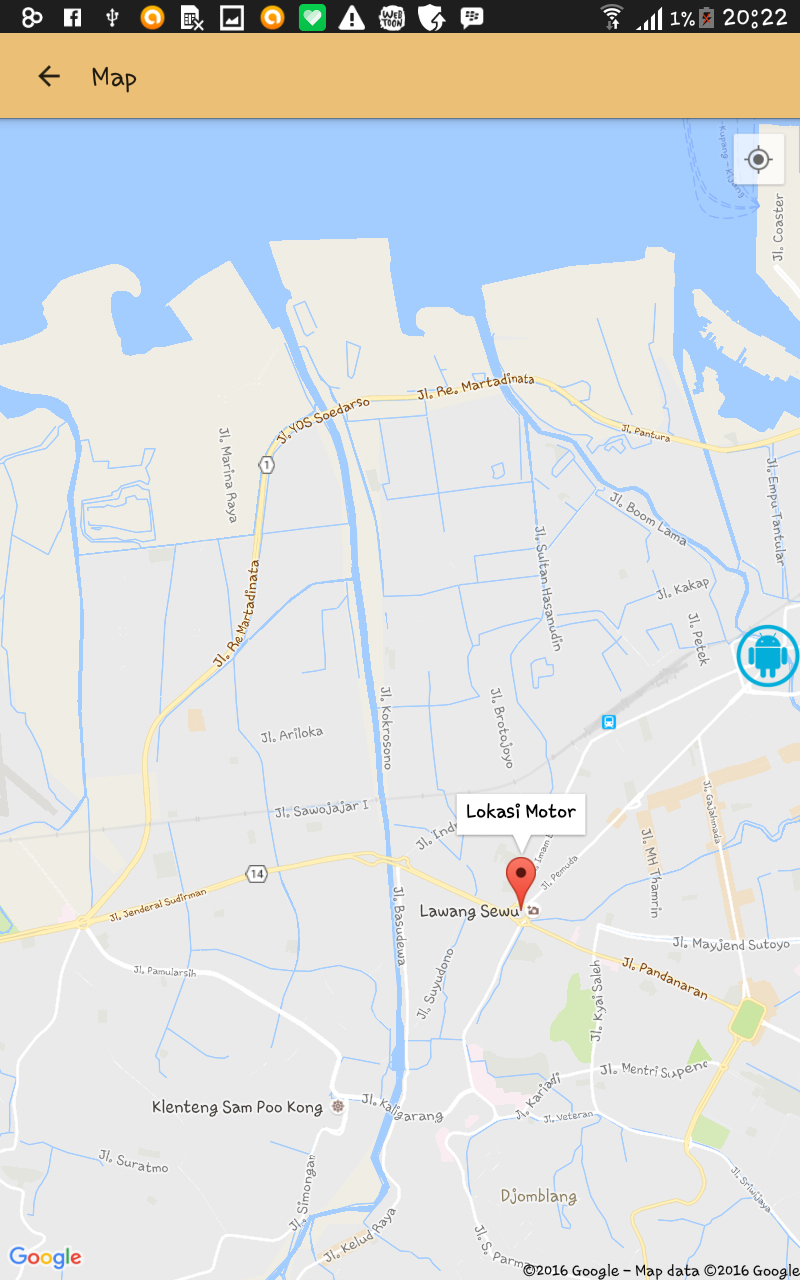
Gambar 4.28 Profile pengguna 2 pada perangkat Android Samsung

Aplikasi dapat berjalan dengan baik pada kedua perangkat Android walaupun pengguna aplikasi berbeda. Pada Gambar 4.27 menampilkan data pengguna pertama sebelum pengguna menekan tombol **UBAH PROFIL**. Sedangkan Gambar 4.28 menunjukkan data pengguna kedua setelah pengguna menekan tombol **UBAH PROFIL** sehingga aplikasi menampilkan **konfirmasi** untuk memperbarui profil.

1. **Maps**

**Maps** adalah salah satu fitur aplikasi *Driver Control* yang dapat menunjukkan lokasi kendaraan dalam waktu terbaru. Gambar 4.29, dan Gambar 4.30 merupakan tampilan halaman **Map** ketika pengguna berhasil menemukan lokasi kendaraan anak menggunakan layanan Google Maps yang ditunjukkan oleh **penanda**. Halaman **Map **juga terdiri dari tombol yang berfungsi untuk kembali ke halaman sebelumnya.

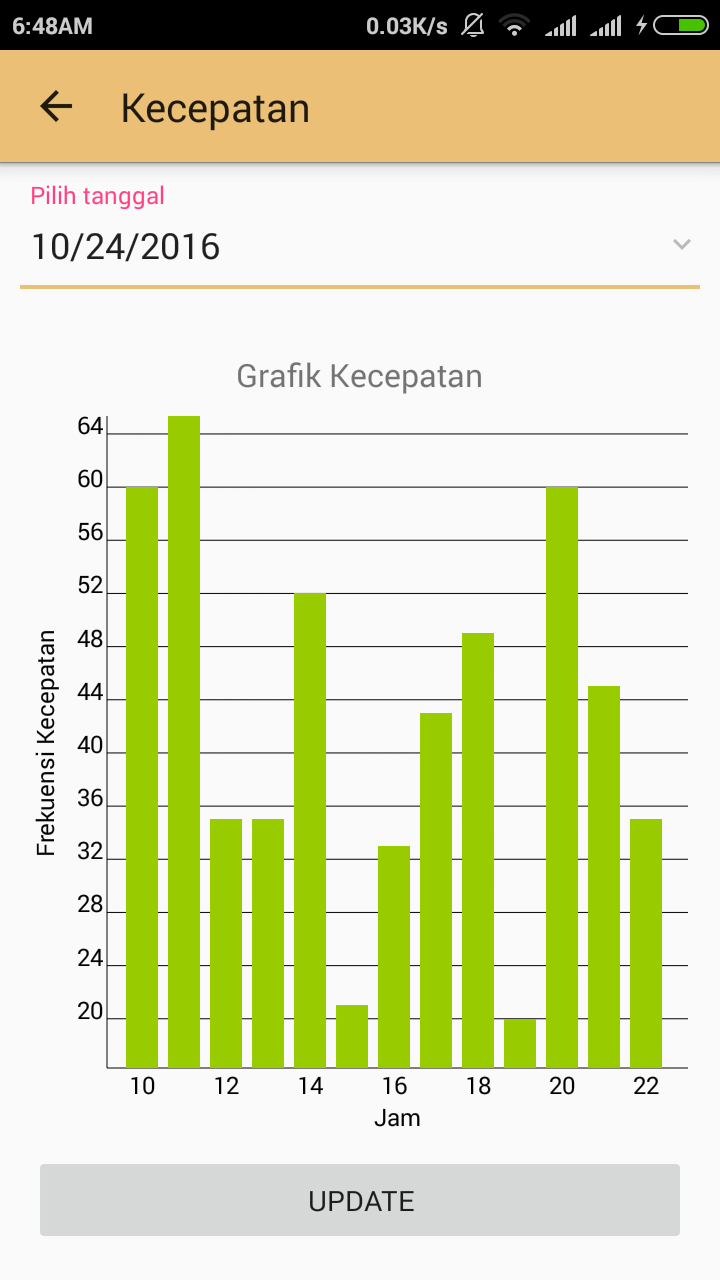
Gambar 4.29 Map pada perangkat Android Lenovo



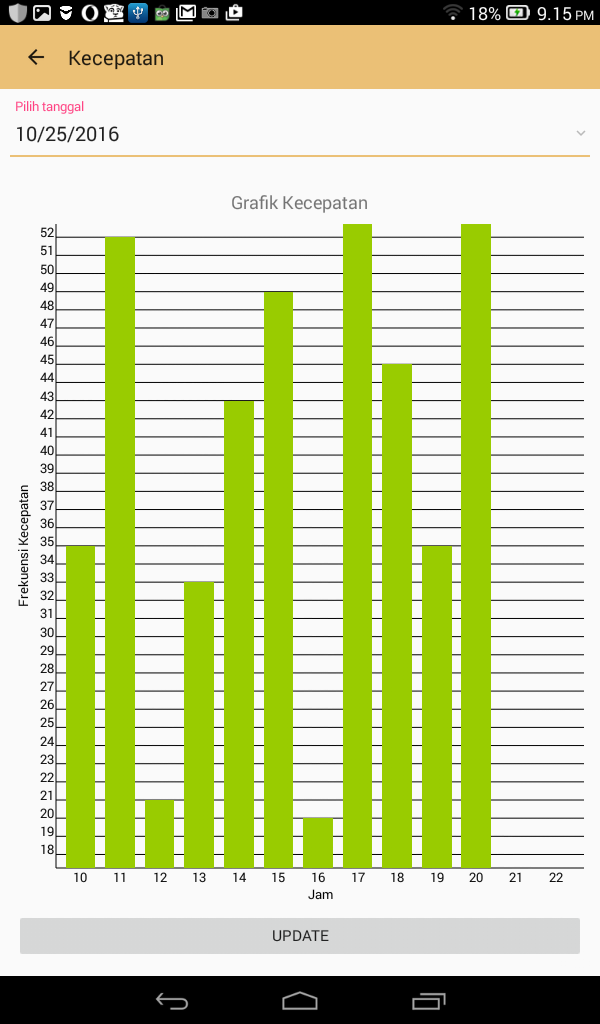
Gambar 4.30 Map pada perangkat Android Samsung

Aplikasi dapat berjalan dengan baik pada kedua perangkat Android dengan pengguna aplikasi yang berbeda, terlihat pada Gambar 4.29, dan Gambar 4.30 dimana aplikasi dapat menunjukkan lokasi motor.

1. **Speed**

**Speed** merupakan fitur aplikasi *Driver Control* yang dapat menampilkan grafik kecepatan pengemudi. Data kecepatan yang ditampilkan dalam satu halaman merupakan data kecepatan pengemudi dalam satu hari terakhir. Halaman **Kecepatan** juga terdiri dari tombol yang berfungsi untuk kembali ke halaman sebelumnya seperti Gambar 4.31, dan Gambar 4.32.

