

RANCANG BANGUN APLIKASI ENTERTAINMENT ACCESS TERMINAL PADA TABLET ANDROID

SKRIPSI

Untuk memenuhi sebagian persyaratan
mencapai gelar Sarjana Komputer

Disusun Oleh:

Zidna Ziada El Karim

NIM: 115060800111035



PROGRAM STUDI INFORMATIKA/ILMU KOMPUTER
PROGRAM TEKNOLOGI INFORMASI DAN ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA

MALANG

2016



PENGESAHAN

RANCANG BANGUN APLIKASI *ENTERTAINMENT ACCESS TERMINAL*
PADA TABLET ANDROID

SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan
mencapai gelar Sarjana Komputer

Disusun Oleh:

Zidna Ziada El Karim

NIM. 115060800111035

Skripsi ini telah diuji dan dinyatakan lulus pada
19 Januari 2016

Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I

Dosen Pembimbing II

Dr.Eng. Herman Tolle, ST., MT.
NIP. 19740823 200012 1 001

Aryo Pinandito, S.T., M.MT.
NIP. 19830519 201404 1 001

Mengetahui
Ketua Program Studi Informatika/Ilmu Komputer

Drs. Marji, M.T
NIP. 19670801 199203 1 001

PERNYATAAN ORISINALITAS

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah skripsi ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis disitasi dalam naskah ini dan disebutkan dalam daftar pustaka.

Apabila ternyata didalam naskah skripsi ini dapat dibuktikan terdapat unsur-unsur plagiasi, saya bersedia skripsi ini digugurkan dan gelar akademik yang telah saya peroleh (sarjana) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).

Malang, 20 Januari 2016



Zidna Ziada El Karim
NIM: 115060800111035

KATA PENGANTAR

Puji dan syukur kehadiran Allah SWT yang telah melimpahkan rahmat dan karunia-Nya sehingga penulis dapat menyelesaikan skripsi dengan judul “Rancang Bangun Aplikasi *Entertainment Access Terminal* Pada Tablet Android” dengan baik. Melalui kesempatan ini, penulis ingin menyampaikan rasa hormat dan terima kasih kepada semua pihak yang telah memberikan bantuan dan dukungan selama pengerjaan skripsi, diantaranya:

1. Bapak Dr. Eng. Herman Tolle, S.T., M.T. dan Bapak Aryo Pinandito, S.T., M.MT. selaku dosen pembimbing skripsi yang telah membimbing dan mengarahkan penulis sehingga dapat menyelesaikan skripsi ini.
2. Orang tua dan keluarga penulis yang telah memberi doa, motivasi, kasih sayang serta dukungan baik moril maupun materiil.
3. Segenap bapak dan ibu dosen program studi Informatika / Ilmu Komputer beserta seluruh staff administrasi yang telah membantu selama kegiatan perkuliahan.
4. Seluruh Civitas Akademika Informatika Universitas Brawijaya yang telah banyak memberi bantuan dan dukungan selama penulis menempuh studi di Prodi Informatika Universitas Brawijaya dan selama penyelesaian skripsi ini.
5. Ahmad Dzulfikar SR dan Irfandi Ahmad, yang telah banyak memberikan dukungan dan saran selama pengerjaan skripsi ini.
6. Semua teman-teman FILKOM, khususnya Informatika angkatan 2011 atas segala bantuan dan dukungannya selama ini
7. Semua pihak yang tidak dapat penulis sebutkan satu per satu yang terlibat baik secara langsung maupun tidak langsung demi terselesaikannya skripsi ini.

Penulis menyadari sepenuhnya bahwa skripsi ini masih belum sempurna karena keterbatasan materi dan pengetahuan yang dimiliki penulis. Penulis mengharapkan kritik dan saran dari berbagai pihak demi penyempurnaan tugas akhir ini. Akhir kata penulis berharap skripsi ini dapat membawa manfaat bagi semua pihak yang menggunakannya.

Malang, 10 Desember 2015

Penulis
zidnaziada@gmail.com

ABSTRAK

Zidna Ziada El Karim. 2016. Rancang Bangun Aplikasi *Entertainment access Terminal* Pada Tablet Android. Program Teknologi Informasi dan Ilmu Komputer, Universitas Brawijaya, Malang. Dosen Pembimbing: Dr.Eng. Herman Tolle, ST., MT. dan Aryo Pinandito, S.T., M.MT.

Sistem hiburan pada alat transportasi umum dibutuhkan untuk mengatasi rasa bosan ketika berada dalam perjalanan. Sistem hiburan yang ada saat ini cenderung bersifat kolektif sehingga tidak semua penumpang terhibur dengan apa yang tersaji. Melihat dari keberagaman penumpang pada alat transportasi umum, diperlukan sistem hiburan yang mana penumpang dapat memilih hiburan sesuai dengan keinginan masing-masing penumpang. Sistem hiburan pada alat transportasi umum dapat direalisasikan menggunakan teknologi perangkat bergerak berbentuk tablet dan sebuah *server*. Penggunaan perangkat tablet dikarenakan tablet didukung dengan layar yang lebar sehingga sangat cocok untuk dijadikan media hiburan. Tablet juga didukung dengan Wi-Fi sehingga dapat menghubungkan perangkat dengan *server*. Penelitian ini bertujuan untuk merancang dan mengimplementasikan aplikasi yang dapat berjalan pada perangkat tablet yang dapat digunakan untuk menikmati hiburan. Dari hasil penelitian yang dilakukan, Aplikasi *entertainment access terminal* yang dibangun berhasil diimplementasikan pada tablet Android dan telah sesuai dengan perancangan dan kebutuhan pengguna. Hasil pengujian fungsionalitas menunjukkan sistem telah sesuai dengan spesifikasi kebutuhan fungsional. Hasil pengujian usabilitas menunjukkan menunjukkan tingkat kepuasan pengguna terhadap sistem mencapai skor 73.33 yang berarti tingkat usabilitas sistem termasuk dalam klasifikasi di atas rata-rata. Hasil pengujian reliabilitas menunjukkan sistem dapat menangani: *request web service* mencapai 1000 user dengan persentase keberhasilan 99.98%, *request konten film* mencapai 50 user dengan persentase keberhasilan 60%, *request konten musik* mencapai 100 user dengan persentase keberhasilan 96%, *request konten buku* mencapai 500 user dengan persentase keberhasilan 99.4%, dan *request konten majalah* mencapai 100 user dengan persentase keberhasilan 82%.

Kata kunci: Android, Hiburan, *Server*, Tablet, Transportasi

ABSTRACT

Zidna Ziada El Karim. 2016. Rancang Bangun Aplikasi *Entertainment Access Terminal* Pada Tablet Android. Information Technology and Computer Science Program, Brawijaya University. Advisor: Dr.Eng. Herman Tolle, ST., MT. dan Aryo Pinandito, S.T., M.MT.

Entertainment system is needed to reducing psycological stress while at the public transportation. Public transportation nowadays tend to be more focussed on collective rather than individual amusement. Therefore not all passenger can enjoy it. Considering the diversity of the passengers, entertainments that can be used individually is needed so every passenger could enjoy it. Entertainment system can be implemented using mobile technology in the form of a tablet and a server. The use of tablet devices due to its wide a screen that suitable to be used as a medium of entertainment. Tablets are also supported with Wi-Fi so it can be connected to the server. This study aims to design and build applications which can run on a tablet device that can be used to enjoy entertainment. results of research conducted, entertainment access terminal application has been built and implemented successfully on Android tablets and in accordance with the design and user requirements. system functionality test indicate that system in accordance with specification of functional requirements. Usability test result shows level of user satisfaction reached score 73.33 that means system usability level reach above average. Reliability test result shows that system can handle web service request up to 1000 concurrent users with success percentage reaches 99.98%, movie request up to 50 concurrent users with success percentage reaches 60%, music request up to 100 concurrent users with success percentage reaches 96%, book request up to 500 concurrent users with success percentage reaches 99.4%, and magazine request up to 100 concurrent users with success percentage reaches 82%.

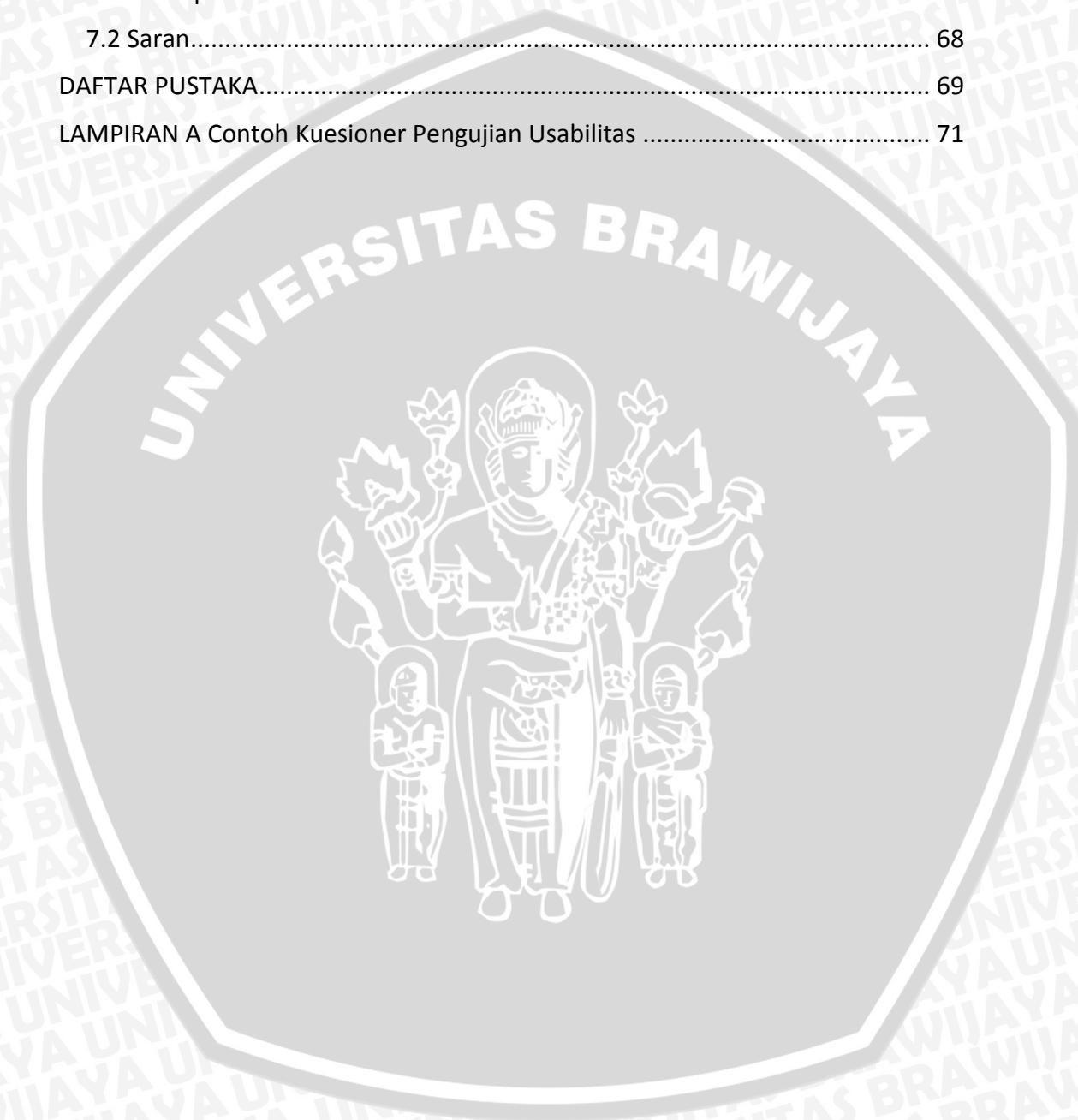
Keyword: Android, Entertainment, Server, Tablet, Transportation,

DAFTAR ISI

PENGESAHAN	ii
PERNYATAAN ORISINALITAS	iii
KATA PENGANTAR.....	iv
ABSTRAK.....	v
ABSTRACT	vi
DAFTAR ISI.....	vii
DAFTAR TABEL.....	x
DAFTAR GAMBAR.....	xi
DAFTAR KODE.....	xii
DAFTAR LAMPIRAN	xiii
BAB 1 PENDAHULUAN.....	1
1.1 Latar belakang.....	1
1.2 Rumusan masalah	2
1.3 Batasan masalah	2
1.4 Tujuan.....	2
1.5 Manfaat	3
1.6 Sistematika penulisan	3
BAB 2 LANDASAN KEPUSTAKAAN	4
2.1 Android.....	4
2.2 Web service	9
2.3 REST.....	11
2.4 JSON	13
2.5 System Usability Scale	16
BAB 3 METODOLOGI	18
3.1 Studi Literatur	19
3.2 Analisis Kebutuhan.....	19
3.3 Perancangan.....	19
3.4 Implementasi.....	20
3.5 Pengujian.....	20
3.6 Pengambilan Kesimpulan dan Saran.....	21

BAB 4 ANALISIS DAN PERANCANGAN	22
4.1 Analisis Kebutuhan	22
4.1.1 Gambaran Umum Aplikasi	22
4.1.2 Identifikasi Aktor	23
4.1.3 Analisis Kebutuhan Fungsional dan Non-Fungsional	23
4.1.4 Diagram <i>Use Case</i>	24
4.1.5 Skenario <i>Use Case</i>	25
4.2 Perancangan	27
4.2.1 Perancangan Arsitektural	27
4.2.2 Perancangan Activity Diagram	27
4.2.3 Perancangan Sequence Diagram	31
4.2.4 Perancangan Class Diagram	33
4.2.5 Perancangan <i>Page Flow</i>	35
4.2.6 Perancangan <i>User Experience</i>	36
4.2.7 Multimedia Content Format	39
BAB 5 IMPLEMENTASI	40
5.1 Spesifikasi Sistem	40
5.1.1 Spesifikasi Perangkat Keras	40
5.1.2 Spesifikasi Perangkat Lunak	41
5.2 Batasan Implementasi	41
5.3 Implementasi Basis Data	42
5.4 Implementasi Kode Program	44
5.4.1 Kode Program Mengelola Konten Pada <i>Server</i>	44
5.4.2 Kode Program Melihat Daftar Konten Pada Klien	48
5.4.3 Kode Program Membuka Konten Pada Klien	50
5.5 Implementasi Antarmuka	51
5.5.1 Implementasi Antarmuka Halaman Home	51
5.5.2 Implementasi Antarmuka Halaman Daftar Konten	54
5.5.3 Implementasi Antarmuka Halaman Detail Konten	55
BAB 6 PENGUJIAN	60
6.1 Pengujian Usabilitas	60