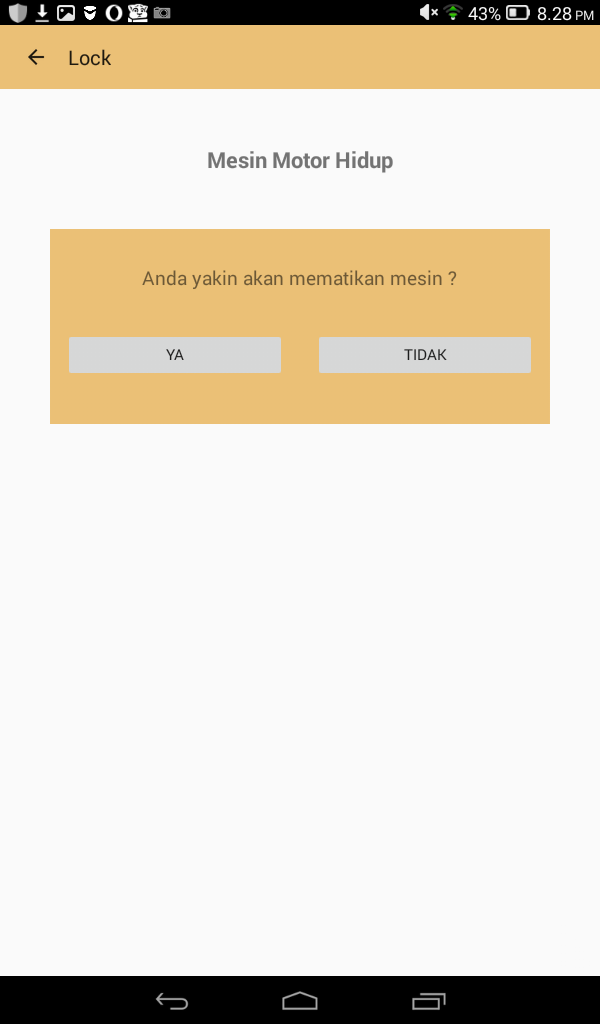
Gambar 4.31 Kecepatan pada perangkat Android 1 hari 1



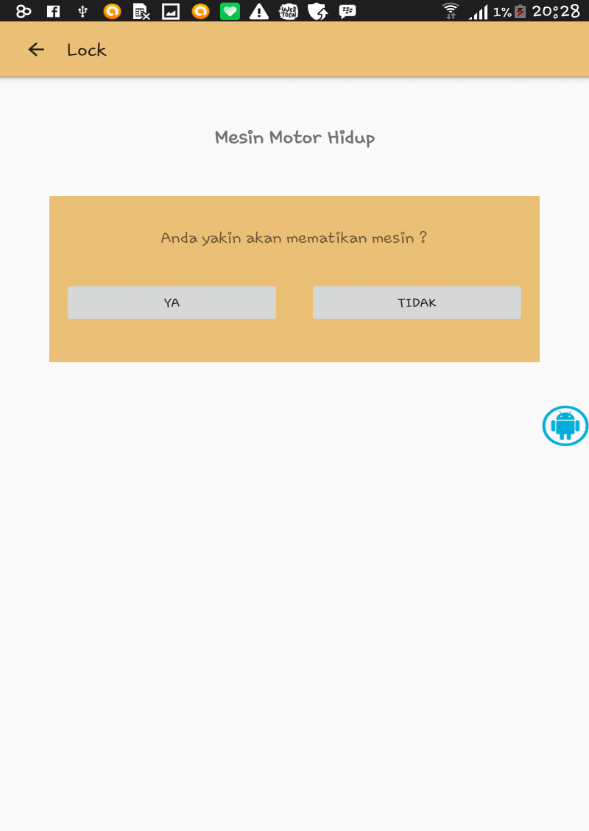
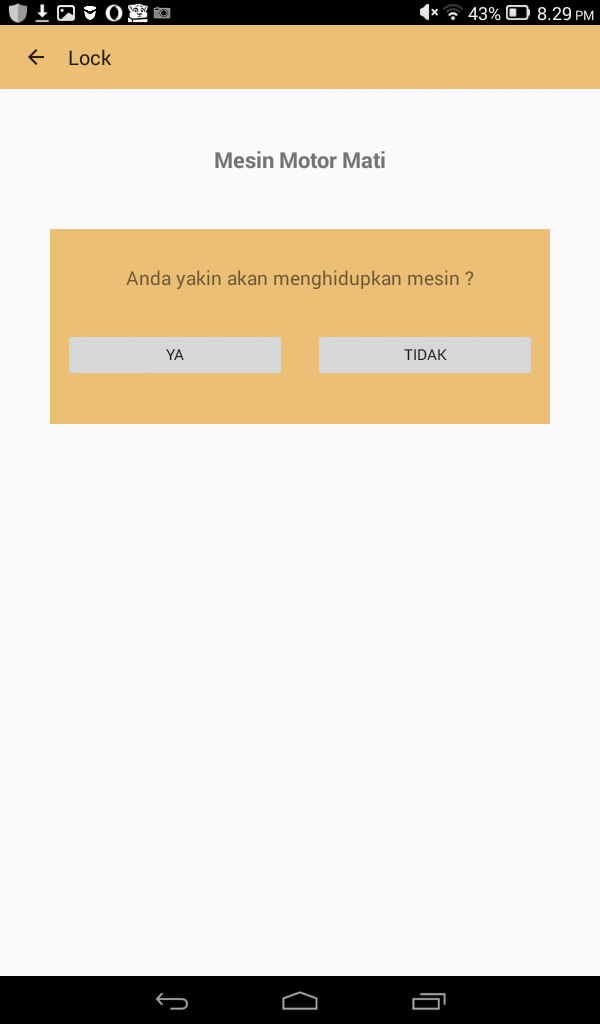
Gambar 4.32 Kecepatan pada perangkat Android 2 hari 2

Aplikasi dapat berjalan dengan baik pada kedua perangkat Android dengan pengguna aplikasi yang berbeda, terlihat pada Gambar 4.31, dan Gambar 4.32 dimana aplikasi dapat menunjukkan grafik kecepatan rata-rata pengemudi dalam hari yang berbeda.

1. **Lock**

**Lock** merupakan fitur aplikasi *Driver Control* yang dapat mematikan mesin motor secara otomatis. Ketika mesin motor hidup, halaman **Lock** akan menampilkan teks **Mesin Motor Hidup** dan **konfirmasi** untuk mematikan mesin motor seperti pada Gambar 4.33, dan Gambar 4.34. Jika pengguna memilih **YA** maka aplikasi akan mengirimkan perintah untuk mematikan mesin motor, dan jika memilih **TIDAK** maka aplikasi tetap berada pada halaman **Lock** dan mesin motor tetap hidup. Sebaliknya, ketika mesin motor mati, halaman **Lock** akan menampilkan teks **Mesin Motor Mati** dan **konfirmasi** untuk menghidupkan mesin motor seperti pada Gambar 4.35 dan Gambar 4.36. Halaman **Lock** juga terdiri dari tombol yang berfungsi untuk kembali ke halaman sebelumnya.

Gambar 4.33 Lock ketika mesin hidup pengguna 1



Gambar 4.34 Lock ketika mesin hidup pengguna 2

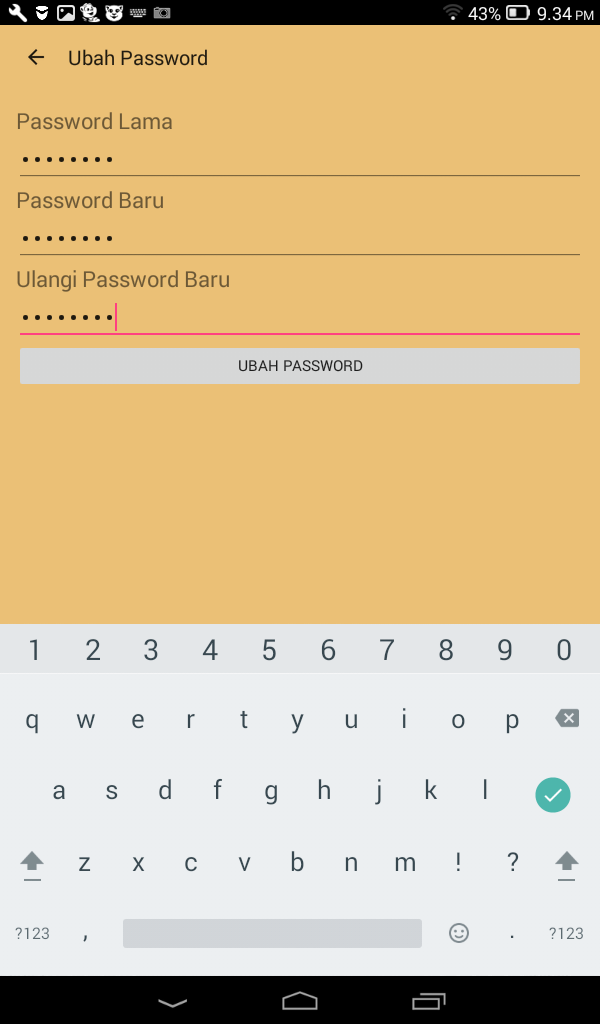
Gambar 4.35 Lock ketika mesin mati pengguna 1



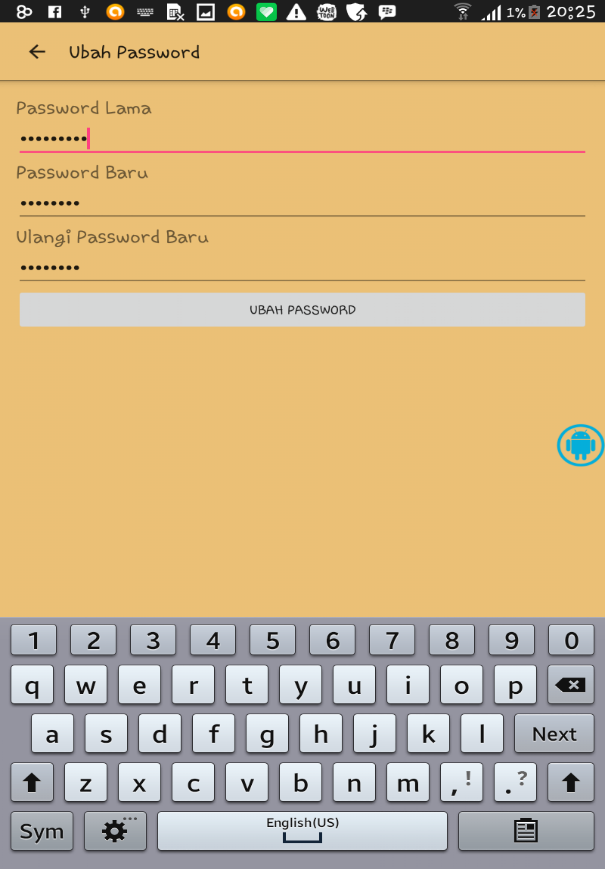
Gambar 4.36 Lock ketika mesin mati pengguna 2

Aplikasi dapat berjalan dengan baik pada kedua perangkat Android dengan pengguna aplikasi yang berbeda, terlihat pada Gambar 4.33, Gambar 4.34, Gambar 4.35, dan Gambar 4.36 dimana aplikasi dapat menunjukkan status mesin motor dan konfirmasi untuk mematikan atau menghidupkan mesin motor secara otomatis.