6.2 Pengujian Reliabilitas	61
6.3 Pengujian Fungsionalitas.....	64
BAB 7 PENUTUP	68
7.1 Kesimpulan	68
7.2 Saran.....	68
DAFTAR PUSTAKA.....	69
LAMPIRAN A Contoh Kuesioner Pengujian Usabilitas	71



DAFTAR TABEL

Tabel 2.1 Format dan codec audio dan video yang didukung oleh Android	5
Tabel 2.2 Perbedaan antara POST dan PUT	12
Tabel 4.1 Identifikasi aktor	23
Tabel 4.2 Daftar kebutuhan fungsional	23
Tabel 4.3 Daftar kebutuhan non-fungsional	24
Tabel 4.4 Skenario <i>use case</i> melihat list	25
Tabel 4.5 Skenario <i>use case</i> membuka konten	25
Tabel 4.6 Skenario <i>use case</i> mengelola data konten	26
Tabel 4.7 Format konten multimedia	39
Tabel 5.1 Spesifikasi perangkat keras <i>server</i>	41
Tabel 5.2 Spesifikasi perangkat keras klien	41
Tabel 5.3 Spesifikasi perangkat lunak <i>server</i>	41
Tabel 5.4 Spesifikasi perangkat lunak klien	41
Tabel 5.5 Contoh nilai tiap atribut pada tabel content	43
Tabel 5.6 Deskripsi tabel movie	43
Tabel 5.7 Contoh nilai tiap atribut pada tabel music	43
Tabel 5.8 Deskripsi tabel book	44
Tabel 6.1 Hasil pengujian usabilitas	60
Tabel 6.2 Hasil pengujian reliabilitas web service	62
Tabel 6.3 Hasil pengujian reliabilitas konten film	63
Tabel 6.4 Hasil pengujian reliabilitas konten musik	63
Tabel 6.5 Hasil pengujian reliabilitas konten buku	63
Tabel 6.6 Hasil pengujian reliabilitas konten majalah	63
Tabel 6.7 Kasus uji fungsional Melihat list	64
Tabel 6.8 Kasus uji fungsional membuka konten berjenis musik	64
Tabel 6.9 Kasus uji fungsional membuka konten berjenis film	65
Tabel 6.10 Kasus uji fungsional membuka konten berjenis buku	65
Tabel 6.11 Kasus uji fungsional membuka konten berjenis majalah	65
Tabel 6.12 Kasus uji fungsional mengelola data konten	66
Tabel 6.13 Hasil pengujian fungsionalitas	66

DAFTAR GAMBAR

Gambar 2.1 Arsitektur <i>Web service</i>	10
Gambar 2.2 Objek pada JSON	14
Gambar 2.3 Array pada JSON	14
Gambar 2.4 <i>Value</i> pada JSON	15
Gambar 2.5 String pada JSON	15
Gambar 2.6 Objek pada JSON	16
Gambar 2.7 Format respon kuesioner SUS	17
Gambar 3.1 Diagram Alir Tahapan Penelitian	18
Gambar 4.1 Ilustrasi gambaran umum aplikasi	22
Gambar 4.2 Diagram use case aplikasi entertainment access terminal	24
Gambar 4.3 Rancangan arsitektural sistem	27
Gambar 4.4 <i>Activity diagram</i> melihat <i>list</i>	28
Gambar 4.5 <i>Activity diagram</i> membuka konten	29
Gambar 4.6 <i>Activity diagram</i> mengelola data konten	30
Gambar 4.7 <i>Sequence diagram</i> melihat list	31
Gambar 4.8 <i>Sequence diagram</i> mengelola data konten	32
Gambar 4.9 <i>Sequence diagram</i> membuka konten	33
Gambar 4.10 <i>Class diagram</i> aplikasi klien	34
Gambar 4.11 <i>Class Diagram server</i>	35
Gambar 4.12 Model data logis	36
Gambar 4.13 Desain user experience halaman home	37
Gambar 4.14 Desain user experience halaman daftar konten	37
Gambar 4.15 <i>Page flow</i> aplikasi	38
Gambar 4.16 Desain user experience halaman <i>detail</i> konten	39
Gambar 5.1 Struktur bab implementasi	40
Gambar 5.2 Implementasi basis data	42
Gambar 5.3 Antarmuka halaman home	54
Gambar 5.4 Antarmuka halaman daftar konten	55
Gambar 5.5 Antarmuka halaman detail konten	58
Gambar 5.6 Tampilan film yang sedang diputar	59

DAFTAR KODE

Kode 2.1 Android Manifest Internet Permission	7
Kode 2.2 Penggunaan AsyncTask pada Android	8
Kode 2.3 Operasi jaringan pada Android	9
Kode 2.4 Penambahan <i>link</i> dalam dokumen	13
Kode 5.1 Kode program melihat konten.....	45
Kode 5.2 Kode program menambah konten.....	46
Kode 5.3 Kode program mengubah konten.....	47
Kode 5.4 Kode program menghapus konten	48
Kode 5.5 Kode program kelas ContentData.....	49
Kode 5.6 Kode program kelas MovieItem.....	49
Kode 5.7 Kode program mengambil konten bertipe film	50
Kode 5.8 Kode program membuka konten.....	51
Kode 5.9 Kode XML tampilan halaman home.....	53
Kode 5.10 Kode XML satu item dalam <i>gridview</i>	55
Kode 5.11 Kode XML antarmuka halaman detail konten	58



DAFTAR LAMPIRAN

LAMPIRAN A Contoh Kuesioner Pengujian Usabilitas	71
--	----



BAB 1 PENDAHULUAN

1.1 Latar belakang

Perkembangan teknologi pada transportasi umum di Indonesia saat ini mengalami kemajuan yang ditandai dengan meningkatnya kualitas transportasi umum baik dari sarana maupun prasarana dan juga fasilitas pendukung pada transportasi umum (BUMN, 2014). Tujuan dari perkembangan tersebut tidak lain adalah untuk meningkatkan tingkat kepuasan pengguna jasa transportasi umum baik transportasi darat, udara, dan laut. Kepuasan pengguna jasa transportasi umum sangat erat kaitannya dengan tingkat kenyamanan pengguna transportasi umum.

Dalam transportasi udara atau penerbangan, terdapat beberapa ketidaknyamanan yang dialami oleh penumpang yakni berupa psikologi yang negatif dan fisik yang stres misalnya, merasa mual, bosan, lelah, dan lain lain. Salah satu cara yang dapat digunakan untuk mengurangi ketidaknyamanan tersebut adalah dengan bersantai dengan bacaan yang menarik, film, musik atau game yang tersedia pada sistem hiburan dalam pesawat (Liu, 2007).

Teknologi hiburan yang ada dalam pesawat dapat juga diimplementasikan pada alat transportasi umum lainnya. Dikarenakan penyebab ketidaknyamanan hampir sama dengan ketika dalam penerbangan yakni psikologi yang negatif dan fisik yang stres, alat transportasi umum juga perlu adanya mekanisme hiburan untuk para penumpang. Teknologi hiburan yang ada pada alat transportasi umum saat ini biasanya hanya bersifat kolektif. Misalnya dalam satu bis hanya terdapat satu buah layar untuk menghibur seluruh penumpang dalam bis. Kendala yang terjadi adalah tidak semua penumpang terhibur dengan apa yang tersaji dalam layar tersebut, bahkan ada yang tidak suka atau bahkan terganggu. Melihat dari keberagaman penumpang yang ada pada alat transportasi umum, diperlukan teknologi hiburan yang mana penumpang dapat memilih konten hiburan yang sesuai dengan keinginan masing-masing penumpang.

Perangkat bergerak mengalami perkembangan yang pesat saat ini memungkinkan pengguna dapat menikmati hiburan secara nyaman dan menyenangkan. Perangkat bergerak yang sesuai untuk penggunaan dengan tujuan hiburan adalah tablet. Hal ini didasarkan pada data statistik dari Google (Admob, 2011) dan Gartner (Felix, 2013) yang menyatakan bahwa sebagian besar penggunaan tablet adalah untuk hiburan. Tablet didukung dengan layar yang lebar sehingga lebih nyaman digunakan untuk menikmati hiburan. Tablet juga didukung dengan fitur Wi-Fi yang dapat menghubungkan perangkat dengan jaringan *wireless*. Fitur-fitur di atas jika dikombinasikan dengan sebuah *server* yang memuat konten-konten hiburan, maka dapat dilakukan pengembangan sebuah aplikasi yang dapat digunakan sebagai media hiburan pada jasa transportasi umum.

Skripsi ini bertujuan untuk merancang dan mengimplementasikan aplikasi *entertainment access terminal* pada tablet Android yang memudahkan pengguna mengakses konten multimedia. Agar aplikasi dapat digunakan dengan maksimal diperlukan pengukuran untuk mengetahui tingkat kemudahan bagi pengguna, tingkat kesesuaian aplikasi terhadap kebutuhan fungsional, dan tingkat konsistensi sistem terhadap *request* klien. Implementasi penggunaan aplikasi ini ditujukan kepada penumpang dalam berbagai macam jasa transportasi umum misalnya pesawat, kereta, bis, dan travel. Implementasi dikhususkan menggunakan perangkat tablet. Penggunaan tablet dikarenakan kesesuaian tujuan dari aplikasi yaitu memuat konten hiburan yang mana pada tablet ditunjang dengan layar yang cukup luas untuk dapat menikmati hiburan film, musik, majalah atau buku.

Jika diperoleh sebuah aplikasi *entertainment access terminal* yang mudah digunakan oleh pengguna, sesuai dengan kebutuhan fungsional, dan konsistensi *server* terhadap *request* klien terjaga, maka aplikasi *entertainment access terminal* bisa menjadi solusi atas ketidaknyamanan pada jasa transportasi umum yang diakibatkan oleh stres dan rasa bosan. Diharapkan dengan aplikasi ini, penumpang dalam berbagai jasa transportasi umum dapat menikmati konten hiburan berupa film, musik, majalah, dan buku sehingga dapat mengurangi stres dan rasa bosan ketika dalam perjalanan.

1.2 Rumusan masalah

Berikut ini merupakan rumusan masalah yang akan dibahas pada skripsi ini:

1. Bagaimana rancangan dan implementasi aplikasi *entertainment access terminal* pada tablet Android yang memudahkan pengguna mengakses konten multimedia?
2. Bagaimana tingkat usabilitas, reliabilitas, dan fungsionalitas dari aplikasi *entertainment access terminal* pada tablet Android?

1.3 Batasan masalah

Agar permasalahan yang dibahas pada skripsi ini tidak menyimpang, maka perlu diberikan pembatasan masalah. Berikut ini merupakan batasan masalah dari skripsi ini:

1. Aplikasi memuat konten berupa musik, film, majalah, dan buku.
2. Aplikasi hanya mengakses konten yang telah ada pada *server* dengan format mkv, mp4, mp3, dan pdf.

1.4 Tujuan

Menghasilkan rancangan dan mengimplementasi aplikasi yang dapat berjalan pada perangkat tablet yang dapat digunakan untuk menikmati musik, film, majalah, dan buku yang mudah digunakan oleh pengguna, sesuai dengan kebutuhan fungsional, dan reliabilitas sistem terhadap banyaknya *request* terjaga.

1.5 Manfaat

Berikut ini merupakan manfaat dari skripsi ini:

1. Membantu para penumpang jasa transportasi umum mengurangi stress yang negatif dan rasa bosan ketika melakukan perjalanan.
2. Memudahkan para penumpang jasa transportasi umum ketika memilih hiburan tanpa mengganggu penumpang lain.
3. Memudahkan para pengelola jasa transportasi umum untuk memanajemen konten hiburan.

1.6 Sistematika penulisan

Sistematika penulisan ini ditujukan untuk memberikan gambaran dan uraian singkat terkait penulisan skripsi ini yang meliputi beberapa bab, yaitu:

1. **BAB 1 Pendahuluan**

Bab ini memuat tentang latar belakang, rumusan masalah, batasan masalah, tujuan, manfaat, dan sistematika penulisan.

2. **BAB 2 Landasan Kepustakaan**

Bab ini menguraikan kajian pustaka dan landasan teori yang mendasari proses perancangan dan pengimplementasian penelitian.

3. **BAB 3 Metodologi Penelitian**

Bab ini membahas tentang metode yang digunakan dalam penelitian dan langkah-langkah yang akan dilakukan dalam penulisan yang terdiri dari studi literatur, analisis dan perancangan sistem, implementasi, pengujian, dan pengambilan kesimpulan dan saran.

4. **BAB 4 Analisis dan Perancangan**

Bab ini membahas analisis kebutuhan dan perancangan aplikasi yang akan dibuat.

5. **BAB 5 Implementasi**

Bab ini membahas tentang implementasi dari sistem berdasarkan metodologi dan perancangan yang telah dibuat.

6. **BAB 6 Pengujian**

Bab ini memuat tentang hasil pengujian terhadap sistem yang telah diimplementasikan.

7. **BAB 7 Penutup**

Memuat kesimpulan yang diperoleh dari hasil pengujian aplikasi yang telah dibangun serta saran untuk pengembangan lebih lanjut.

BAB 2 LANDASAN KEPUSTAKAAN

2.1 Android

Android merupakan sebuah sistem operasi *open source* berbasis linux untuk perangkat bergerak yang dikembangkan oleh Google dan selanjutnya oleh Open Handset Alliance. *Programmer* Android dapat menulis kode dalam bahasa java dan mengontrol perangkat melalui *library* yang dikembangkan oleh Google. Google membeli pengembang Android pada tahun 2005 kemudian diluncurkan pada tahun 2007. Pihak Google merilis kode Android secara *open source* di bawah lisensi Apache. Pada mulanya, para pengembang membuat kode program dalam bahasa java kemudian aplikasi dapat diunduh melalui situs pihak ketiga ataupun toko *online*. Pada Februari 2012, 450.000 aplikasi Android telah tersedia untuk Android tetapi untuk perkiraan aplikasi yang telah diunduh sejak Desember 2011 telah lebih dari 10 miliar *download* (S, C, & S, 2013).

Fitur-fitur standar yang ada pada Android saat ini (Komputer, 2012):

1. Java Support: walaupun aplikasi Android ditulis di Java, tapi tidak ada *Java Virtual Machine* (JVM) di *platform* ini, sehingga java byte code tidak bisa dieksekusi. *Class* Java akan dikompilasi ke *dalvik executable* dan dijalankan di *dalvik* sebuah *virtual machine* khusus yang didesain secara spesial untuk Android.
2. Dukungan Multi bahasa
3. *Web Browser*
4. Dukungan Multimedia
5. Dukungan *Streaming Media*
6. Dukungan hardware tambahan: Android dapat juga menggunakan kamera video/ gambar, *touchscreen*, *Global Positioning System* (GPS), *accelerometer*, *gyroscope*, *barometer*, *magnetometer*, kontrol game khusus, dan sensor tekanan. Ada juga termometer, *blit* 2D bit (dengan orientasi *hardware*, *scaling*, *pixel*, dan akselerasi grafis 3D).
7. *Multi-touch*
8. *Video calling*
9. *Multi-tasking*
10. *Tethering*: Android mendukung *tethering* yang memungkinkan perangkat Android digunakan untuk hotspot Wi-Fi. Sebelum Android 2.2, fitur ini hanya didukung oleh aplikasi pihak ketiga.

Android didukung dengan kemampuan *streaming* multimedia. Protokol jaringan yang didukung oleh Android adalah:

1. RTSP (RTP, SDP)

2. HTTP/HTTPS *progressive streaming*3. HTTP/HTTPS *live streaming*

Format dan codec audio dan video yang didukung oleh Android dijelaskan pada Tabel 2.1. Android mendukung beberapa format dalam penyajian multimedia dengan kualitas yang berbeda-beda sesuai dengan format tipe file (Developer, Supported Media Formats, n.d.).

Tabel 2.1 Format dan codec audio dan video yang didukung oleh Android

Type	Format / Codec	Encoder	Decoder	Supported File Type(s) / Container Formats
Audio	AAC LC	•	•	• 3GPP (.3gp)
	HE-AACv1 (AAC+)	• (Android 4.1+)	•	• MPEG-4 (.mp4, .m4a)
	HE-AACv2 (enhanced AAC+)		•	• ADTS raw AAC (.aac, decode in Android 3.1+, encode in Android 4.0+, ADIF not supported)
	AAC ELD (enhanced low delay AAC)	• (Android 4.1+)	• (Android 4.1+)	• MPEG-TS (.ts, not seekable, Android 3.0+)
	AMR-NB	•	•	3GPP (.3gp)
	AMR-WB	•	•	3GPP (.3gp)
	FLAC		• (Android 3.1+)	FLAC (.flac) only
	MP3		•	MP3 (.mp3)
	MIDI		•	• Type 0 and 1 (.mid, .xmf, .mxmf) • RTTTL/RTX (.rtttl, .rtx) • OTA (.ota) • iMelody (.imy)

	Vorbis		•	• Ogg (.ogg) • Matroska (.mkv, Android 4.0+)
	PCM/WAVE	• (Android 4.1+)	•	WAVE (.wav)
	Opus		• (Android 5.0+)	Matroska (.mkv)
Video	H.263	•	•	• 3GPP (.3gp) • MPEG-4 (.mp4)
	H.264 AVC	• (Android 3.0+)	•	• 3GPP (.3gp) • MPEG-4 (.mp4) • MPEG-TS (.ts, AAC audio only, not seekable, Android 3.0+)
	H.265 HEVC		• (Android 5.0+)	• MPEG-4 (.mp4)
	MPEG-4 SP		•	3GPP (.3gp)
	VP8	• (Android 4.3+)	• (Android 2.3.3+)	• WebM (.webm) • Matroska (.mkv, Android 4.0+)
	VP9		• (Android 4.4+)	• WebM (.webm) • Matroska (.mkv, Android 4.0+)

Framework mesin media streaming yang ada pada Android terbagi menjadi lima layer yaitu user interface, data capture, data output, codec, dan network transport (Zhao & Tian, 2012).

- Layer user interface* merupakan antar muka dari mesin media streaming. Secara sederhana antar muka terdiri dari *start* dan *stop* dari mesin media streaming. Ketika fungsi start dipanggil, pesan akan dikirim dari *data capture*

melalui codec ke *network transport* atau dari *network transport* melalui codec ke *data output*.

- b. *Layer data capture* mempunyai tugas utama untuk menangkap data audio atau video. *Layer data capture* didefinisikan sebagai kelas turunan yang menyediakan beberapa antarmuka termasuk antarmuka pesan *input* dan *output*, antarmuka *open* dan *close* perangkat.
- c. *Layer data output* mempunyai desain yang mirip dengan *layer data capture*. *Layer data output* juga menghasilkan kelas turunan bernama *data output* dan menyediakan antarmuka perangkat *open* dan *close*, antarmuka *data output*.
- d. *Layer codec* berperan dalam kompresi dan dekompresi data media streaming, *layer codec* didefinisikan sebuah codec *class* yang mempunyai inisialisasi, pemilihan codec, *data input*, dan antarmuka *output*.
- e. *Layer network transport* mempunyai tugas utama untuk menyempurnakan fungsi pengiriman pada sistem media *streaming*. *Layer network transport* terbuat dari *network transmission class*, termasuk data input dan antarmuka inisialisasi *session*.

Metode yang dapat digunakan pada Android untuk dapat mengakses konten yang ada pada *server* melalui koneksi jaringan http ada 2 yakni `URLConnection` dan `HttpClient`. Keduanya mendukung fitur `HTTPS`, *streaming uploads* dan *downloads*, *timeout* yang dapat dikonfigurasi, `IPv6` dan *connection pooling* (Developer, Connecting to the Network, n.d.). Untuk dapat menggunakan koneksi dengan jaringan, manifest aplikasi harus disertakan *permission* sesuai dalam Kode 2.1.

```
<uses-permission Android:name="Android.permission.INTERNET" />
```

Kode 2.1 Android Manifest Internet Permission

Dalam operasi jaringan dapat terjadi *delay* yang tidak terprediksi. Untuk mencegah hal ini agar tidak menyebabkan permasalahan, operasi jaringan dapat dilakukan pada *thread* yang terpisah dari UI. Hal tersebut dapat dicapai dengan menggunakan `AsyncTask`. Dalam Kode 2.2 dan Kode 2.3 dijelaskan mengenai contoh penggunaan `AsyncTask` dan operasi jaringan untuk mengakses konten yang ada pada *server*. Pada Kode 2.2 terdapat kelas `DownloadWebpageTask`. Kelas `DownloadWebpageTask` merupakan kelas asynchronous yang mempunyai fungsi untuk mengunduh konten *webpage* yang ditunjukkan oleh fungsi `downloadUrl`.

```
public class HttpExampleActivity extends Activity {
    private static final String DEBUG_TAG = "HttpExample";
    private EditText urlText;
    private TextView textView;

    @Override
    public void onCreate(Bundle savedInstanceState) {
```

```

super.onCreate(savedInstanceState);
setContentView(R.layout.main);
urlText = (EditText) findViewById(R.id.myUrl);
textView = (TextView) findViewById(R.id.myText);
}

// When user clicks button, calls AsyncTask.
public void myClickHandler(View view) {
    // Gets the URL from the UI's text field.
    String urlString = urlText.getText().toString();
    ConnectivityManager connMgr = (ConnectivityManager)
        getSystemService(Context.CONNECTIVITY_SERVICE);
    NetworkInfo networkInfo = connMgr.getActiveNetworkInfo();
    if (networkInfo != null && networkInfo.isConnected()) {
        new DownloadWebpageTask().execute(urlString);
    } else {
        textView.setText("No network connection available.");
    }
}

private class DownloadWebpageTask extends AsyncTask<String, Void,
String> {
    @Override
    protected String doInBackground(String... urls)
    {
        try {
            return downloadUrl(urls[0]);
        } catch (IOException e) {
            return "Unable to retrieve web page. URL may be
invalid.";
        }
    }
    @Override
    protected void onPostExecute(String result) {
        textView.setText(result);
    }
}
...
}

```

Kode 2.2 Penggunaan AsyncTask pada Android

```
private String downloadUrl(String myurl) throws IOException {
```

```
InputStream is = null;
// Only display the first 500 characters of the retrieved
// web page content.
int len = 500;

try {
    URL url = new URL(myurl);
    HttpURLConnection conn = (HttpURLConnection)
url.openConnection();
    conn.setReadTimeout(10000 /* milliseconds */);
    conn.setConnectTimeout(15000 /* milliseconds */);
    conn.setRequestMethod("GET");
    conn.setDoInput(true);
    // Starts the query
    conn.connect();
    int response = conn.getResponseCode();
    Log.d(DEBUG_TAG, "The response is: " + response);
    is = conn.getInputStream();
    // Convert the InputStream into a string
    String contentAsString = readIt(is, len);
    return contentAsString;
} finally {
    if (is != null) {
        is.close();
    }
}
}
```

Kode 2.3 Operasi jaringan pada Android

Pada Kode 2.3 dijelaskan mengenai penggunaan `HttpURLConnection` untuk mengakses konten yang ada pada *server* dalam hal ini berupa *webpage*. Sebelum aplikasi memulai proses akses ke *server*, terlebih dahulu dilakukan konfigurasi sesuai dengan baris tujuh sampai baris ke-14. Kemudian mulai melakukan akses ke *server* pada baris ke 14.

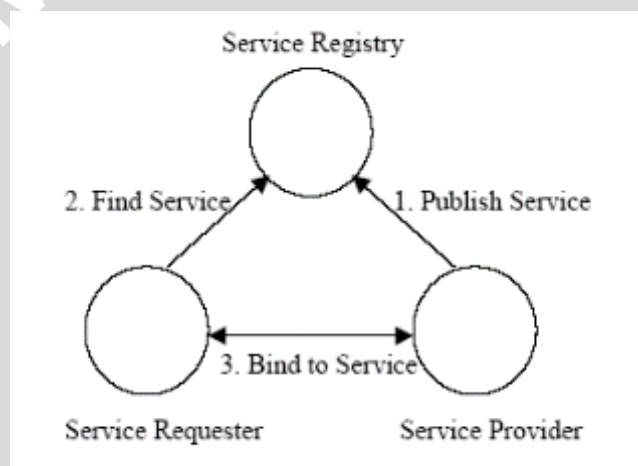
2.2 Web service

Web service merupakan sistem software yang dirancang untuk interaksi mesin ke mesin pada suatu jaringan yang memungkinkan untuk bertukar informasi serta menggunakannya. *Web service* mempunyai antarmuka yang didefinisikan dalam format yang dapat dibaca oleh mesin. Sistem lain dapat berinteraksi dengan *web service* menggunakan format yang telah didefinisikan dalam pesan SOAP, disampaikan melalui HTTP dalam bentuk XML (Booth, 2004).

Menurut penjelasan *web service* di atas, interaksi antar mesin dapat dilakukan dengan mudah meskipun mempunyai *platform* yang berbeda. Pada skripsi ini, *web service* digunakan sebagai bentuk interaksi antara *server* dengan *client*. *Server* menggunakan *platform* PHP sedangkan *client* menggunakan OS Android. Interaksi dapat dilakukan dengan cara *client* mengirim pesan ke *server* menggunakan format yang dikenali oleh keduanya kemudian *server* menanggapi dengan cara yang sama. Format pertukaran data yang dipakai dalam skripsi ini menggunakan JavaScript Object Notation (JSON).

Web service memiliki tiga entitas dalam arsitekturnya sesuai dengan Gambar 2.1 yaitu (Utama):

1. *Service Requester* (peminta layanan)
2. *Service Provider* (penyedia layanan)
3. *Service Registry* (daftar layanan)



Gambar 2.1 Arsitektur Web service

Sumber: (Utama)

1. *Service Provider*: Berfungsi untuk menyediakan layanan/service dan mengolah sebuah registry agar layanan-layanan tersebut dapat tersedia.
2. *Service Registry*: Berfungsi sebagai lokasi central yang mendeskripsikan semua layanan/service yang telah di-register.
3. *Service Requestor*: Peminta layanan yang mencari dan menemukan layanan yang dibutuhkan serta menggunakan layanan tersebut.

Beberapa alasan mengapa digunakannya *web service* adalah sebagai berikut (Utama):

1. *Web service* dapat digunakan untuk mentransformasikan satu atau beberapa bisnis *logic* atau *class* dan objek yang terpisah dalam satu ruang lingkup yang menjadi satu, sehingga tingkat keamanan dapat ditangani dengan baik.
2. *Web service* memiliki kemudahan dalam proses *deployment*-nya, karena tidak memerlukan registrasi khusus ke dalam suatu sistem operasi. *Web service*

cukup di-upload ke web server dan siap diakses oleh pihak-pihak yang telah diberikan otorisasi.

3. *Web service* berjalan di port 80 yang merupakan protokol standar HTTP, dengan demikian *web service* tidak memerlukan konfigurasi khusus di sisi *firewall*.

2.3 REST

REpresentational State Transfer (REST) merupakan salah satu metode dalam *web service*. Berbeda dengan protokol-protokol lain dalam *web service* seperti SOAP atau XML-RPC, REST lebih bersifat mendasar atau berupa prinsip-prinsip daripada sebuah protokol itu sendiri. REST merupakan suatu gagasan mengenai bagaimana data dapat dikirim secara elegan. REST mengambil keuntungan yang besar dari fitur-fitur yang telah ada pada HTTP yakni empat HTTP *verbs* yang dapat digunakan untuk menghasilkan sekumpulan *Create, Read, Update, Delete* (CRUD) dasar. Empat HTTP *verbs* tersebut adalah POST, GET, PUT, DELETE. [MIT-13]

Prinsip-prinsip arsitektural dalam REST (Burke, 2013):

1. *Addressability*

Addressability merupakan prinsip pada REST yang menekankan bahwa setiap objek dan *resource* dalam sistem dapat dijangkau melalui URI (*Unique Resource Identifiers*).

2. *The Uniform, Constrained Interface*

Prinsip ini bertujuan untuk membatasi operasi-operasi pada protokol aplikasi. Hal ini mempunyai arti bahwa aplikasi tidak mempunyai parameter "aksi" pada URI, melainkan hanya menggunakan HTTP *method* pada *web service*. Berikut beberapa http method beserta fungsinya:

GET

GET merupakan operasi yang bersifat *read-only* yang digunakan untuk meminta suatu informasi ke *server*. Informasi yang diminta akan terlihat pada URI yang dikirim ke *server*. GET biasanya digunakan untuk melakukan operasi *read*.