1. **Ubah Password**

Pada halaman **Profile** terdapat tombol **UBAH PASSWORD** ketika ditekan akan menuju halaman **Ubah Password** seperti pada Gambar 4.37, dan Gambar 4.38. Pada halaman **Ubah Password** terdapat tiga kolom yang harus diisi oleh pengguna jika ingin mengubah *password*yaitu kolom **Password Lama, Password Baru,** dan **Ulangi Password Baru**. Setelah mengisi ketiga kolom pengguna harus menekan tombol **UBAH PASSWORD** agar kata sandi disimpan pada basisdata, kemudian pengguna harus melakukan konfirmasi, jika pengguna memilih **OK** maka kata sandi akan disimpan ke dalam basisdata dan pengguna harus melakukan **Login** kembali. Sedangkan jika pengguna memilih **BATAL** maka aplikasi tetap pada halaman **Ubah** Password. Pada halaman **Ubah Password** juga terdapat tombol yang berfungsi ****untuk kembali ke halaman sebelumnya.

Gambar 4.37 Ubah Password perangkat Android Lenovo

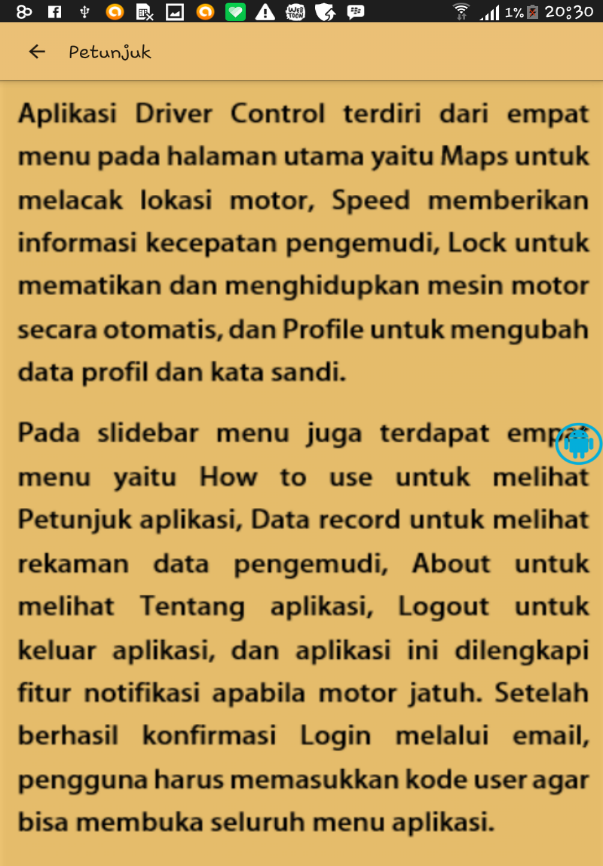
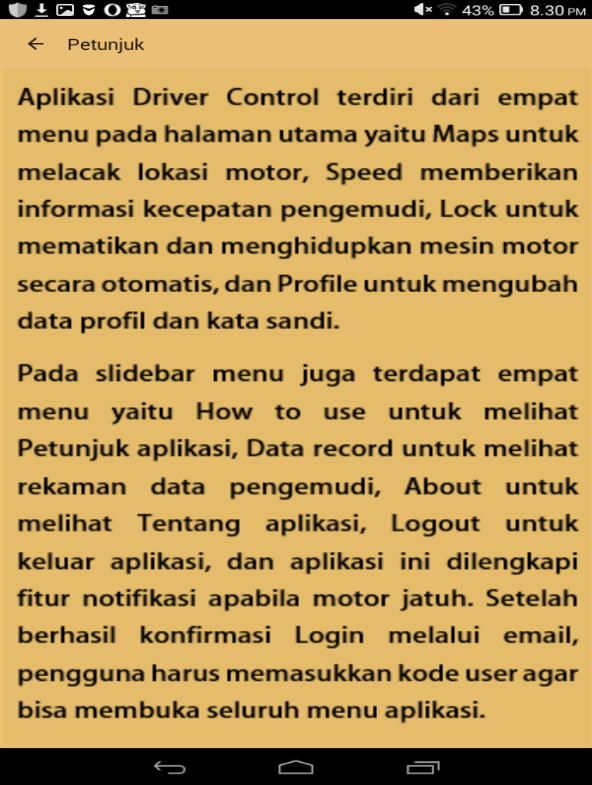
****

Gambar 4.38 Ubah Password perangkat Android Samsung

Aplikasi dapat berjalan dengan baik pada kedua perangkat Android dengan pengguna aplikasi yang berbeda, terlihat pada Gambar 4.37, dan Gambar 4.38 yang menunjukkan pengguna aplikasi dapat merubah kata sandi sesuai keinginan.

1. **How to use**

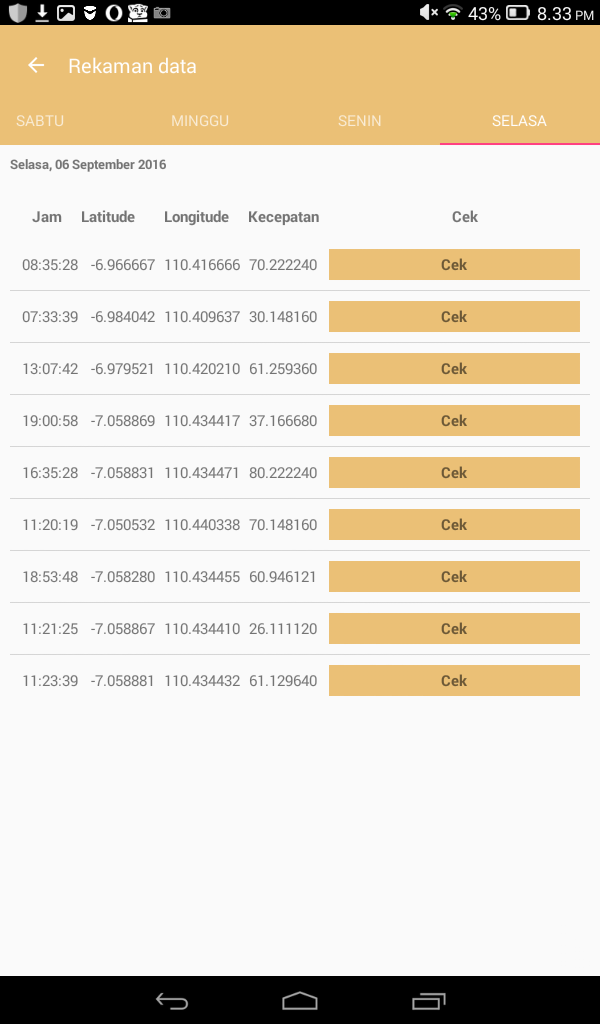
**How to use** merupakan menu aplikasi *Driver Control* yang memberikan petunjuk cara pemakaian aplikasi. Untuk menuju halaman **Petunjuk**, pengguna harus menekan tombol **How to use** pada **slidebar menu**. Pada halaman **Petunjuk** terdapat tombol yang berfungsi untuk kembali ke halaman sebelumnya seperti pada Gambar 4.39, dan Gambar 4.40.

****

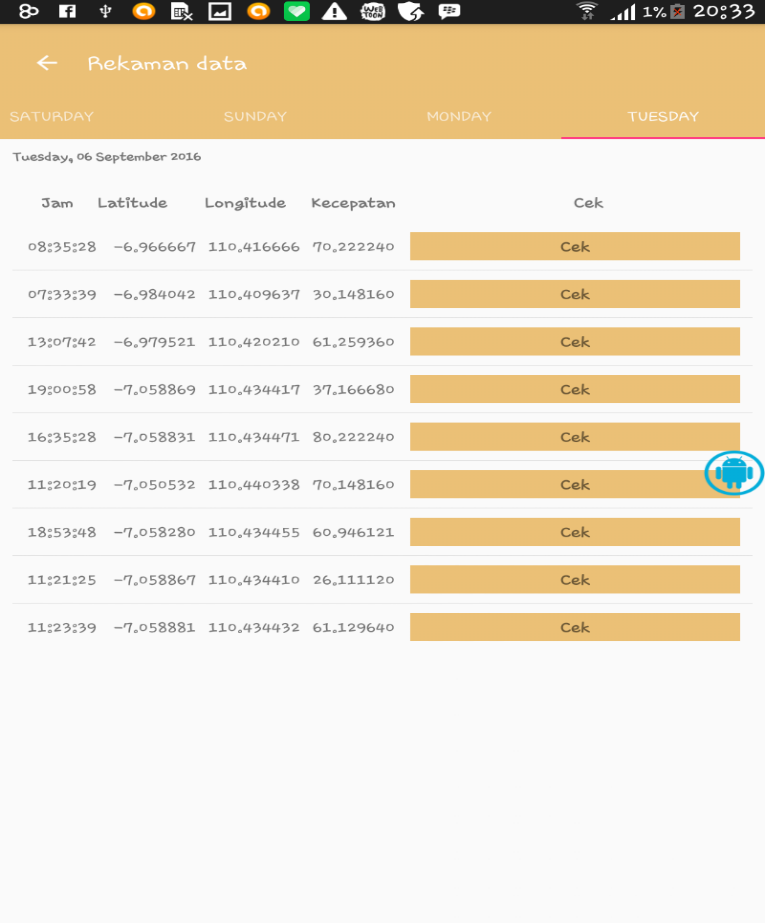
Gambar 4.40 Petunjuk pada perangkat Android Samsung

Gambar 4.39 Petunjuk pada perangkat Android Lenovo

1. **Data record**

**Data record** merupakan menu aplikasi *Driver Control* yang merekam data aktivitas kendaraan selama digunakan, data yang direkam berupa **latitude, longitude, kecepatan, jam,** dan **tanggal**. Data direkam selama satu minggu terakhir saat kendaraan digunakan. Untuk menuju halaman **Rekaman data**, pengguna harus menekan tombol **Data record** pada **slidebar menu**. Pada halaman **Rekaman data** terdapat tombol aksi **Cek**yang berfungsi untuk melihat peta lokasi kendaraan, dan terdapat tombol yang berfungsi untuk kembali ke halaman sebelumnya seperti pada Gambar 4.41, dan Gambar 4.42.