PUT

PUT biasanya digunakan untuk melakukan perubahan pada *resource* yang telah ada ataupun membuat *resource* baru sesuai dengan URI. Klien mengirimkan representasi *resource* bersama dengan *request* tersebut. *Server* menerima *request* tersebut kemudian akan membaca representasinya, dan melakukan *update*. Jika *resource* yang dimaksud belum ada, maka akan terjadi operasi pembuatan *resource* baru.

DELETE

DELETE digunakan untuk menghapus *resource*.

POST

POST hampir mirip dengan PUT. POST juga dapat membuat *resource* baru. Kunci perbedaan antara PUT dan POST adalah PUT merupakan idempotent sedangkan POST tidak. Meskipun berapa kali *request* PUT dikirim, hasilnya akan sama (idempotent). Sedangkan melakukan request POST yang berkali akan mengakibatkan *resource* yang akan dibuat pada *server* menjadi berlipat-lipat (Vaqqas, 2014).

Pada Tabel 2.2 merupakan penjelasan mengenai perbedaan antara POST dan PUT.

Tabel 2.2 Perbedaan antara POST dan PUT

Request	Operasi
PUT http://MyService/Persons	Tidak akan bekerja. PUT memerlukan URI yang lengkap
PUT http://MyService/Persons/1	Melakukan <i>insert</i> person dengan PersonID = 1 jika <i>resource</i> belum ada, atau akan melakukan <i>update resource</i> yang telah ada.
POST http://MyService/Persons	Melakukan <i>insert</i> person tiap kali request dilakukan dan akan <i>generate</i> PersonID baru
POST http://MyService/Persons/1	Melakukan <i>update</i> person yang telah ada dengan PersonID = 1

3. Representation-Oriented

Pada sistem *web service* REST, kerumitan interaksi *client-server* berada pada representasi yang dikirimkan keduanya. Representasi dapat berupa XML, JSON, YAML atau yang lainnya.

4. Communicate Statelessly

Pada REST, yang dimaksud dengan *stateless* adalah tidak ada data session klien yang disimpan dalam *server*. *Server* hanya merekam dan mengelola *state* dari *resource* yang dibuka. Jika dibutuhkan data yang tergantung pada *session*, hal tersebut harus ditangani oleh klien dan session akan dikirim ke *server* setiap kali dibutuhkan saat *request*.

5. HATEOAS

Hypermedia as the Engine of Application State (HATEOAS) merupakan prinsip REST terakhir yakni menggunakan hypermedia sebagai mesin dari *state* aplikasi. Hypermedia merupakan pendekatan yang berpusat pada dokumen dengan tambahan dukungan untuk menambahkan *link* ke *service* dan informasi lain dalam format dokumen.

```

<orderid="111">
<customer>http://customers.myintranet.com/customers/32133
  </customer>
  <order-entries>
<order-entry>
<quantity>5</quantity>
<product>http://products.myintranet.com/products/111
  </product>
...

```

Kode 2.4 Penambahan *link* dalam dokumen

Kode 2.4 merupakan contoh *link* ditanamkan dalam dokumen sehingga memungkinkan untuk ‘membawa’ informasi tambahan yang dibutuhkan. Pada Kode 2.4 terdapat dua buah pranala untuk *customer* dan *product* yang masing-masing membawa informasi mengenai id *customer* atau *product* yang berkaitan.

2.4 JSON

JavaScript Object Notation (JSON) merupakan format pertukaran data yang ringan. JSON mudah dibaca dan ditulis oleh manusia. JSON juga mudah untuk melakukan *parse* maupun *generate* oleh mesin. JSON merupakan format teks yang sepenuhnya bahasa tersendiri, tetapi menggunakan ketentuan yang familiar bagi *programmer* keluarga bahasa C yakni C, C++, C#, Java, JavaScript, Perl, Python, dan masih banyak lainnya. Hal ini membuat JSON merupakan bahasa pertukaran data yang ideal (JSON, n.d.).

Pada bahasa pemrograman apapun, terdapat berbagai macam tipe *number* untuk berbagai kapasitas dan komplemen, seperti fixed atau float, biner atau desimal, dll. Keberagaman tersebut dapat membuat proses pertukaran data antar bahasa pemrograman yang berbeda menjadi susah. Pada JSON, *number* hanya direpresentasikan sesuai dengan yang digunakan manusia yakni *sequence of digits* (1, 2, 3, 4 ...). Semua bahasa pemrograman dapat menerima format *number sequence of digits* sehingga pertukaran dapat dilakukan (International, 2013).

Teks JSON merupakan rangkaian dari karakter yang dibentuk dari kode Unicode yang sesuai dengan aturan JSON *value*.

Enam karakter struktural:

[U+005B	awalan kurung siku
{ U+007B	awalan kurung kurawal
] U+005D	akhiran kurung siku
} U+007D	akhiran kurung kurawal
: U+003A	titik dua
, U+002C	koma

Tiga karakter literal:

true U+0074 U+0072 U+0075 U+0065

false U+0066 U+0061 U+006c U+0073 U+0065

null U+006E U+0075 U+006C U+006C

Whitespace yang diperbolehkan sebelum atau setelah karakter-karakter di atas

U+0020 Spasi

U+0009 *Horizontal Tab*

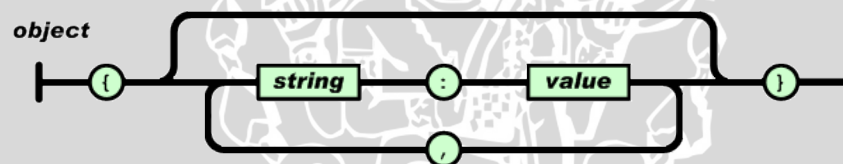
U+000A Baris baru

U+000D *Carriage return*

Bentuk-bentuk yang ada dalam JSON [JSN-15]:

1. Objek

Objek merupakan sekumpulan pasangan *name/value* yang tidak teratur. Sebuah objek dimulai dengan tanda awalan kurung kurawal ({) dan diakhiri dengan akhiran kurung kurawal (}). Masing- masing name diikuti dengan tanda titik dua (:) dan pasangan *name/value* dipisahkan oleh tanda koma (,). Gambar 2.2 merupakan struktur json untuk tipe objek.

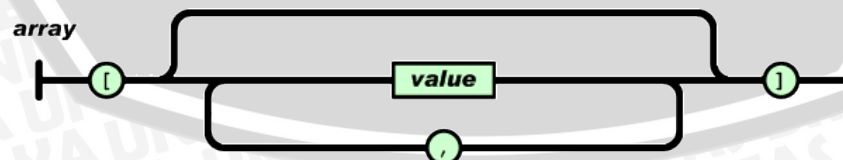


Gambar 2.2 Objek pada JSON

Sumber: (JSON, n.d.)

2. Array

Array merupakan kumpulan dari *value* yang teratur. Array dimulai dengan awalan kurung siku ([) dan diakhiri dengan akhiran kurung siku (]). *Value* dipisahkan oleh koma (,). Gambar 2.3 merupakan struktur json untuk tipe array.

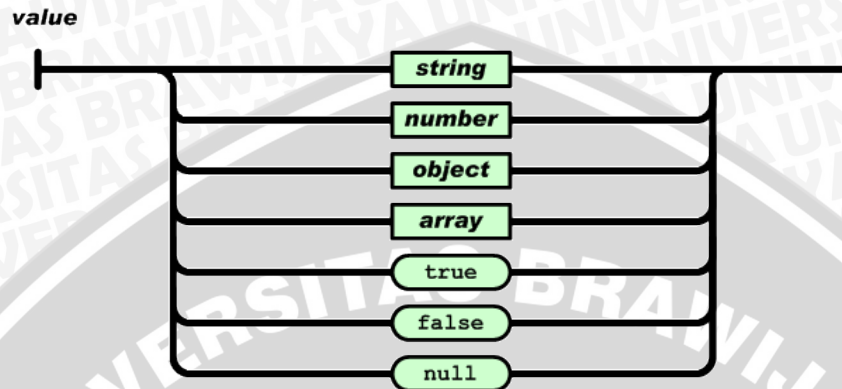


Gambar 2.3 Array pada JSON

Sumber: (JSON, n.d.)

3. Value

Value dapat berupa *string* di dalam *double quotes* ("), *number*, *true*, *false*, *null*, *object*, atau *array*. Gambar 2.4 merupakan struktur json untuk tipe *value*.

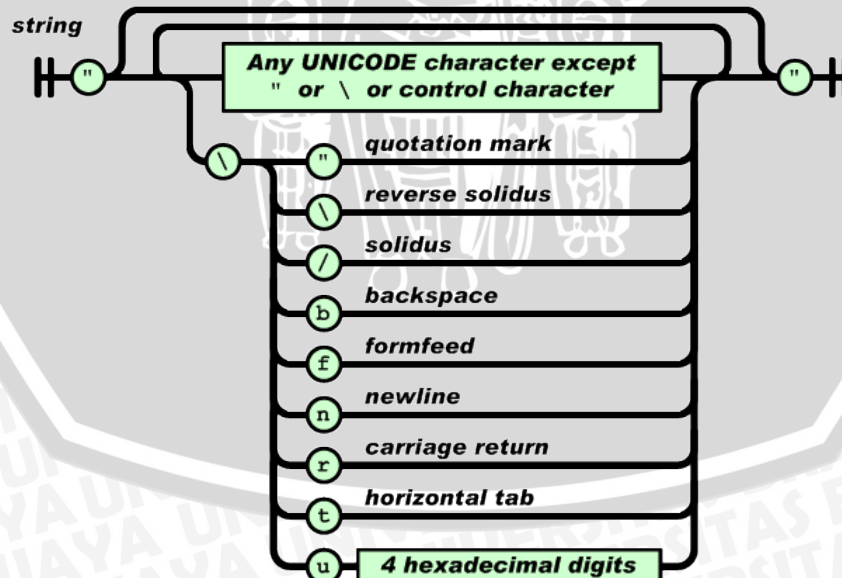


Gambar 2.4 Value pada JSON

Sumber: (JSON, n.d.)

4. String

String merupakan rangkaian dari nol atau lebih karakter Unicode yang dibungkus dengan *double quotes* ("). String tunggal dinamakan karakter. String hampir sama seperti string yang ada pada bahasa C atau Java. Gambar 2.5 merupakan struktur json untuk tipe string.

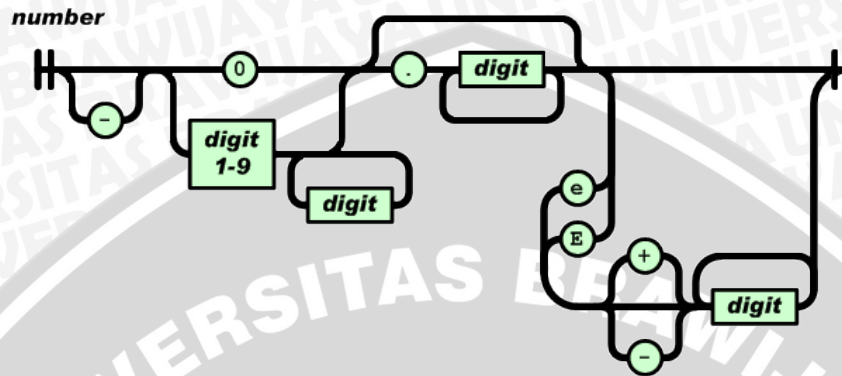


Gambar 2.5 String pada JSON

Sumber: (JSON, n.d.)

5. Number

Number hampir sama seperti yang ada pada bahasa C dan Java, terkecuali tidak digunakannya oktal dan heksadesimal. Gambar 2.6 merupakan struktur JSON untuk tipe *number*.



Gambar 2.6 Objek pada JSON

Sumber: (JSON, n.d.)

2.5 System Usability Scale

System Usability Scale (SUS) adalah kuesioner 10 item dengan lima pilihan jawaban untuk mengukur usabilitas yang dirilis pada tahun 1986 oleh John Brooke saat bekerja di Digital Equipment Corporation. SUS adalah teknologi independen dan telah diuji pada *hardware*, *software* konsumen, dan *website*. Sepuluh kuesioner dalam SUS dalam bahasa inggris yaitu:

1. *I think that I would like to use this system frequently.*
2. *I found the system unnecessarily complex.*
3. *I thought the system was easy to use.*
4. *I think that I would need the support of a technical person to be able to use this system.*
5. *I found the various functions in this system were well integrated.*
6. *I thought there was too much inconsistency in this system.*
7. *I would imagine that most people would learn to use this system very quickly.*
8. *I found the system very cumbersome to use.*
9. *I felt very confident using the system.*
10. *I needed to learn a lot of things before I could get going with this system.*

Format respon untuk kuesioner SUS menggunakan skor satu sampai dengan lima untuk tiap-tiap nomor sesuai dalam Gambar 2.7. Metode penghitungan kuesioner dilakukan dengan memberi nilai pada masing-masing pilihan jawaban

yaitu Sangat Tidak Setuju (STS) bernilai 1, Tidak Setuju (TS) bernilai 2, Netral (N) bernilai 3, Setuju (S) bernilai 4, dan Sangat Setuju (SS) bernilai 5.

Strongly Disagree 1	2	3	4	Strongly Agree 5
☐	☐	☐	☐	☐

Gambar 2.7 Format respon kuesioner SUS

Pemberian nilai untuk nomor kuesioner ganjil didapat dengan cara mengurangi satu dari nilai jawaban kemudian dikalikan dengan jumlah pemilih jawaban tersebut. Pemberian nilai untuk nomor kuesioner genap, pemberian nilai dengan cara mengurangi lima dengan nilai jawaban kemudian dikalikan dengan jumlah pemilih jawaban tersebut. Menjumlahkan tiap-tiap kuesioner dan mengalikannya dengan 2.5 (Sauro, 2011).

Skor SUS tidak dalam bentuk persentase. Meskipun rentang skor SUS dari 0 sampai 100, ini bukan persentase. Secara teknis benar bahwa sebuah skor SUS 70 dari 100 mewakili 70% dari kemungkinan nilai maksimum, ini menunjukkan skor berada pada titik persentil ke-70. Skor pada tingkat ini akan berarti aplikasi yang telah diuji berada diatas rata-rata. Faktanya, skor 70 lebih dekat pada rata-rata skor SUS 68. Sebenarnya lebih tepat untuk menyebutnya 50%. Ketika menyampaikan skor SUS kepada *stakeholders*, dan khususnya kepada yang tidak familiar dengan SUS, lebih baik untuk mengubah skor SUS asli menjadi sebuah persentil, sehingga 70% benar-benar berarti di atas rata-rata.