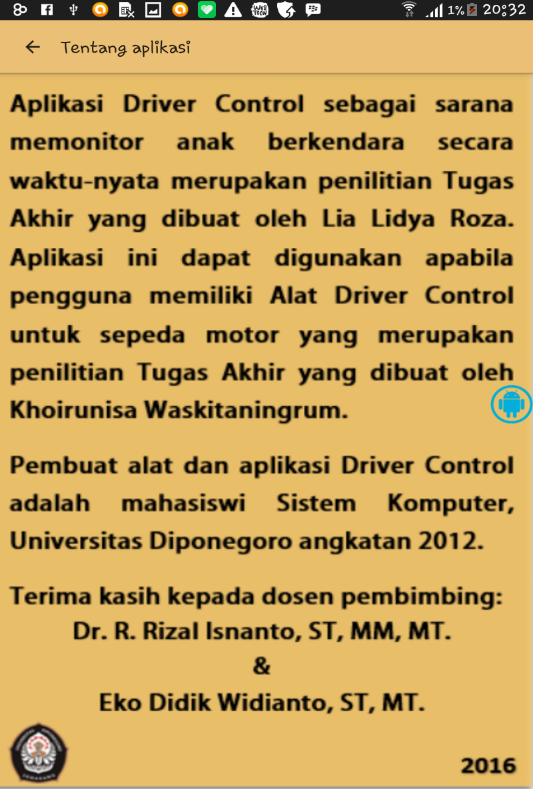
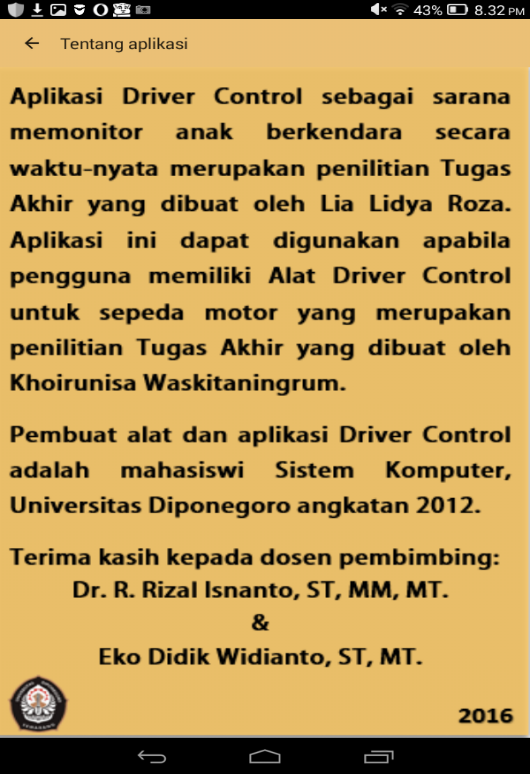
Gambar 4.41 Data record pada perangkat Android Lenovo



Gambar 4.42 Data record pada perangkat Android Samsung

Aplikasi dapat berjalan dengan baik pada kedua perangkat Android dengan pengguna aplikasi yang berbeda, terlihat pada Gambar 4.41, dan Gambar 4.42 yang menunjukkan aplikasi dapat menampilkan rekaman data pengemudi berupa **latitude**, **longitude** sebagai koordinat lokasi motor, **kecepatan** pengemudi, **jam** dan **tanggal**.

1. **About**

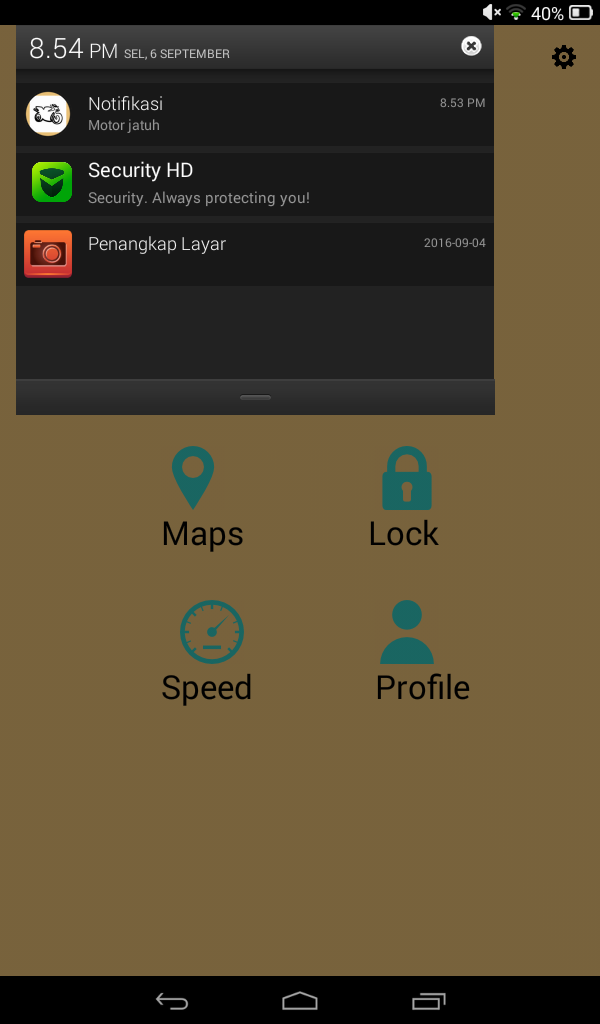
**About** merupakan menu aplikasi *Driver Control* untuk mengetahui tentang pengembang aplikasi. Untuk menuju halaman **Tentang aplikasi**, pengguna harus menekan tombol **About** pada **slidebar menu**. Pada halaman **Tentang aplikasi** terdapat tombol yang berfungsi untuk kembali ke halaman sebelumnya seperti pada Gambar 4.43, dan Gambar 4.44.

Gambar 4.44 Tentang aplikasi pada perangkat Android Lenovo

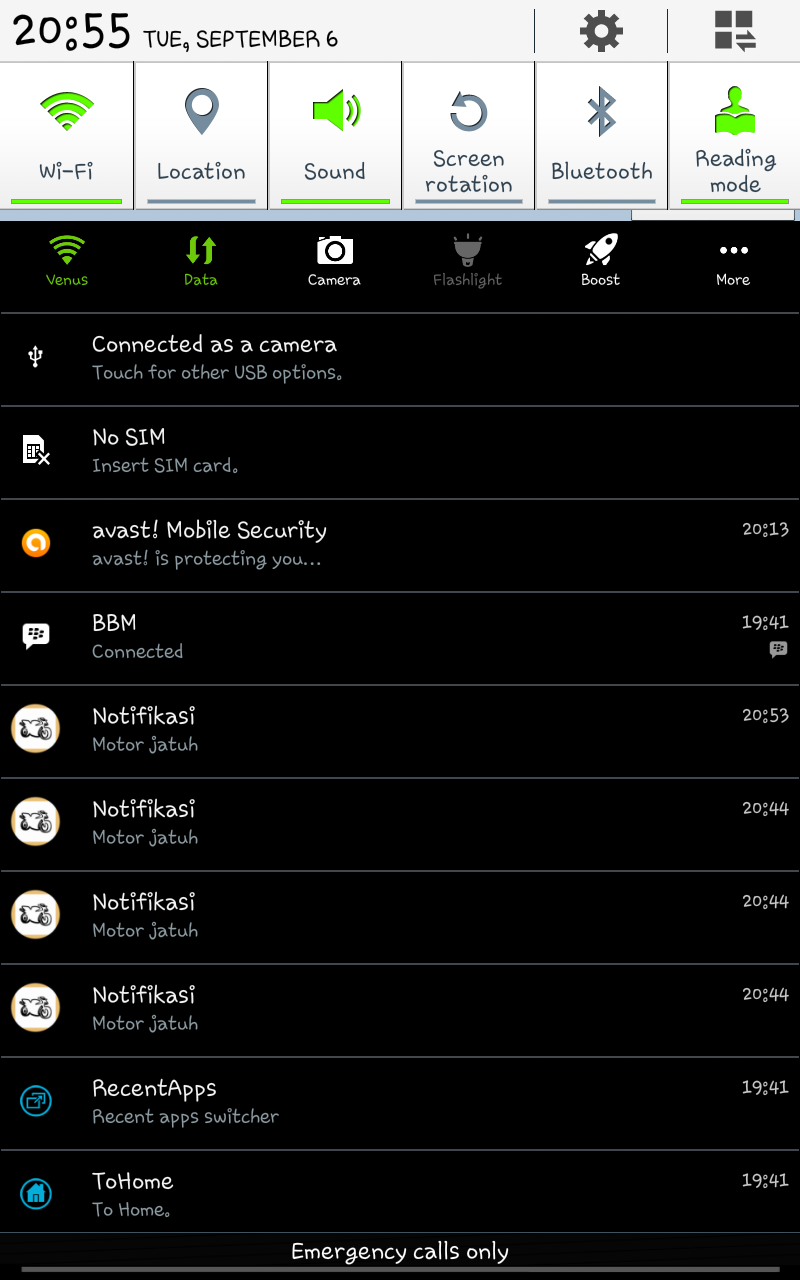
Gambar 4.43 Tentang aplikasi pada perangkat Android Samsung

Aplikasi dapat berjalan dengan baik pada kedua perangkat Android dengan pengguna aplikasi yang berbeda, terlihat pada Gambar 4.43, dan Gambar 4.44 yang menunjukkan aplikasi dapat menampilkan tentang pengembang aplikasi.

1. **Notifikasi Motor Jatuh**

**Notifikasi motor jatuh** adalah salah satu menu aplikasi *Driver Control* yang dapat diterima pengguna ketika motor jatuh. Notifikasi dapat dilihat pengguna selama Internet pada perangkat Android dalam keadaan aktif.