Validitas mengacu pada seberapa baik sesuatu dapat mengukur apa yang dimaksudkan untuk mengukur. Dalam hal ini validitas untuk mengukur usability. SUS terbukti secara efektif dapat membedakan antara sistem yang tidak dapat digunakan dan yang dapat digunakan sama baiknya atau bahkan lebih baik daripada kuesioner lain. SUS juga sangat berhubungan dengan pengukuran usability lain yang berbasis kuesioner (disebut validitas bersamaan).

SUS tidak dimaksudkan untuk mendiagnosa permasalahan usability. Dalam penggunaannya yang asli, SUS diadakan setelah uji usability dimana semua sesi pengguna tercatat pada rekaman video (VHS dan Betamax). Skor SUS yang rendah menunjukkan kepada peneliti bahwa mereka perlu melakukan peninjauan ulang terhadap rekaman dan mengidentifikasi masalah yang ditemui dengan antarmukanya. SUS dapat digunakan diluar dari pengujian usability untuk perbandingan. Namun, hasilnya tidak memberikan penjelasan mengenai mengapa user merespon seperti itu. (Sauro, 2011).

BAB 3 METODOLOGI

Pada bab ini dijelaskan langkah-langkah yang akan dilakukan dalam penelitian ini. Langkah pertama yang dilakukan yaitu mengumpulkan teori-teori pendukung dituliskan ke dalam studi literatur. Kemudian dilanjutkan dengan proses analisis kebutuhan. Setelah analisis kebutuhan dilakukan, proses selanjutnya yaitu melakukan perancangan berdasarkan analisis kebutuhan yang telah dilakukan. Kemudian melakukan implementasi sesuai dengan perancangan dan melakukan pengujian. Pengambilan kesimpulan dan saran dilakukan sebagai catatan atas penelitian yang telah dilakukan serta sebagai petunjuk ke arah pengembangan penelitian lebih lanjut. Gambar 3.1 merupakan diagram alir tahapan pengerjaan penelitian.



Gambar 3.1 Diagram Alir Tahapan Penelitian

3.1 Studi Literatur

Studi literatur menjelaskan dasar teori yang digunakan sebagai acuan serta penunjang dalam penulisan skripsi. Studi literatur dilakukan dengan tujuan untuk menambah pengetahuan penulis sehingga dapat meningkatkan kualitas penelitian yang dibuat. Teori pendukung tersebut meliputi:

1. Android
2. Web service
3. REST
4. JSON.
5. *System Usability Scale*

3.2 Analisis Kebutuhan

Analisis kebutuhan diperlukan sebagai dasar dalam pembuatan aplikasi. Analisis kebutuhan bertujuan untuk mendapatkan semua kebutuhan yang diperlukan oleh aplikasi yang akan dibangun. Analisis kebutuhan diperoleh melalui identifikasi kebutuhan apa saja yang ingin didapatkan pada aplikasi (*requirements*). Kemudian melakukan pendekatan dengan cara memodelkan *requirements* yang telah didapat menggunakan bahasa pemodelan Unified Modeling Language (UML).

Sebelum analisis kebutuhan dilakukan, penelitian ini diawali dengan melakukan observasi untuk menentukan kebutuhan sistem. Observasi dilakukan dengan mencari informasi mengenai sistem hiburan yang ada di dalam pesawat melalui katalog produk sistem hiburan dalam pesawat. Hasil dari observasi kemudian dijadikan acuan untuk melakukan analisis kebutuhan. Proses yang dilakukan selanjutnya dalam analisis kebutuhan ini adalah membuat gambaran secara umum mengenai aplikasi berdasarkan hasil observasi yang telah dilakukan serta menentukan daftar kebutuhan yang diperlukan dalam pembangunan aplikasi. Kemudian memodelkan daftar kebutuhan ke dalam diagram *use case* dan skenario *use case*.

3.3 Perancangan

Tahap perancangan sistem dilakukan dengan tujuan untuk membuat rancangan sistem yang digunakan sebagai acuan dalam pembuatan aplikasi. Perancangan yang dilakukan menggunakan bahasa pemodelan UML. Perancangan sistem yang dilakukan pada penelitian ini mengacu pada *use case* dan skenario *use case* pada tahap analisis kebutuhan.

Perancangan dimulai dengan pembuatan arsitektur aplikasi secara keseluruhan. Kemudian membuat *activity diagram* berdasarkan alur aktivitas dalam skenario *use case* yang telah dibuat pada tahap analisis kebutuhan. Kemudian membuat diagram *sequence*, *page flow*, dan desain *user experience*

aplikasi berdasarkan diagram *activity*. Selanjutnya, membuat *class diagram* dan merancang basis data menggunakan diagram konseptual entitas untuk mendukung pembuatan aplikasi.

3.4 Implementasi

Implementasi merupakan tahap pembangunan aplikasi berdasarkan hasil yang diperoleh pada tahap perancangan yang telah dilakukan. Hasil dari tahap perancangan berupa diagram *sequence*, *page flow*, desain *user experience*, *class diagram*, dan model data logis. Implementasi dilakukan dengan menggunakan bahasa *native* Android yakni bahasa pemrograman Java pada sisi klien dan menggunakan bahasa pemrograman PHP pada sisi *server*. Implementasi *user interface* didasarkan pada perancangan *page flow* dan *user experience* pada tahap perancangan menggunakan bahasa XML.

Implementasi *server* menggunakan XAMPP dan basis data untuk pengelolaan data secara terpusat pada *server* menggunakan DBMS yakni MySQL. Format pertukaran data antara *server* dan klien menggunakan JSON.

3.5 Pengujian

Teknik pengujian perangkat lunak yang digunakan dalam penelitian ini adalah teknik *black box testing*. Pengujian yang dilakukan pada penelitian ini ada tiga metode yaitu pengujian usabilitas, pengujian reliabilitas, dan pengujian fungsionalitas.

Skenario pengujian usabilitas dilakukan dengan menggunakan kuesioner *software usability scale* untuk mengukur tingkat usabilitas dari aplikasi. Kuesioner dijawab oleh responden. Kemudian tiap-tiap kuesioner yang telah dijawab oleh responden dihitung dan diambil rata-rata dari skor tiap kuesioner.

Pengujian reliabilitas dilakukan dengan menggunakan apache benchmark tools. Apache bench tools merupakan alat yang dapat digunakan untuk melakukan *load test* sebuah aplikasi. Pada apache bench tools terdapat fitur yang memungkinkan untuk mengatur jumlah *request* ke *server* serta banyaknya *user* yang melakukan *request* secara bersamaan (*concurrent*). Pengujian dilakukan dengan mengatur jumlah *request* yang berbeda-beda ke *server* serta banyaknya *user* yang melakukan *request*.

Pengujian fungsionalitas dilakukan untuk memeriksa apakah aplikasi yang telah dibuat telah sesuai dengan daftar kebutuhan yang ada pada tahap analisis kebutuhan. Skenario pengujian fungsionalitas dilakukan dengan menjalankan aplikasi sesuai dengan kebutuhan fungsional. Pengujian tiap kebutuhan fungsional bernilai valid jika sistem telah memenuhi kebutuhan fungsional dan bernilai tidak valid jika tidak memenuhi kebutuhan fungsional.

3.6 Pengambilan Kesimpulan dan Saran

Pengambilan kesimpulan dilakukan setelah semua tahapan analisis, perancangan, implementasi, dan pengujian telah selesai dilaksanakan. Kesimpulan diambil dari hasil pengujian aplikasi yang telah dibangun untuk mengetahui apakah permasalahan yang telah dirumuskan dapat diselesaikan atau tidak. Tahap terakhir penulisan adalah saran yang dimaksudkan untuk memperbaiki kekurangan yang terjadi dan menyempurnakan penulisan serta mengembangkan penelitian lebih lanjut.



BAB 4 ANALISIS DAN PERANCANGAN

Pada bab ini akan menjelaskan mengenai analisis kebutuhan dan perancangan aplikasi *entertainment access terminal* pada tablet Android. Tahap pertama yakni proses analisis kebutuhan yang terdiri dari gambaran umum aplikasi, identifikasi aktor, analisis kebutuhan fungsional dan non-fungsional, kemudian dimodelkan ke dalam *use case*. Tahap kedua yakni proses perancangan yang terdiri dari perancangan arsitektural, perancangan *activity diagram*, perancangan *sequence diagram*, perancangan *class diagram*, perancangan basis data, perancangan *page flow*, perancangan *user experience* dan perancangan format konten multimedia.

4.1 Analisis Kebutuhan

Sebelum analisis kebutuhan dilakukan, pengumpulan *requirements* diperoleh dari proses analisis *paper* (Liu, 2007) dan katalog produk (Schuberth, 2011) yang berkaitan dengan sistem hiburan dalam pesawat. Sistem hiburan dalam pesawat digunakan sebagai referensi dikarenakan kemiripan tujuan dari pembangunan aplikasi pada penelitian ini dengan aplikasi yang ada pada sistem hiburan dalam pesawat. Setelah didapat *requirements*, kemudian merumuskan kebutuhan fungsional. Selanjutnya, dimodelkan ke dalam diagram *use case*.

4.1.1 Gambaran Umum Aplikasi

Konsep dari "Rancang Bangun Aplikasi *Entertainment Access Terminal* Pada Tablet Android" adalah menampilkan konten-konten hiburan pada tablet Android yang mana konten hiburan tersebut diperoleh dari *server*. Konten hiburan terbagi menjadi empat jenis yakni film, musik, buku, dan majalah. Pada *prototype* aplikasi ini akan digunakan sebuah tablet Android yang telah terkoneksi pada *server*. Ilustrasi dari gambaran umum perangkat lunak dapat dilihat dalam Gambar 4.1.



Gambar 4.1 Ilustrasi gambaran umum aplikasi

Skenario sistem yang diusulkan adalah sebagai berikut. Pada *server* tersedia konten hiburan yang dapat dikelola oleh admin. Klien menggunakan tablet sebagai media yang menampilkan konten hiburan. Pengguna aplikasi mengakses aplikasi klien. Kemudian pengguna memilih jenis hiburan yang tersedia. Aplikasi akan melakukan *request* data konten ke *server* sesuai dengan jenis hiburan yang dipilih pengguna. *Server* akan membalas *request* tersebut dalam format json. Aplikasi klien menampilkan data dari *server*. Kemudian pengguna memilih konten yang diinginkan dan aplikasi akan menampilkan konten yang didapat dari *server*.

4.1.2 Identifikasi Aktor

Tahap identifikasi aktor bertujuan untuk mengetahui siapa saja aktor yang terlibat dalam aplikasi *entertainment access terminal* pada tablet Android. Aktor merupakan segala sesuatu yang berada di luar sistem yang akan berinteraksi dengan aplikasi ini. Peran dari aktor dapat dilihat pada Tabel 4.1.

Tabel 4.1 Identifikasi aktor

Aktor	Deskripsi
Pengguna (<i>User</i>)	Pengguna merupakan semua orang yang menggunakan aplikasi ini.
Admin	Admin merupakan aktor yang bertanggung jawab dalam mengelola data dan konten yang ada pada <i>server</i>

4.1.3 Analisis Kebutuhan Fungsional dan Non-Fungsional

Pada tahap analisis kebutuhan fungsional ini akan didaftar kebutuhan-kebutuhan apa saja yang diperlukan untuk membangun aplikasi *entertainment access terminal* pada tablet Android. Daftar kebutuhan fungsional aplikasi dapat dilihat pada Tabel 4.2.

Tabel 4.2 Daftar kebutuhan fungsional

Kode	Kebutuhan Fungsional	Use case
F01	Aplikasi dapat mengambil data konten multimedia dari <i>server</i>	Melihat list
F02	Aplikasi dapat memainkan konten multimedia dari <i>server</i>	Membuka konten
F03	Konten pada <i>server</i> dapat ditambah, diubah dan dihapus	Mengelola konten

Tabel 4.2 merupakan tabel daftar kebutuhan fungsional yang terdiri dari tiga kebutuhan yang harus ada pada aplikasi. Setiap kebutuhan fungsional aplikasi

dimodelkan ke dalam diagram *use case*. Daftar kebutuhan non-fungsional aplikasi dapat dilihat pada Tabel 4.3.

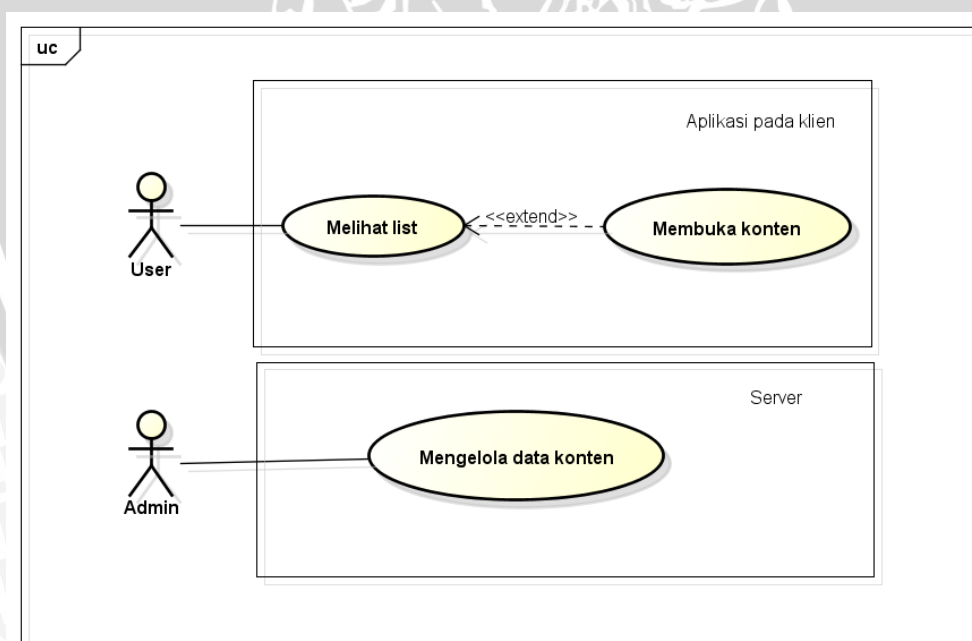
Tabel 4.3 Daftar kebutuhan non-fungsional

Kode	Kebutuhan non-fungsional	Keterangan
NF01	Usabilitas	Kemudahan penggunaan aplikasi oleh user.
NF02	Reliabilitas	Konsistensi <i>output</i> tiap-tiap <i>request</i> dari <i>client</i> dengan <i>server</i>

Tabel 4.3 merupakan tabel daftar kebutuhan non-fungsional yang terdiri dari dua kebutuhan yakni usabilitas dan reliabilitas. Usabilitas yakni aplikasi dapat digunakan oleh pengguna dengan mudah. Reliabilitas yakni konsistensi *output* tiap-tiap *request* dari *client* dengan *server*.