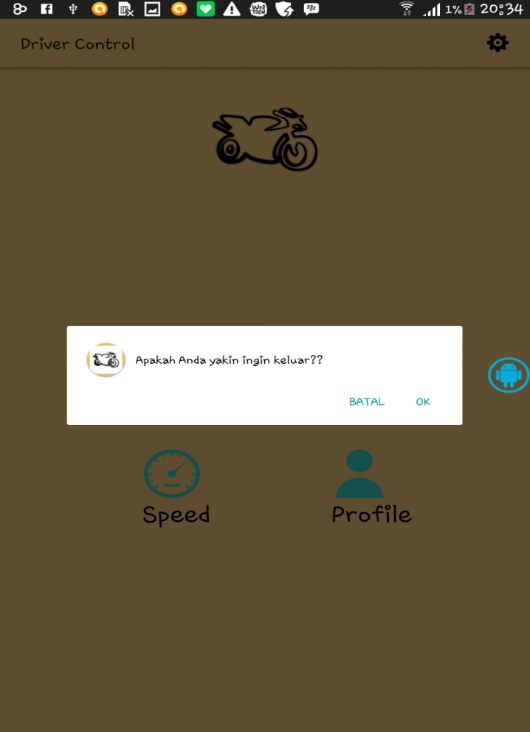
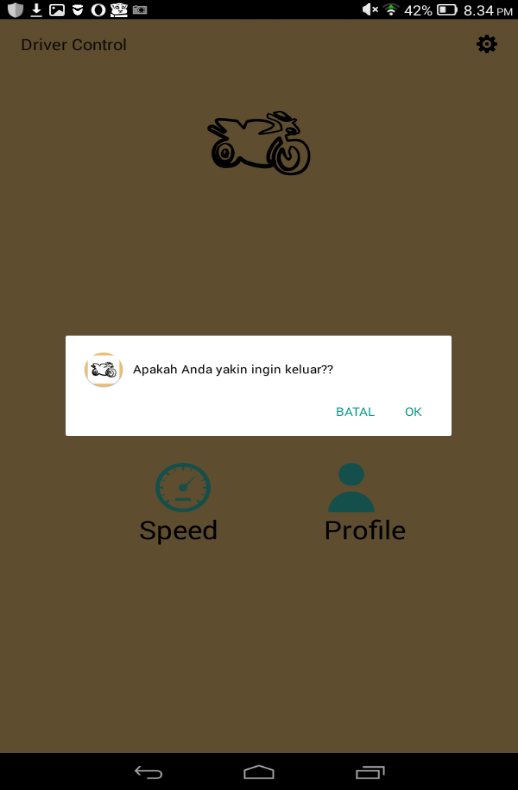
Gambar 4.45 Notifikasi pada perangkat Android Lenovo



Gambar 4.46 Notifikasi pada perangkat Android Samsung

Aplikasi dapat berjalan dengan baik pada kedua perangkat Android dengan pengguna aplikasi yang berbeda, terlihat pada Gambar 4.45, dan Gambar 4.46 yang menunjukkan aplikasi dapat menampilkan notifikasi pada perangkat Android di saat yang bersamaan ketika motor jatuh pada pukul 20.53.

1. **Logout**

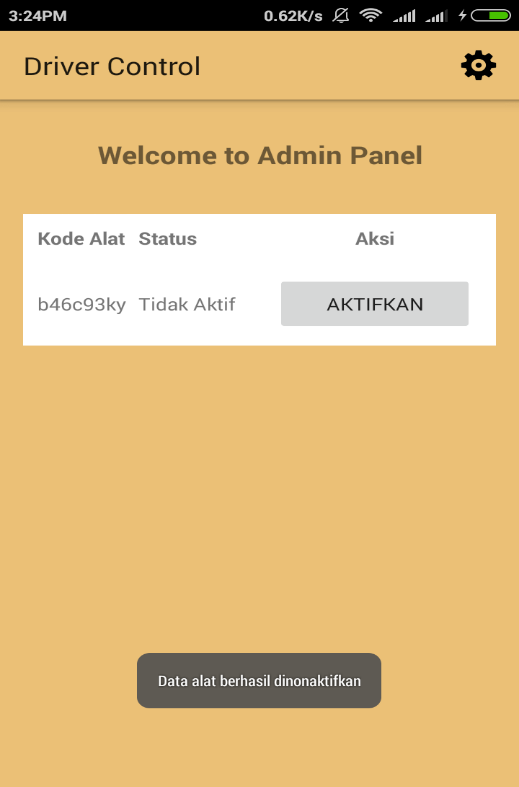
**Logout** adalah menu aplikasi yang berfungsi untuk keluar dari aplikasi. Ketika pengguna menekan tombol **Logout** pada **slidebar menu**, akan muncul **konfirmasi** untuk keluar dari aplikasi seperti pada Gambar 4.47, dan Gambar 4.48. Jika pengguna memilih **OK** aplikasi akan ditutup, sedangkan jika memilih **BATAL** ****pengguna akan tetap berada pada halaman **utama**.

Gambar 4.47 Logout pada perangkat Android Lenovo

Gambar 4.48 Logout pada perangkat Android Samsung

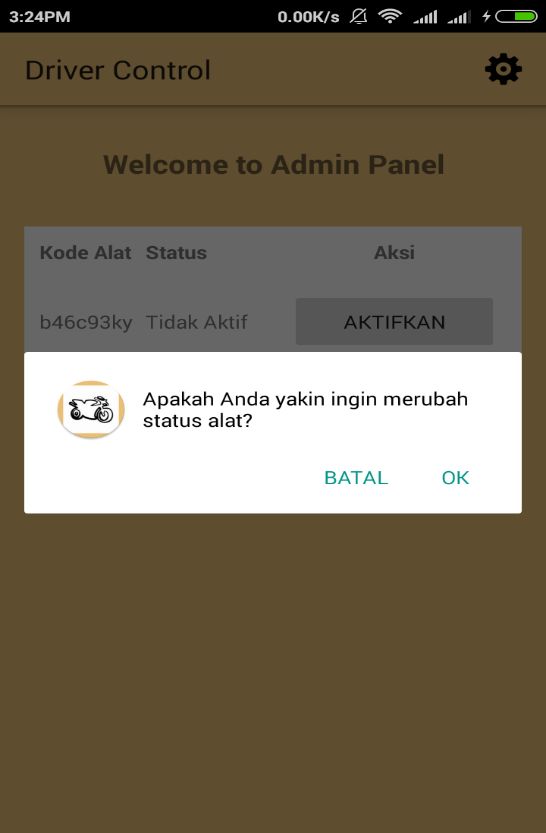
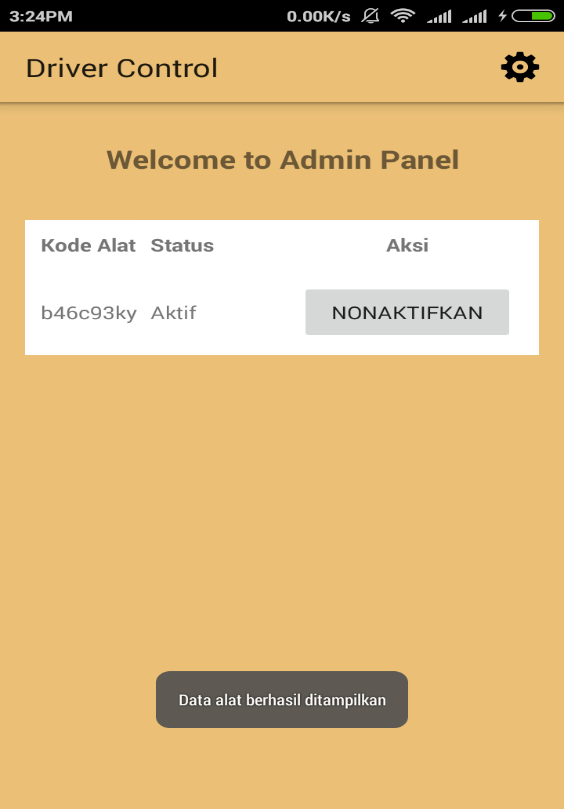
Aplikasi dapat berjalan dengan baik pada kedua perangkat Android dengan pengguna aplikasi yang berbeda, terlihat pada Gambar 4.47, dan Gambar 4.48 yang menunjukkan aplikasi dapat menampilkan **konfirmasi** ketika tombol **Logout** ditekan.

1. **Halaman Admin**

Pada aplikasi ini admin bertugas untuk menyaring data dengan cara mengaktifkan atau menonaktifkan alat. Pertama, admin harus melakukan **Login** terlebih dahulu agar dapat masuk halaman **Admin**. Kemudian, admin dapat melihat **kode alat** dan **status alat** untuk mengambil keputusan untuk mengaktifkan alat atau menonaktifkan alat menggunakan tombol **aksi** pada halaman **Admin**.

Gambar 4.49 Status alat tidak aktif

Gambar 4.49 menunjukkan aplikasi dapat berjalan dengan baik, admin dapat melihat **kode alat** dan **status alat** yang menunjukkan alat dalam kondisi tidak aktif. Jika admin ingin mengaktifkan alat, admin dapat menekan tombol **AKTIFKAN**.



Gambar 4.51 Status alat aktif

Gambar 4.50 Konfirmasi untuk mengaktifkan alat

Gambar 4.50 menunjukkan kondisi setelah admin menekan tombol **aksi AKTIFKAN**. Pada halaman **Admin** menunjukkan **konfirmasi** untuk merubah status alat. Gambar 4.51 menunjukkan kondisi setelah admin menekan tombol **OK** pada **konfirmasi** untuk merubah status alat, kemudian admin dapat melihat **kode alat** dan **status alat** yang menunjukkan alat dalam kondisi aktif. Jika admin ingin menonaktifkan alat, admin dapat menekan tombol **NONAKTIFKAN**.

1. **Analisis dan Pengujian**

Pengujian pada pembuatan aplikasi ini akan menggunakan pengujian kotak hitam. Pengujian ini dilakukan untuk menunjukkan fungsi program yang dibuat tentang cara operasi dan kegunaannya, apakah keluaran data sesuai dengan yang diharapkan. Pengujian aplikasi dibuat dalam bentuk tabel pengujian kotak hitam dari masing-masing menu yang ada dalam aplikasi. Pengujian yang akan dilakukan dengan metode pengujiankotak hitamyang berfokus pada persyaratan fungsional perangkat lunak.

1. **Pengujian Login**

Pengujian dimulai dengan pengujian menu **Login** yang muncul pertama kali ketika membuka aplikasi. Tabel 4.1 menunjukkan hasil pengujian **Login***.*

Tabel 4.1 Pengujian **Login**

|  |  |  |  |
| --- | --- | --- | --- |
| **Nama Pengujian** | **Bentuk Pengujian** | **Hasil yang Diharapkan** | **Hasil Pengujian** |
| Tampil halaman **Login** | Membuka aplikasi | Menampilkan halaman **Login** | Berhasil |
| Berhasil masuk | Mengklik tombol **Login** setelah mengisi nama pengguna dan kata sandi dengan benar | Masuk halaman **utama** aplikasi | Berhasil |

Tabel 4.1 Pengujian **Login** (lanjutan)

|  |  |  |  |
| --- | --- | --- | --- |
| Gagal masuk | Mengklik tombol **Login** setelahmengisi nama pengguna atau kata sandi salah atau sedang tidak terhubung jaringan | Menampilkan **pesan kesalahan** | Berhasil |
| Daftar akun baru | Mengklik tombol **Register** untuk menuju halaman pembuatan akun baru | Menampilkan halaman **Register** | Berhasil |

Hasil pengujian Tabel 4.1 menunjukkan bahwa fungsi yang ada pada tampilan sebagai pengguna berhasil dijalankan sesuai dengan harapan mulai dari menampilkan halaman **Login**, masuk ke halaman utama saat pengguna menekan tombol **Login**, menampilkan **pesan kesalahan** ketika pengguna salah memasukkan data, menampilkan halaman **Register** saat pengguna menekan tombol **Register**. Hal ini terjadi karena aplikasi mengirimkan data ke *Web Service* kemudian *server* mencocokkan data yang diisi pengguna dengan tabel user yang ada pada basisdata. Jika data pengguna tidak sama dengan data yang sudah terdaftar pada basisdata akan tampil **pesan kesalahan**. Sebelum melakukan **Login** pengguna harus mendaftar akun baru terlebih dahulu, setelah registrasi akun baru pada halaman **Register** sistem akan mengirimkan *email* untuk diverifikasi oleh pengguna. Ketika belum diverifikasi status pengguna masih bernilai 0 pada tabel user dalam basisdata, sehingga pengguna harus melakukan verifikasi *email* terlebih dahulu agar pengguna dapat masuk ke dalam halaman **utama** aplikasi. Setelah berhasil verifikasi, kolom status pada tabel user dalam basisdata akan bernilai 1 dan pengguna dapat masuk ke halaman **utama** aplikasi.

1. **Pengujian Register**

Pengujian ini dilakukan untuk membuat akun baru agar pengguna dapat melakukan **Login** untuk masuk ke dalam aplikasi. Tabel 4.2 menunjukkan hasil pengujian **Register.**

|  |  |  |  |
| --- | --- | --- | --- |
| **Nama Pengujian** | **Bentuk Pengujian** | **Hasil yang Diharapkan** | **Hasil Pengujian** |
| Daftar pengguna baru | Mengisi data pengguna | Data disimpan dalam basisdata | Berhasil |
| Berhasil daftar | Mengklik tombol **Register** setelah mengisi data nama, email, password, dan no. plat motor. | Mengirim *email* verifikasi agar pengguna dapat **Login** | Berhasil |
| Gagal daftar | Mengklik tombol **Register** setelahmengisi data nama, email, password, dan no. plat motor, tetapi terdapat data yang salah atau gagal terhubung ke jaringan. | Menampilkan **pesan kesalahan** | Berhasil |

Tabel 4.2 Pengujian **Register**

Hasil pengujian Tabel 4.2 menunjukkan bahwa fungsi yang ada pada tampilan sebagai pengguna berhasil dijalankan sesuai dengan harapan mulai dari mengisi data pengguna baru untuk disimpan ke basisdata, mengirim *email* verifikasi agar pengguna dapat melakukan **Login**, dan menampilkan **pesan kesalahan** jika terjadi kesalahan dalam pengisian data atau gagal terhubung ke jaringan. Pada proses pendaftaran akun baru aplikasi akan melakukan autentikasi, jika data sudah benar maka data akan disimpan pada tabel user dan tabel devices yang ada pada basisdata. Kemudian sistem akan mengirimkan *email* verifikasi agar pengguna dapat melakukan **Login**.

1. **Pengujian Halaman Utama**

Pengujian dimulai pada halaman **utama** yang muncul ketika pengguna sudah berhasil melewati proses **Login**. Tabel 4.3 menunjukkan hasil pengujian halaman **Utama.**

Tabel 4.3 Pengujian halaman **utama**

|  |  |  |  |
| --- | --- | --- | --- |
| **Nama Pengujian** | **Bentuk Pengujian** | **Hasil yang Diharapkan** | **Hasil Pengujian** |
| Tombol **Maps** | Mengklik tombol **Maps** | Menampilkan halaman **Map** , dan menunjukan lokasi motor | Berhasil |
| Tombol **Speed** | Mengklik tombol **Speed** | Menampilkan halaman **Kecepatan** | Berhasil |
| Tombol **Lock** | Mengklik tombol **Lock** | Menampilkan halaman **Lock** | Berhasil |
| Tombol **Profile** | Mengklik tombol **Profile** | Menampilkan halaman **Profile** | Berhasil |
| **Slidebar menu** | Mengklik icon **slidebar menu** | Menampilkan daftar menu **How to use, Data record, About,** dan **Logout** | Berhasil |

Hasil pengujian Tabel 4.3 menunjukkan bahwa fungsi yang ada pada tampilan sebagai pengguna berhasil dijalankan sesuai dengan harapan, mulai dari menampilkan halaman **Map** ketika tombol **Maps** ditekan, menampilkan halaman **Kecepatan** ketika tombol **Speed** ditekan, menampilkan halaman **Lock** ketika tombol **Lock** ditekan, menampilkan halaman **Profile** ketika tombol **Profile** ditekan, dan menampilkan daftar menu **How to use, Data record, About,** dan **Logout** ketika icon **slidebar menu** ditekan. Hal ini terjadi karena pemanggilan fungsi Intent pada mainactivity.java sudah dihubungkan sesuai dengan setiap activity.java yang dibuat.

1. **Pengujian Slidebar Menu**

Pengujian ini dilakukan ketika pengguna sudah membuka halaman **utama** aplikasi, pengujian ini dilakukan untuk memeriksa apakah menu yang ada pada **slidebar menu** berfungsi dengan baik atau tidak. Tabel 4.4 menunjukkan hasil pengujian **slidebar menu**.

|  |  |  |  |
| --- | --- | --- | --- |
| **Nama Pengujian** | **Bentuk Pengujian** | **Hasil yang Diharapkan** | **Hasil Pengujian** |
| Tombol **How to use** | Mengklik tombol **How to use** | Menampilkan halaman **Petunjuk** | Berhasil |
| Tombol **Data record** | Mengklik tombol **Data record** | Menampilkan halaman **Rekaman Data** | Berhasil |
| Tombol **About** | Mengklik tombol **About** | Menampilkan halaman **Tentang aplikasi** | Berhasil |
| Tombol **Logout** | Mengklik tombol **Logout** | Menampilkan **konfirmasi** untuk keluar dari Aplikasi | Berhasil |

Tabel 4.4 Pengujian Slidebar Menu

Hasil pengujian Tabel 4.4 menunjukkan bahwa fungsi yang ada pada tampilan sebagai pengguna berhasil dijalankan sesuai dengan harapan, mulai dari menampilkan halaman **Petunjuk**, menampilkan halaman **Rekaman Data**, menampilkan halaman **Tentang aplikasi**, dan menampilkan **konfirmasi** untuk keluar dari Aplikasi. Hal ini terjadi karena pemanggilan fungsi Intent pada mainactivity.java sudah dihubungkan sesuai dengan setiap activity.java yang dibuat.

1. **Pengujian Menu Lock**

Pengujian ini dilakukan ketika pengguna sudah membuka halaman **Lock**, pengujian ini dilakukan untuk memeriksa apakah pesan dan tombol konfirmasi pada halaman **Lock** berfungsi dengan baik atau tidak. Tabel 4.5 menunjukkan hasil pengujian menu **Lock** saat motor hidup, dan Tabel 4.6 menunjukkan hasil pengujian menu **Lock** saat motor mati.

Tabel 4.5 Pengujian menu **Lock** saat motor hidup

|  |  |  |  |
| --- | --- | --- | --- |
| **Nama Pengujian** | **Bentuk Pengujian** | **Hasil yang Diharapkan** | **Hasil Pengujian** |
| Status mesin hidup | Memasuki halaman **Lock** | Menampilkan status mesin motor hidup | Berhasil |
| Konfirmasi mematikan mesin | Memasuki halaman **Lock** saat mesin motor hidup | Menampilkan konfirmasi untuk mematikan mesin motor | Berhasil |

Tabel 4.5 Pengujian menu **Lock** saat motor hidup (lanjutan)

|  |  |  |  |
| --- | --- | --- | --- |
| Tombol **Ya** | Mengklik tombol **Ya** | Memberi perintah untuk mematikan mesin motor | Berhasil |
| Tombol **Tidak** | Mengklik tombol **Tidak** | Menampilkan status mesin motor hidup | Berhasil |

Hasil pengujian Tabel 4.5 menunjukkan fungsi **Lock** saat motor hidup yang ada pada tampilan sebagai pengguna berhasil dijalankan sesuai dengan harapan, mulai dari menampilkan **status mesin** motor hidup, **konfirmasi** untuk mematikan mesin motor, memberi perintah untuk mematikan mesin motor, dan membatalkan perintah untuk mematikan mesin motor. Saat mesin motor hidup, alat *Driver Control* mengirimkan data status relay 1 ke basisdata yang terdapat dalam tabel bacarelay, kemudian ketika pengguna membuka menu **Lock** pada aplikasi, akan ditampilkan **status mesin** motor hidup sekaligus **konfirmasi** untuk mematikan mesin motor. Jika pengguna memilih **Ya**, maka status relay pada tabel bacarelay yang ada pada basisdata akan berubah menjadi 0, kemudian *server* mengirim perintah untuk memutus relay yang ada pada alat *Driver Control* sehingga mesin motor mati. Proses pengiriman perintah untuk mematikan mesin membutuhkan waktu kurang lebih 54 detik. Ketika mesin motor mati, pengemudi tidak bisa menyalakan mesin motor kembali sebelum pengguna aplikasi menghidupkan mesin motor melalui aplikasi *Driver Control*.