4.1.4 Diagram Use Case

Diagram *use case* merupakan salah satu diagram dalam UML yang digunakan untuk menggambarkan interaksi antara aplikasi dan aktor berdasarkan kebutuhan fungsional. Gambar 4.2 menunjukkan diagram *use case* aplikasi *entertainment access terminal*.



powered by Astah

Gambar 4.2 Diagram use case aplikasi entertainment access terminal

Gambar 4.2 merupakan diagram *use case* aplikasi *entertainment access terminal* yang terdiri dari 2 aktor yakni *user* (pengguna) dan *admin*. *Admin* dapat mengelola konten yang ada pada *server* sedangkan *user* dapat mengakses aplikasi dan tidak dapat mengelola konten.

4.1.5 Skenario *Use Case*

Pada Tabel 4.4 merupakan skenario *use case* melihat list. *List* yang dimaksud disini merupakan daftar data konten hiburan yang ditampilkan oleh aplikasi. Daftar data konten dipisahkan tiap-tiap jenis hiburan yang dipilih. Pada Tabel 4.5 merupakan skenario *use case* membuka konten. *Use case* membuka konten dapat dipenuhi ketika *use case* melihat list telah terpenuhi. Skenario mengelola data konten dijelaskan pada Tabel 4.6. Pengelolaan data konten dilakukan pada *server* yang dapat diakses melalui *web browser*.

Tabel 4.4 Skenario *use case* melihat list

Kode <i>use case</i>	F01
Nama <i>use case</i>	Melihat list
Aktor	<i>User</i>
Tujuan	Untuk mengetahui informasi mengenai konten yang dimuat dalam aplikasi.
Deskripsi	Fitur yang ada dalam aplikasi yang berguna untuk mengetahui daftar konten yang tersedia.
Pra-kondisi	<i>User</i> membuka jenis hiburan yang diinginkan pada aplikasi.
Pos-kondisi	<i>User</i> akan mendapatkan informasi daftar konten yang tersedia.
Alur Utama	Aktivitas
	1. <i>User</i> membuka aplikasi <i>entertainment access terminal</i> 2. Aplikasi akan menampilkan jenis hiburan 3. <i>User</i> memilih jenis hiburan 4. Aplikasi menampilkan informasi mengenai data konten sesuai jenis hiburan yang dipilih.

Tabel 4.5 Skenario *use case* membuka konten

Kode <i>use case</i>	F02
Nama <i>use case</i>	Membuka konten
Aktor	<i>User</i>
Tujuan	Untuk mengeksekusi pilihan konten dari user.
Deskripsi	Fitur yang ada dalam aplikasi yang berguna untuk mengeksekusi pilihan konten dari user.

Pra-kondisi	User memilih daftar konten yang tersedia pada aplikasi.
Pos-kondisi	User dapat menikmati konten hiburan yang telah dipilih.
Alur Utama	Aktivitas 1. User membuka aplikasi <i>entertainment access terminal</i> 2. Aplikasi akan menampilkan jenis hiburan 3. User memilih jenis hiburan 4. Aplikasi menampilkan informasi mengenai data konten sesuai jenis hiburan yang dipilih. 5. User memilih konten hiburan yang akan dibuka. 6. Aplikasi akan membuka konten yang telah dipilih oleh user.

Tabel 4.6 Skenario *use case* mengelola data konten

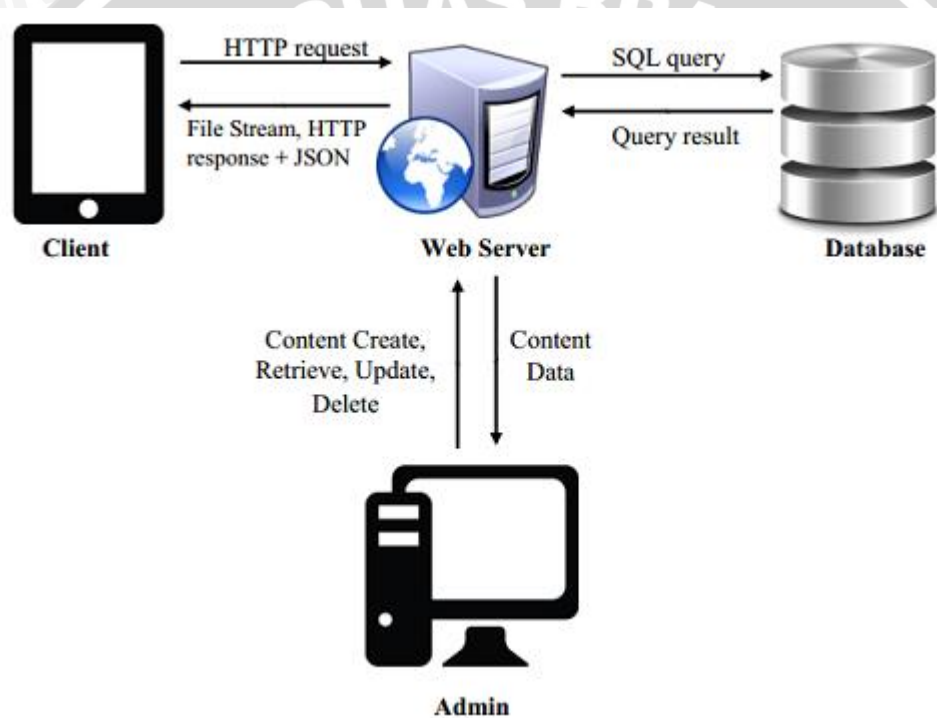
Kode <i>use case</i>	F03
Nama <i>use case</i>	Mengelola data konten
Aktor	Admin
Tujuan	Untuk mengelola data konten yang ada pada <i>server</i>
Deskripsi	Fitur yang ada dalam aplikasi yang berfungsi untuk menambah, melihat, mengganti, atau menghapus data konten yang ada pada <i>server</i> .
Pra-kondisi	Admin membuka halaman web yang digunakan sebagai <i>server</i> dari aplikasi.
Pos-kondisi	Data konten dapat dikelola oleh admin.
Alur Utama	Aktivitas 1. Admin membuka halaman web yang digunakan sebagai <i>server</i> dari aplikasi. 2. <i>Server</i> menampilkan pilihan CRUD yang digunakan pada data konten. 3. Admin melakukan pengelolaan data konten.

4.2 Perancangan

Tahap perancangan merupakan proses yang dilakukan untuk merancang sistem yang akan dibuat berdasarkan analisis kebutuhan yang telah dilakukan. Proses perancangan dimulai dengan merancang arsitektural sistem, *activity diagram*, *sequence diagram*, *class diagram*, basis data, *page flow*, *user experience* dan format konten multimedia.

4.2.1 Perancangan Arsitektural

Perancangan arsitektural sistem menjelaskan mengenai gambaran kinerja sistem secara keseluruhan. Berikut ini merupakan rancangan arsitektural sistem yang akan dijelaskan dalam Gambar 4.3.



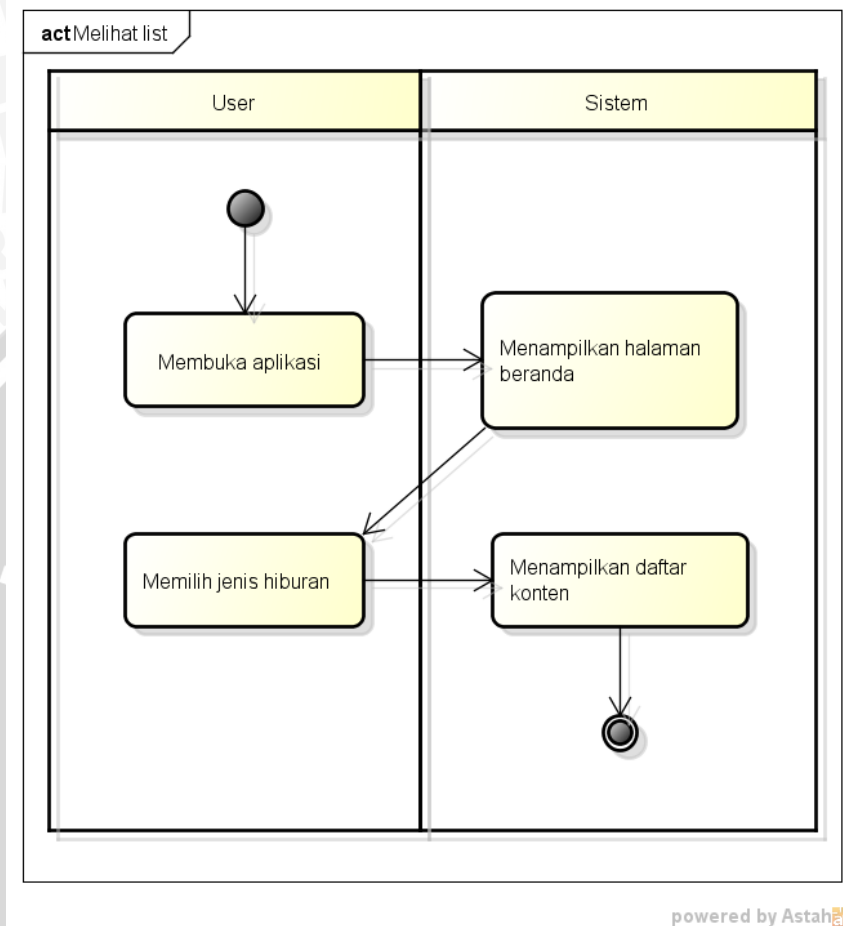
Gambar 4.3 Rancangan arsitektural sistem

Gambar 4.3 menjelaskan kinerja sistem secara keseluruhan. Sistem dimulai dengan aplikasi pada klien melakukan *request web service* pada *web server* sesuai aksi yang dikerjakan oleh *user*. Kemudian *web server* akan melakukan komunikasi ke *database server* menggunakan *sql query*. Setelah *sql query* diproses, maka *database server* akan mengirimkan *query result* ke *web server*. *Web server* akan menerima *query result* kemudian melakukan proses terhadap *query result* tersebut dan mengirimnya pada aplikasi klien dalam format JSON. Data konten yang ada pada *server* dapat ditambah, diubah, ataupun dihapus oleh admin.

4.2.2 Perancangan Activity Diagram

Activity diagram merupakan diagram yang digunakan untuk memodelkan interaksi antara aktor yang ada pada *use case* dengan sistem. Pada *activity*

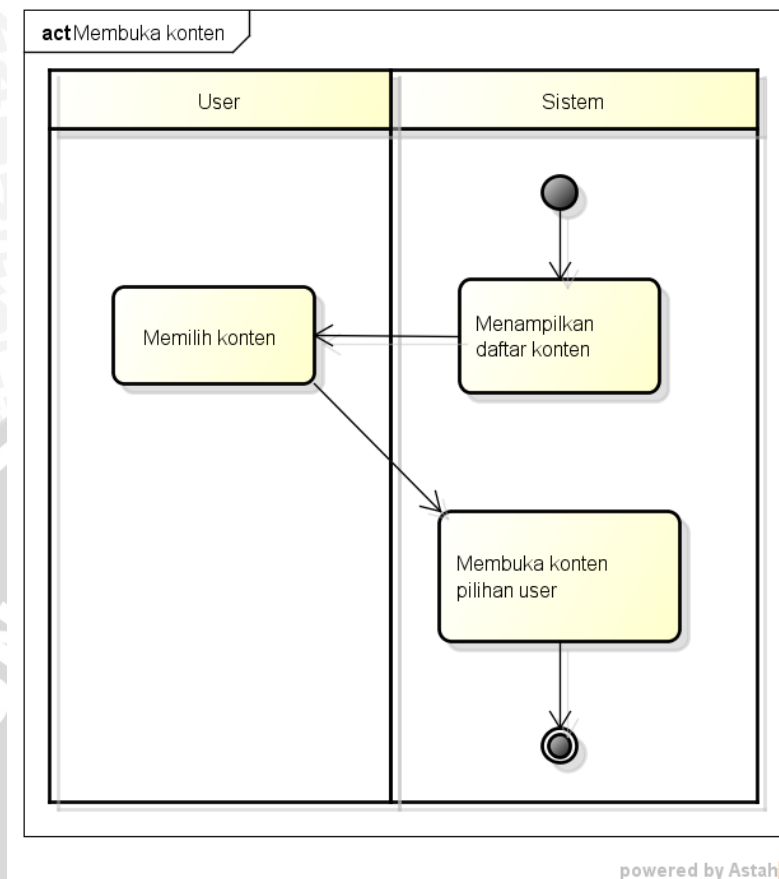
diagram terdapat dua panel yaitu panel User dan Sistem. *Activity diagram* menggambarkan alur kerja sistem pada tiap-tiap *use case* secara langkah demi langkah.