Tabel 4.6 Pengujian menu **Lock** saat motor mati

|  |  |  |  |
| --- | --- | --- | --- |
| **Nama Pengujian** | **Bentuk Pengujian** | **Hasil yang Diharapkan** | **Hasil Pengujian** |
| Status mesin mati | Memasuki halaman **Lock** | Menampilkan status mesin motor mati | Berhasil |
| Konfirmasi menghidupkan mesin motor | Memasuki halaman **Lock** saat mesin motor mati | Menampilkan konfirmasi untuk menghidupkan mesin motor | Berhasil |
| Tombol **Ya** | Mengklik tombol **Ya** | Memberi perintah untuk menghidupkan mesin motor | Berhasil |
| Tombol **Tidak** | Mengklik tombol **Tidak** | Menampilkan status mesin motor mati | Berhasil |

Hasil pengujian Tabel 4.6 menunjukkan fungsi **Lock** saat motor mati yang ada pada tampilan sebagai pengguna berhasil dijalankan sesuai dengan harapan, mulai dari menampilkan **status mesin** motor mati, **konfirmasi** untuk menghidupkan mesin motor, memberi perintah untuk menghidupkan mesin motor, dan membatalkan perintah untuk menghidupkan mesin motor. Saat mesin motor mati, alat *Driver Control* mengirimkan data status relay 0 ke basisdata yang terdapat dalam tabel bacarelay, kemudian ketika pengguna membuka menu **Lock** pada aplikasi, akan ditampilkan **status mesin** motor mati sekaligus **konfirmasi** untuk menghidupkan mesin motor. Jika pengguna memilih **Ya**, maka status relay pada tabel bacarelay yang ada pada basisdata akan berubah menjadi 1, kemudian *server* mengirim perintah untuk menyambung relay yang ada pada alat *Driver Control* dengan aki sehingga mesin motor hidup. Proses pengiriman perintah untuk menghidupkan mesin motor membutuhkan waktu kurang lebih 54 detik.

1. **Pengujian Menu Profile**

Pengujian ini dilakukan ketika pengguna sudah membuka halaman **Profile**, pengujian ini dilakukan untuk memeriksa apakah fungsi ubah profil berjalan dengan baik atau tidak. Tabel 4.7 menunjukkan hasil pengujian menu **Profile**.

Tabel 4.7 Pengujian menu **Profile**

|  |  |  |  |
| --- | --- | --- | --- |
| **Nama Pengujian** | **Bentuk Pengujian** | **Hasil yang Diharapkan** | **Hasil Pengujian** |
| Halaman **Ubah Profil** | Mengklik tombol **Profile** pada halaman utama | Menampilkan halaman **Ubah Profil** | Berhasil |
| Mengubah data profil | Mengubah *username*, *email*, atau kode user | Data berhasil diubah dan disimpan ke dalam basisdata | Berhasil |
| Tombol **Ubah Profil** | Mengklik tombol **Ubah Profil** | Menampilkan konfirmasi untuk ubah profil | Berhasil |
| Tombol **OK** | Mengklik tombol **OK** | Menyimpan perubahan data, dan menampilkan **pesan berhasil** | Berhasil |

Tabel 4.7 Pengujian menu **Profile** (lanjutan)

|  |  |  |  |
| --- | --- | --- | --- |
| Tombol **BATAL** | Mengklik tombol **BATAL** | Tetap berada pada halaman **Profil** | Berhasil |

Hasil pengujian Tabel 4.7 menunjukkan bahwa fungsi yang ada pada tampilan sebagai pengguna berhasil dijalankan sesuai dengan harapan, mulai dari menampilkan halaman **Ubah Profil**, mengubah data profil, menampilkan konfirmasi ubah profil, menyimpan perubahan data profil, dan membatalkan perintah ubah profil. Menu **Profile** berkaitan langsung dengan tabel user yang ada pada basisdata, ketika pengguna menekan tombol **Profile** pada halaman **utama**, aplikasi akan meminta data ke s*erver*, kemudian *server* akan mengirimkan data pengguna dari tabel user yang ada pada basisdata untuk ditampilkan pada aplikasi. Setelah pengguna mengubah data, aplikasi akan mengirimkan perubahan data ke *server* kemudian disimpan ke dalam basisdata.

1. **Pengujian UBAH PASSWORD**

Pengujian ini dilakukan ketika pengguna sudah membuka halaman **UBAH PASSWORD**, pengujian ini dilakukan untuk memeriksa apakah fungsi tombol **UBAH PASSWORD** berjalan dengan baik atau tidak. Tabel 4.8 menunjukkan hasil pengujian **UBAH PASSWORD.**

Tabel 4.8 Pengujian **UBAH PASSWORD**

|  |  |  |  |
| --- | --- | --- | --- |
| **Nama Pengujian** | **Bentuk Pengujian** | **Hasil yang Diharapkan** | **Hasil Pengujian** |
| Tombol **Ubah Password** | Mengklik tombol **UBAH PASSWORD** | Menampilkan halaman **Ubah Password** | Berhasil |
| Mengubah kata sandi | Memasukkan kata sandi lama, kata sandi baru, dan ulangi kata sandi baru | Mencocokkan kata sandi | Berhasil |

Tabel 4.8 Pengujian **UBAH PASSWORD** (lanjutan)

|  |  |  |  |
| --- | --- | --- | --- |
| Tombol **UBAH PASSWORD** | Mengklik tombol **UBAH PASSWORD** | Menampilkan konfirmasi ubah kata sandi | Berhasil |
| Tombol **OK** | Mengklik tombol **OK** | Menyimpan perubahan kata sandi, dan menuju halaman **Login** | Berhasil |
| Tombol **BATAL** | Mengklik tombol **BATAL** | Tidak ada perubahan kata sandi | Berhasil |

Hasil pengujian Tabel 4.8 menunjukkan bahwa fungsi yang ada pada tampilan sebagai pengguna berhasil dijalankan sesuai dengan harapan, mulai dari menampilkan halaman **Ubah Password**, mengubah kata sandi, menampilkan konfirmasi ubah kata sandi, menyimpan perubahan kata sandi, membuat pengguna menuju halaman **Login** ketika sudah mengubah kata sandi, dan membatalkan perintah ubah kata sandi. Menu **Ubah Password** berkaitan langsung dengan tabel user yang ada pada basisdata, ketika pengguna menekan tombol **Ubah Password** pada halaman **Profile**, aplikasi akan menampilkan kolom kata sandi dalam kondisi kosong sehingga harus diisi oleh pengguna. Setelah pengguna mengisi kata sandi lama dan kata sandi baru, aplikasi akan mengirimkan perubahan data ke *server* kemudian disimpan ke dalam basisdata.

1. **Pengujian Menu Data record**

Pengujian ini dilakukan ketika pengguna sudah membuka halaman **Rekaman data**, pengujian ini dilakukan untuk memeriksa apakah fungsi tombol **Cek** berjalan dengan baik atau tidak. Tabel 4.9 menunjukkan hasil pengujian menu **Data record**.

Tabel 4.9 Pengujian menu **Data record**

|  |  |  |  |
| --- | --- | --- | --- |
| **Nama Pengujian** | **Bentuk Pengujian** | **Hasil yang Diharapkan** | **Hasil Pengujian** |
| Tombol **Hari** | Memilih hari | Menampilkan rekaman data | Berhasil |
| Tombol **Cek** | Mengklik tombol **Cek** | Menampilkan halaman **Map** | Berhasil |

Hasil pengujian Tabel 4.9 menunjukkan bahwa fungsi yang ada pada tampilan sebagai pengguna berhasil dijalankan sesuai dengan harapan, mulai dari menampilkan rekaman data pengemudi, dan menampilkan halaman **Map** yang menunjukkan lokasi motor ketika tombol **Cek** ditekan. Ketika pengguna menekan menu **Data record**, aplikasi akan menampilkan halaman **Rekaman data** dengan data terbaru. Pada halaman **Rekaman data** terdapat pilihan waktu selama satu minggu terakhir, Jika pengguna ingin melihat rekaman data pengemudi pada hari sebelumnya, pengguna harus memilih hari, maka aplikasi akan meminta data ke *server*. *Server* mengirimkan rekaman data berupa latitude, longitude, kecepatan, dan waktu dari tabel ditrol pada basisdata. Jika pengguna ingin melihat lokasi motor, pengguna dapat menekan tombol **Cek**, aplikasi akan meminta layanan Google Map untuk menampilkan lokasi motor pada halaman **Map**.

1. **Pengujian Notifikasi Motor Jatuh**

Pengujian ini dilakukan ketika pengguna sudah mengaktifkan Internet, dan memasang aplikasi pada perangkat Android. Pengujian ini dilakukan untuk memeriksa apakah fungsi **Notifikasi Motor Jatuh** berjalan dengan baik atau tidak. Tabel 4.10 menunjukkan hasil pengujian **Notifikasi Motor Jatuh**.

Tabel 4.10 Pengujian Notifikasi Motor Jatuh

|  |  |  |  |
| --- | --- | --- | --- |
| **Nama Pengujian** | **Bentuk Pengujian** | **Hasil yang Diharapkan** | **Hasil Pengujian** |
| Notifikasi Motor Jatuh | Mengguncangkan alat ***Driver Control*** | Menampilkan **notifikasi motor jatuh** pada perangkat Android | Berhasil |

Hasil pengujian Tabel 4.10 menunjukkan bahwa fungsi yang ada pada tampilan sebagai pengguna berhasil dijalankan sesuai dengan harapan, ketika alat *Driver Control*  diguncangkan, alat akan mengirim nilai jatuh 1 pada tabel ditrol yang ada pada basisdata. Ketika nilai jatuh sama dengan 1 sistem akan mengirimkan notifikasi motor jatuh pada perangkat Android yang terdaftar pada tabel devices yang ada dalam basisdata. Sedangkan ketika nilai jatuh 0 pada tabel ditrol aplikasi tidak mengirim notifikasi karena motor terdeteksi dalam kondisi normal.

1. **Pengujian Speed**

Pengujian ini dilakukan ketika pengguna ingin melihat kecepatan rata-rata dari pengemudi, hasil pengujian **Speed** ditunjukkan pada Tabel 4.11.

Tabel 4.11 Pengujian **Speed**

|  |  |  |  |
| --- | --- | --- | --- |
| **Nama Pengujian** | **Bentuk Pengujian** | **Hasil yang Diharapkan** | **Hasil Pengujian** |
| Pilih tanggal | Menekan tombol **V** | Menampilkan tanggal dalam seminggu | Berhasil |
| Tombol **UPDATE** | Mengklik tombol **UPDATE** | Memperbarui data kecepatan pada grafik | Berhasil |

Hasil pengujian Tabel 4.11 menunjukkan bahwa menu **Speed** berhasil dijalankan sesuai dengan harapan mulai dari menampilkan tanggal dalam seminggu, dan memperbarui data kecepatan pada grafik. Data tanggal dalam seminggu dapat ditampilkan karena menggunakan fungsi spinner pada Android Studio, dan proses saat memperbarui grafik kecepatan menggunakan fungsi reset array dan updateSpeedTask() pada Android Studio.