Gambar 4.4 Activity diagram melihat list

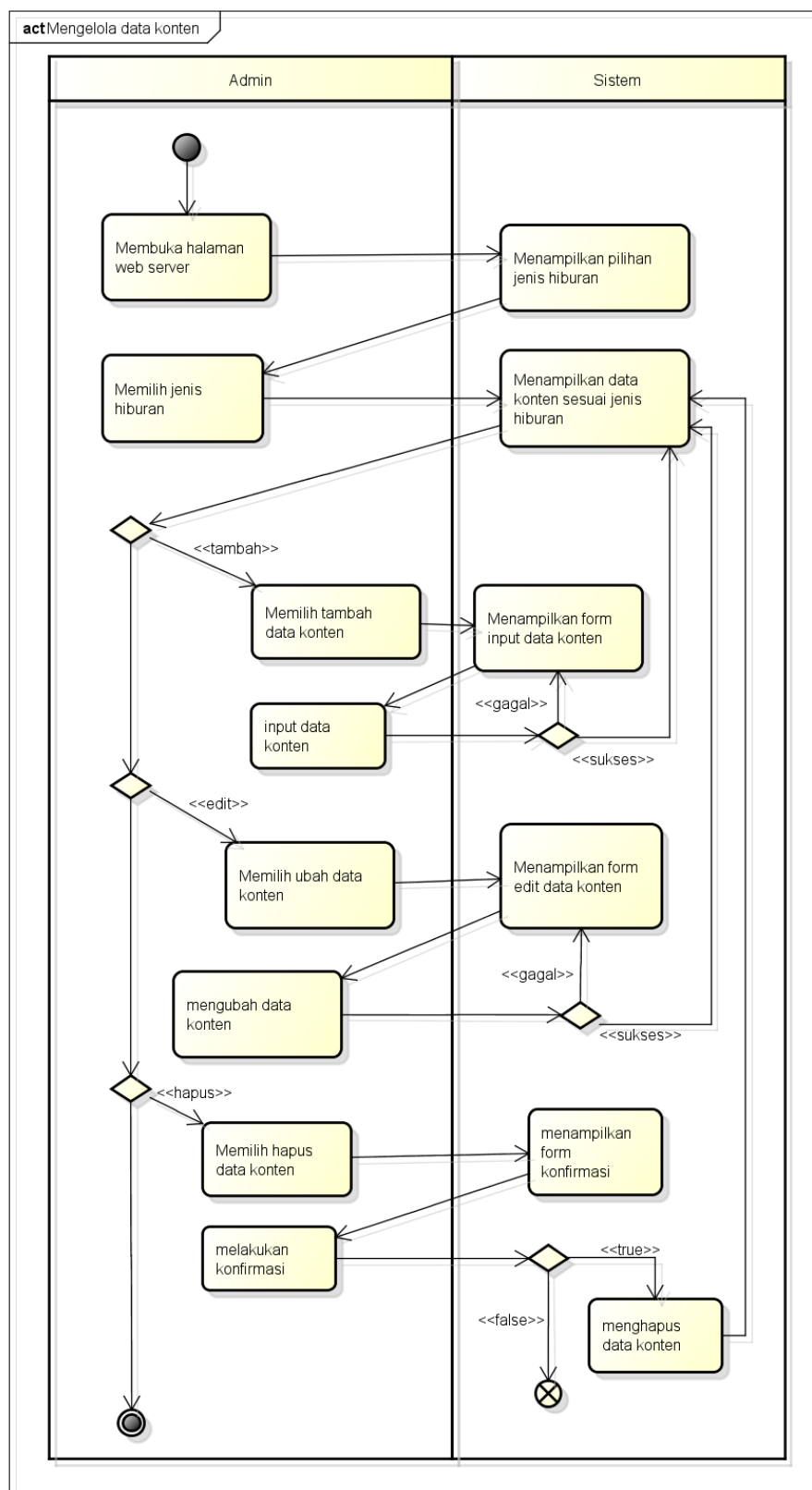
Gambar 4.4 menjelaskan aktivitas yang dilakukan oleh aktor *user* dan aktivitas pada sistem dalam *use case* melihat *list*. Deskripsi *activity diagram* sesuai dengan skenario *use case* melihat *list* yang ada dalam Tabel 4.3. Alurnya adalah sebagai berikut: *user* membuka aplikasi; kemudian sistem akan menampilkan halaman beranda yang berisi pilihan jenis-jenis hiburan yang berbeda yakni film, musik, buku, dan majalah; *user* memilih jenis hiburan yang diinginkan; sistem akan menampilkan daftar konten sesuai jenis hiburan yang dipilih.

Gambar 4.5 menjelaskan aktivitas yang dilakukan oleh aktor *user* dan aktivitas pada sistem dalam *use case* membuka konten. Deskripsi *activity diagram* sesuai dengan skenario *use case* membuka konten yang ada dalam Tabel 4.4. Alurnya adalah sebagai berikut: sistem menampilkan daftar konten sesuai jenis hiburan yang dipilih sebelumnya; *user* memilih konten yang disediakan oleh sistem; sistem membuka konten sesuai dengan pilihan *user*.



Gambar 4.5 Activity diagram membuka konten

Gambar 4.6 menjelaskan aktivitas yang dilakukan oleh aktor admin dan aktivitas pada sistem dalam *use case* mengelola. Deskripsi *activity diagram* sesuai dengan skenario *use case* membuka konten yang ada dalam Tabel 4.5. Alurnya adalah sebagai berikut: admin membuka halaman *web server*; sistem akan menampilkan pilihan untuk jenis hiburan; admin memilih jenis hiburan yang akan dikelola; sistem menampilkan data konten sesuai jenis hiburan; admin memilih pengelolaan data konten yakni tambah, ubah, dan hapus. Jika admin memilih tambah data konten, maka sistem akan menampilkan *form* untuk *input* data kemudian admin dapat melakukan penambahan data konten, sistem akan melakukan validasi terhadap *input* yang dilakukan oleh admin. Format konten yang telah dikirim akan diperiksa yakni format .mkv, .mp4 untuk jenis konten *movie*; format .mp3 untuk jenis konten *music*; format .pdf untuk jenis konten *book* dan *magazine*. Jika admin memilih ubah data konten, maka sistem akan menampilkan *form* untuk mengubah data konten kemudian admin dapat melakukan pengubahan data konten, sistem akan melakukan validasi input yang telah dikirim oleh admin kemudian dilanjutkan ke model untuk dieksekusi pada *database*. Jika admin memilih hapus data konten, maka akan muncul *form* konfirmasi terhadap penghapusan data konten, kemudian data akan terhapus.

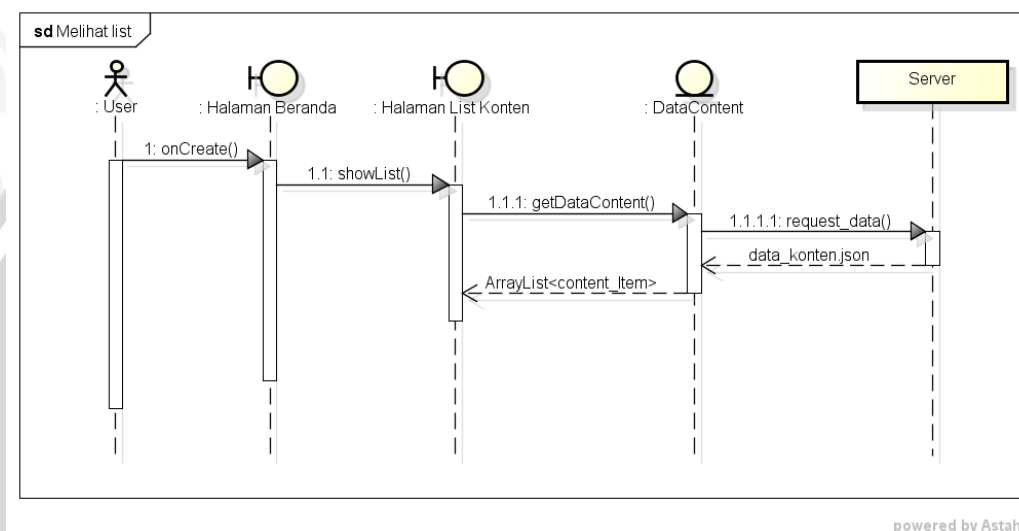


powered by Astah

Gambar 4.6 Activity diagram mengelola data konten

4.2.3 Perancangan Sequence Diagram

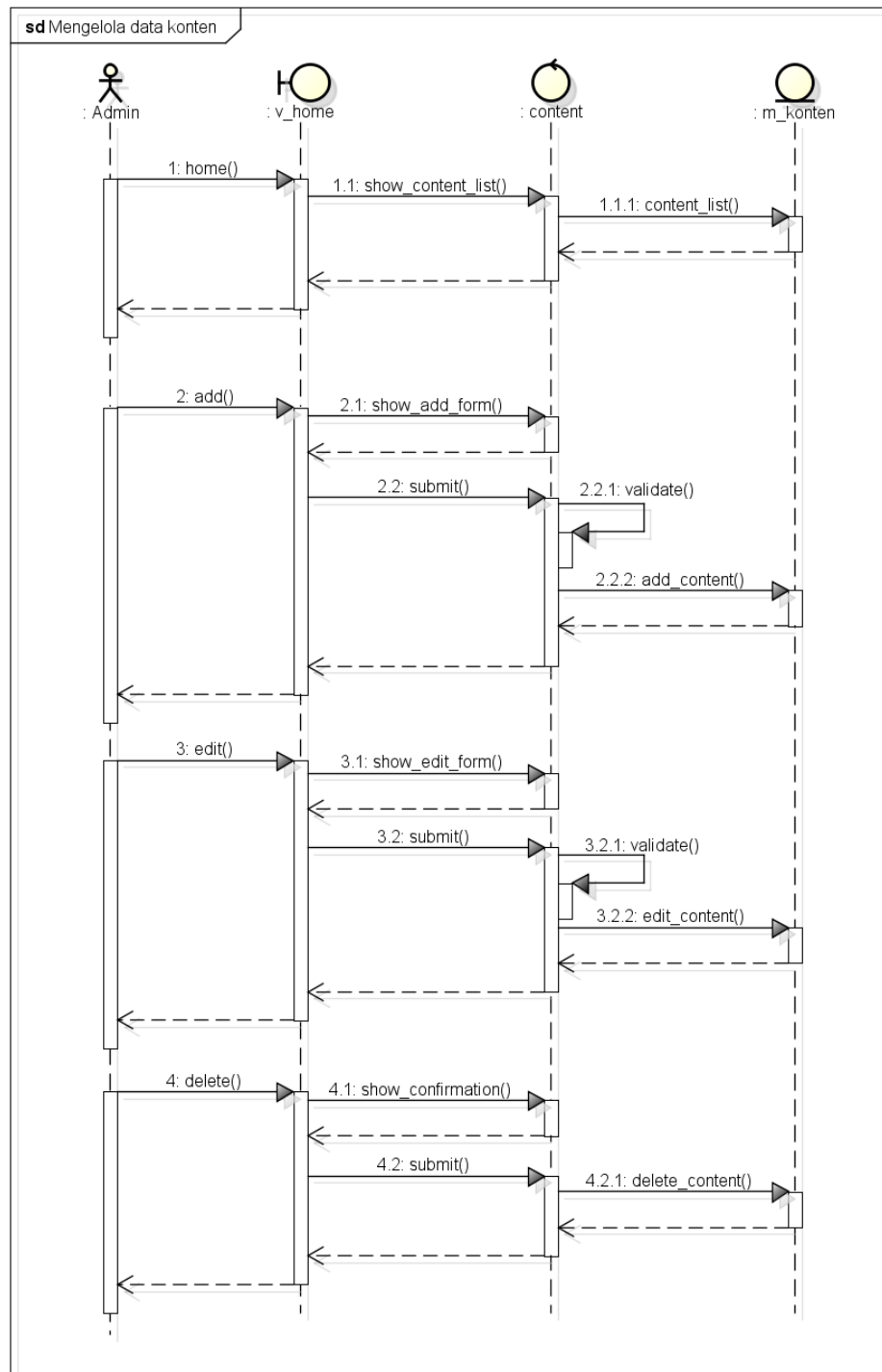
Sequence diagram merupakan gambaran interaksi antar objek dalam bentuk urutan pertukaran pesan terhadap waktu dalam sistem, baik pada klien maupun *server*. *Sequence diagram* digunakan untuk merepresentasikan skenario kegiatan yang ada dalam *use case* dalam urutan langkah-langkah aksi dan balasan dari tiap pesan antar objek. Gambar 4.6 merupakan *sequence diagram* ketika *user* melihat *list* konten sesuai dengan jenis hiburan yang dipilih. Gambar 4.7 merupakan *sequence diagram* ketika *user* membuka konten. Gambar 4.8 merupakan *sequence diagram* ketika admin mengelola data konten.



Gambar 4.7 Sequence diagram melihat list

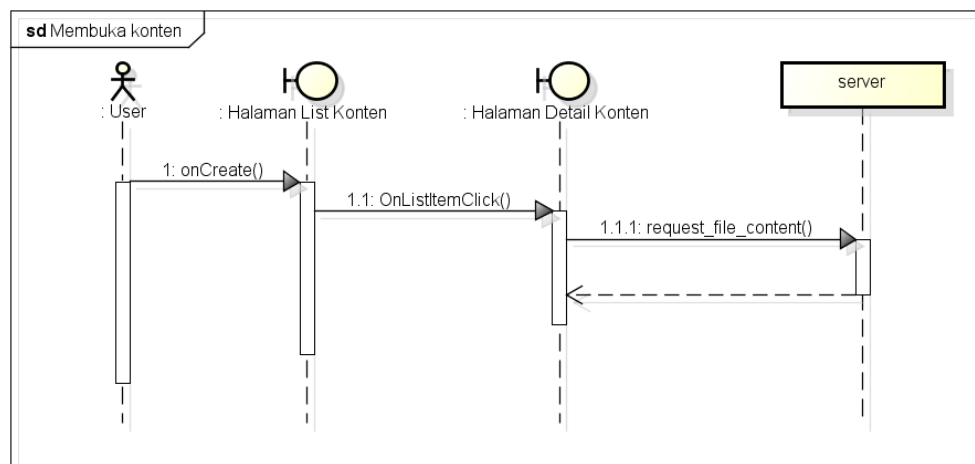
Gambar 4.7 menjelaskan mengenai interaksi antar objek dalam proses *user* melihat *list*. Proses dimulai dengan *user* membuka aplikasi kemudian memilih jenis hiburan pada halaman beranda. Kemudian akan masuk pada halaman *list* konten. Aplikasi akan melakukan *request* data konten melalui kelas *DataContent* ke *server*. *Server* akan mengirim balasan berupa data konten dalam format json. Aplikasi akan melakukan *decode* dan memasukkan dalam bentuk *arraylist* kemudian menampilkannya pada halaman *list* konten.