1. **Pengujian Logout**

Pengujian ini dilakukan ketika pengguna ingin keluar dari aplikasi, pengujian **Logout** ditunjukkan pada Tabel 4.12.

Tabel 4.12 Pengujian **Logout**

|  |  |  |  |
| --- | --- | --- | --- |
| **Nama Pengujian** | **Bentuk Pengujian** | **Hasil yang Diharapkan** | **Hasil Pengujian** |
| Logout | Menekan tombol **Logout** | Menampilkan konfirmasi | Berhasil |
| Tombol **OK** | Mengklik tombol **OK** | Keluar dari aplikasi | Berhasil |
| Tombol **BATAL** | Mengklik tombol **OK** | Pengguna tetap berada dalam aplikasi | Berhasil |

Hasil pengujian Tabel 4.12 menunjukkan bahwa fungsi yang ada pada tampilan sebagai pengguna berhasil dijalankan sesuai dengan harapan mulai dari menampilkan konfirmasi untuk keluar dari aplikasi, mengeluarkan pengguna dari aplikasi, dan membatalkan perintah untuk keluar dari aplikasi. Fungsi menu ini bertujuan agar sistem tidak disalahgunakan oleh orang yang tidak bertanggung jawab.

1. **Pengujian Admin**

Pada pengujian ini Admin melakukan **Login** terlebih dahulu kemudian mengecek **kode alat** dan **status alat**, setelah itu mengambil keputusan untuk mengaktifkan alat atau menonaktifkan alat. Hasil pengujian admin ditunjukkan pada Tabel 4.13.

Tabel 4.13 Pengujian **Admin**

|  |  |  |  |
| --- | --- | --- | --- |
| **Nama Pengujian** | **Bentuk Pengujian** | **Hasil yang Diharapkan** | **Hasil Pengujian** |
| Login | Menekan tombol **Login** | Menampilkan halaman **Admin** yang berisi kode alat dan status alat | Berhasil |
| Mengaktifkan alat | Menekan tombol **AKTIFKAN** | Menampilkan konfirmasi untuk mengaktifkan alat | Berhasil |
| Menonaktifkan alat | Menekan tombol **NONAKTIFKAN** | Menampilkan konfirmasi untuk menonaktifkan alat | Berhasil |
| Tombol **OK** konfirmasi | Menekan Tombol **OK** pada konfirmasi | Merubah status alat | Berhasil |
| Tombol **Batal** konfirmasi | Menekan Tombol **Batal** pada konfirmasi | Membatalkan perintah merubah status alat | Berhasil |

Hasil pengujian Tabel 4.13 menunjukkan bahwa fungsi yang ada pada tampilan sebagai admin berhasil dijalankan sesuai dengan harapan mulai dari **Login** admin, menampilkan halaman **Admin**, mengaktifkan alat, dan menonaktifkan alat. Fungsi menu ini bertujuan agar sistem tidak disalahgunakan oleh orang yang tidak bertanggung jawab dengan cara menyaring data yang dikirim oleh alat.



# BAB V

# PENUTUP

1. **Kesimpulan**

Berdasarkan hasil pengujian dan analisis pembuatan aplikasi dapat disimpulkan sebagai berikut :

1. Aplikasi yang dibuat pada Tugas Akhir ini dapat menampilkan peta lokasi kendaraan motor, grafik kecepatan pengemudi, profil pengguna aplikasi, rekaman data pengemudi, petunjuk pemakaian aplikasi, tentang pengembang aplikasi, notifikasi ketika motor jatuh, dan pengguna dapat mematikan atau menghidupkan mesin motor secara otomatis melalui aplikasi ini.
2. Aplikasi ini dapat digunakan oleh beberapa pengguna untuk memantau sebuah sepeda motor yang sudah terpasang alat *Driver Control*.
3. Pengguna membutuhkan koneksi Internet, dan perangkat keras *Driver Control* untuk bisa mengakses menu aplikasi.
4. **Saran**

Berdasarkan pengujian dan analisis terhadap aplikasi Android *Driver Control* yang telah dibuat, dapat diberikan beberapa saran sebagai berikut.

1. Pengembangan desain tampilan antarmuka halaman aplikasi agar lebih menarik.
2. Perlu dilakukan pengecekan pulsa pada alat *Driver Control* agar dapat mencegah kegagalan saat dalam melakukan perintah.
3. Penambahan fitur telepon sebagai notifikasi ketika motor jatuh, saat kecepatan motor di atas rata-rata, fitur lupa *password* saat login, dan fitur tambah kendaraan agar bisa memonitor lebih dari satu motor.
4. Pengembangan aplikasi agar dapat dijalankan pada berbagai perangkat telepon cerdas seperti Windows Phone, Blackberry, dan iOS.

# DAFTAR PUSTAKA

1. “Global status report on road safety 2015”. [Online]. Available: http://www.who.int/violence\_injury\_prevention/roar\_safety\_status/2015/en/. [Accessed: 03-June-2016].
2. Azmi, N.A. “Modus Operandi Kejahatan Pencurian Kendaraan Bermotor Roda Dua”. Jurnal Fakultas Hukum Universitas Brawijaya, 2014.
3. Hanifah, Raidah., “Simulasi Sistem Informasi Geografis (SIG) Pemantauan Posisi Kendaraan Via SMS Gateway”*,* Skripsi S-1, Teknik Elektro, Universitas Diponegoro, 2010.
4. Rusnandar, dkk. “Sistem Pelacak Kendaraan Berbasis Open GTS”. Jurnal Fakultas Teknologi Industri Universitas Ahmad Dahlan Yogyakarta, 2013.
5. Pria Utama, Hendra., “Pembuatan Aplikasi Memantau Lokasi Anak Berbasis Android Menggunakan Location Based Service*”,* Skripsi S-1, Program Studi Sistem Komputer, Universitas Diponegoro, 2015.
6. A. A. Huda, “Live Coding : 9 Aplikasi Android Buatan Sendiri”, Yogyakarta: Andi, 2013.
7. Nugroho, Adi. “Ebook: Rekayasa Perangkat Lunak Menggunakan UML dan Java.”Andi Offset. 2009.
8. Rompas, B.R “Aplikasi Location Based Service Pencarian Tempat Di Kota Manado Berbasis Android”, Jurnal Teknik Elektro dan Komputer Universitas Sam Ratulangi Vo. 1, 2012.
9. Solichin, A., “MySQL 5: Dari Pemula Hingga Mahir*”.* Jakarta: Univ. Budi Luhur, 2010.
10. Rosa, A.S., dan Shalahuddin, “Rekayasa Perangkat Lunak”, Bandung: Modula, 2011.
11. “Penerapan Metodologi Waterfall dalam Pengembangan Sistem Informasi Akademik Universitas Negeri Surabaya”*.* [Online]. Available: http://jenengku-dayat.mhs.narotama.ac.id/files/2013/10/makalah-analisis-SSI.pdf. [Accessed: 03-March-2016].
12. Deviana, H., “Penerapan XML Web service Pada Sistem Distribusi Barang,” *J. Geneic*, vol. 6, no. 2, 2011.
13. Surendra, M. R. S., “Implementasi PHP Web Service Sebagai Penyedia Data Aplikasi Mobile,” vol. VI, no. 2, pp. 85–93, 2014.
14. Rouf, A., “Pengujian Perangkat Lunak Dengan Menggunakan Metode White Box dan Black Box,” STMIK HIMSYA, Semarang, 2010.

# LAMPIRAN I BIODATA MAHASISWA

****

Nama : Lia Lidya Roza

NIM : 21120112120011

Tempat, Tanggal Lahir : Bekasi, 5 Agustus 1994

Alamat domisili : Jl. Tahir No.12 Bekasi Barat, Bekasi

Alamat sekarang : Jl. Tembalang Selatan VIII No.8, Semarang

Nomor Handphone : 089638706850

Moto Hidup : Berkarya Prestasi, Berguna Selagi Muda & Bisa

Alamat email : lialidya@ce.undip.ac.id / lialidyaroza@gmail.com

Agama : Kristen Protestan

Orang Tua : Sahata Simatupang, dan Ana Siregar

IP Kumulatif : 3,62

Tanggal Lulus :

Masa studi : 4 tahun, bulan

Pengalaman kerja : Call Center SNMPTN-SBMPTN (2014)

Asisten Praktikum Laboratorium Sistem Embedded & Robotika (2013 s/d 2016)

Asisten Praktikum Laboratorium Mobile Software (2015 s/d 2016)

**Aktivitas Keorganisasian yang Pernah / Sedang Diikuti**

|  |  |  |  |
| --- | --- | --- | --- |
| **Nama Organisasi** | **Jabatan** | **Tingkat** | **Periode** |
| Palang Merah Remaja SMPN 14 Bekasi | Ketua Perawatan Keluarga & Humas | SMP | 2006-2009 |
| Karya Ilmiah Remaja SMAN 3 Bekasi | Humas | SMA | 2009-2012 |
| HIMASKOM UNDIP | Sekretaris Departemen Infokom | Jurusan | 2014 |
| PMK-FT UNDIP | Anggota Divisi Nafiri | Fakultas | 2014 |

**Prestasi yang Pernah Diraih (Akademis/ Non Akademis)**

|  |  |  |
| --- | --- | --- |
| **Prestasi** | **Tingkat** | **Periode** |
| Juara Harapan 3 Siswa Berprestasi | Kota Bekasi | 2005 |
| Juara 1 Pertolongan Pertama Palang Merah Remaja | Kota Bekasi | 2007 |
| Juara 2 Perawatan Keluarga Palang Merah Remaja | Kota Bekasi | 2008 |
| Juara Harapan 1 Siswa Berprestasi OSN | Kota Bekasi | 2009 |
| Finalis Olimpiade Sains Nasional | Provinsi Jawa Barat | 2009 |
| Peserta terbaik 3 LKMM Pra Dasar Sistem Komputer | Jurusan | 2012 |
| Finalis Gemastik 8 | Nasional | 2015 |

**LATAR BELAKANG PENDIDIKAN**

|  |  |  |  |
| --- | --- | --- | --- |
| Universitas/Sekolah | Fakultas | Jurusan | Tahun |
| Universitas Diponegoro | Teknik | Sistem Komputer | 2012- 2016 |
| SMAN 3 Bekasi | - | IPA | 2009 - 2012 |
| SMPN 14 Bekasi | - | - | 2006- 2009 |
| SDN Kranji XI | - | - | 2000- 2006 |

Semarang, 9 Oktober 2016

Lia Lidya Roza