Gambar 4.9 menjelaskan mengenai interaksi antar objek dalam proses *user* membuka konten. Proses dimulai dengan *user* melihat *list* yang ada pada halaman *list* konten, kemudian memilih konten yang tersedia dengan cara melakukan klik pada konten yang diinginkan. Kemudian aplikasi akan menampilkan detail konten yang telah dipilih serta melakukan *request file* ke *server*. *Server* akan memberikan *file* kemudian aplikasi akan membuka *file* tersebut.



powered by Astah

Gambar 4.8 Sequence diagram mengelola data konten



powered by Astah

Gambar 4.9 Sequence diagram membuka konten

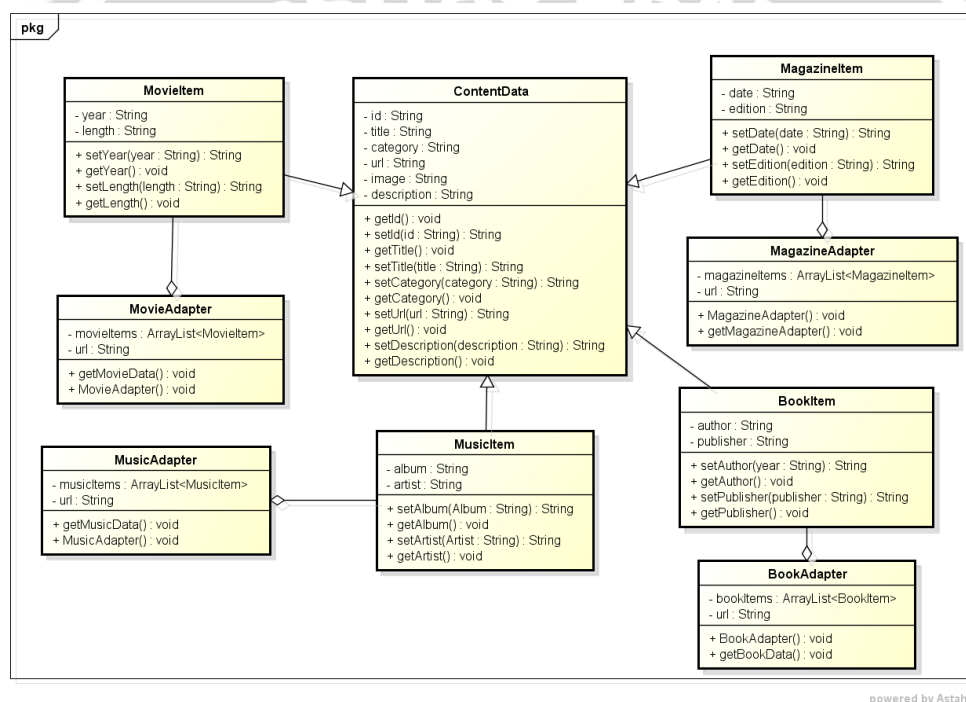
Gambar 4.8 menjelaskan mengenai interaksi antar objek dalam proses mengelola data konten. Proses dimulai dengan admin membuka halaman *web server*, kemudian *server* akan menampilkan daftar data konten sesuai dengan jenis hiburan. Admin dapat melakukan penambahan, pengubahan, atau penghapusan data konten. Penambahan data konten dimulai dengan admin memilih tambah data konten, kemudian *v_home* akan menampilkan *form* untuk melakukan penambahan data konten. Setelah admin mengisi *form* kemudian melakukan *submit*. *Class* konten akan melakukan validasi pada *form* yang telah dikirim, kemudian dilanjutkan ke *m_konten* untuk proses penambahan pada *database*. Pengubahan data konten dimulai dengan admin memilih ubah data konten, kemudian *v_home* akan menampilkan *form* untuk melakukan pengubahan data konten. Setelah admin mengisi *form* kemudian melakukan *submit*. *Class* konten akan melakukan validasi pada *form* yang telah dikirim, kemudian dilanjutkan ke *m_konten* untuk proses pengubahan pada *database*. Penghapusan data konten dimulai dengan admin memilih hapus konten, kemudian akan muncul *form* untuk konfirmasi penghapusan konten. Setelah melakukan *submit*, *class* konten akan melanjutkan ke *m_konten* untuk melakukan penghapusan pada *database*.

4.2.4 Perancangan Class Diagram

Perancangan *class* diagram memberikan gambaran mengenai pemodelan elemen-elemen *class* yang membentuk sebuah aplikasi. *Class* dapat diperoleh dari proses menganalisis secara detail terhadap *use case*. *Class diagram* merupakan sebuah gambaran mengenai sistem beserta aktivitas-aktivitas yang ada didalamnya. *Class diagram* juga menggambarkan relasi-relasi antar aktivitas yang ada dalam sistem. Gambar 4.10 dan Gambar 4.11 merupakan perancangan *class diagram* yang akan digunakan dalam pembuatan aplikasi *entertainment access terminal* pada tablet Android.

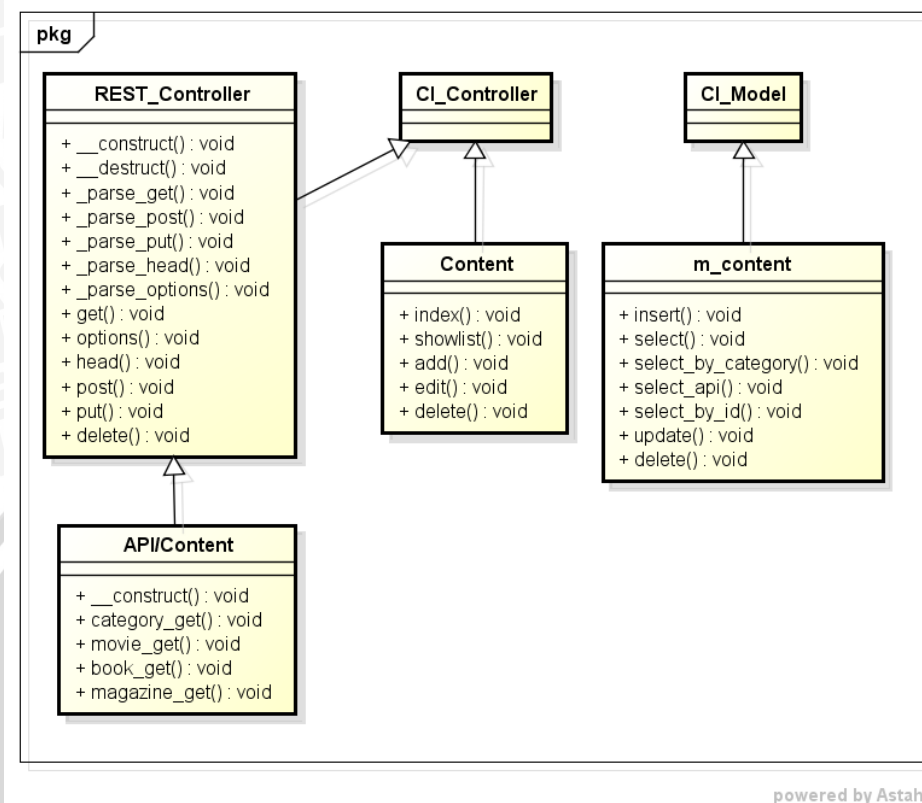
Gambar 4.10 menjelaskan mengenai *class-class* yang terdapat pada aplikasi *entertainment access terminal* pada tablet Android. *Class* *ContentData* merupakan

class yang berhubungan dengan daftar data konten secara umum. *Class* *ContentData* merupakan generalisasi dari *class* *MovieItem*, *class* *MusicItem*, *class* *BookItem*, dan *class* *MagazineItem*. *Class* *MovieItem* merupakan *class* yang digunakan untuk merepresentasikan konten secara individual yang bertipe *movie*. *Class* *MusicItem* merupakan *class* yang digunakan untuk merepresentasikan konten secara individual yang bertipe *music*. *Class* *BookItem* merupakan *class* yang digunakan untuk merepresentasikan konten secara individual yang bertipe *book*. *Class* *MagazineItem* merupakan *class* yang digunakan untuk merepresentasikan konten secara individual yang bertipe *magazine*. *Class* *MovieAdapter*, *MusicAdapter*, *BookAdapter*, dan *MagazineAdapter* merupakan kelas yang berfungsi untuk melakukan *request* konten, kemudian menyimpan balasan dari *server* ke dalam bentuk *arraylist*



Gambar 4.10 Class diagram aplikasi klien

Gambar 4.11 menjelaskan mengenai kelas-kelas yang ada pada *server*. Diagram kelas pada *server* terdapat empat kelas utama yakni tiga kelas *controller* dan satu kelas *model*. Kelas *controller* yang ada pada *server* yaitu: *REST_Controller* yang berguna sebagai mesin utama dari *web service* pada *server*, *API/Content* yang berguna sebagai kelas yang menangani jenis-jenis *request* yang masuk ke *server*, dan kelas *Content* yang berguna untuk melakukan pengelolaan data konten yang terdapat pada *server*. Kelas *model* yang ada pada *server* adalah kelas *m_content* yang berguna untuk melakukan interaksi antara *server* dengan *database*.

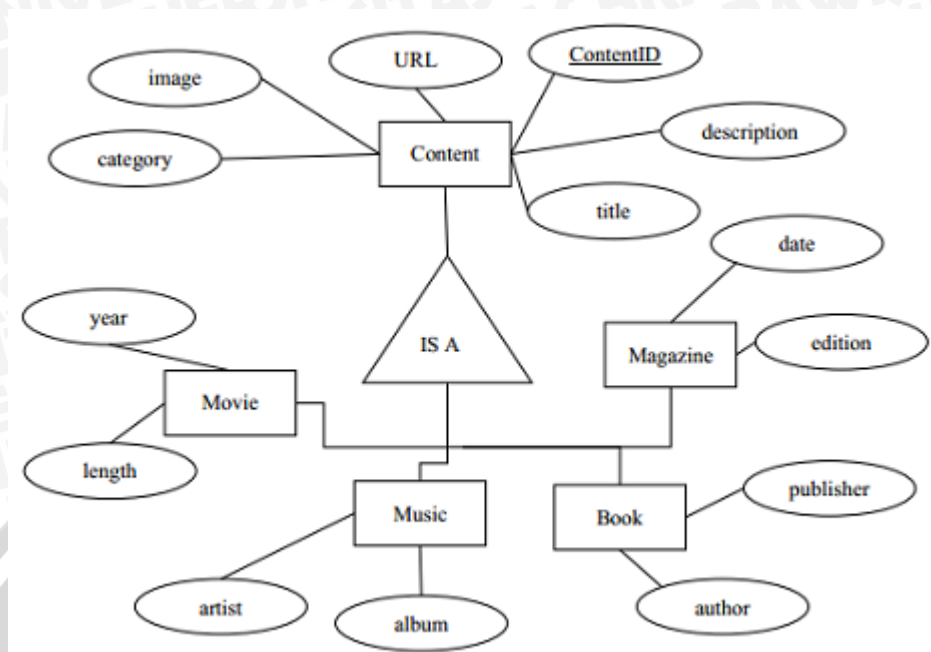


Gambar 4.11 Class Diagram server

Model data logis dalam Gambar 4.12 merepresentasikan lima Tabel yaitu Tabel content untuk menyimpan data konten secara umum, Tabel movie untuk menyimpan data konten yang berjenis film, Tabel music untuk menyimpan data konten yang berjenis musik, Tabel book untuk menyimpan data konten yang berjenis buku, Tabel magazine untuk menyimpan data konten yang berjenis majalah. Relasi IS-A merupakan hubungan spesialisasi, Tabel content dispesialisasikan menjadi empat Tabel lainnya yakni Tabel movie, Tabel music, Tabel book, dan Tabel magazine.

4.2.5 Perancangan Page Flow

Pada perancangan *page flow* dijelaskan mengenai alur antarmuka yang terdapat dalam aplikasi. Perancangan *page flow* dijelaskan dalam Gambar 4.15. *Page flow* aplikasi dimulai pada halaman utama yakni halaman *home* yang terdapat empat tombol yang berfungsi sebagai pilihan tipe hiburan yakni *movie*, *music*, *book*, dan *magazine*. Tombol-tombol tersebut jika ditekan akan membuka halaman baru yakni halaman *content list*. Pada halaman *content list*, terdapat pilihan judul konten sesuai dengan tipe yang telah dipilih pada halaman *home*. Jika judul ditekan maka, akan muncul deskripsi mengenai konten yang dipilih pada panel detail. Pada panel detail terdapat tombol *Open file* yang berfungsi untuk membuka file konten yang dipilih.



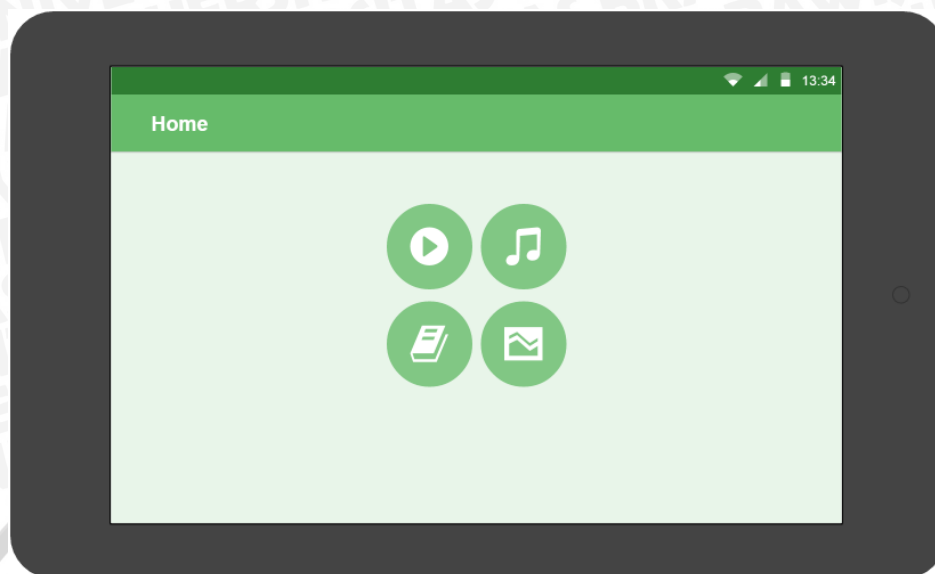
Gambar 4.12 Model data logis

4.2.6 Perancangan User Experience

Perancangan *user experience* dibutuhkan sebagai acuan dalam membuat tampilan (*user interface*) pada aplikasi. *User experience* yang dibuat didasarkan pada aplikasi yang ada pada sistem hiburan pada pesawat. Perancangan *user experience* dilakukan dengan cara membuat desain *mockup* sesuai dengan halaman aplikasi yang akan dibuat dengan menggunakan acuan yang ada pada desain aplikasi hiburan pada pesawat.

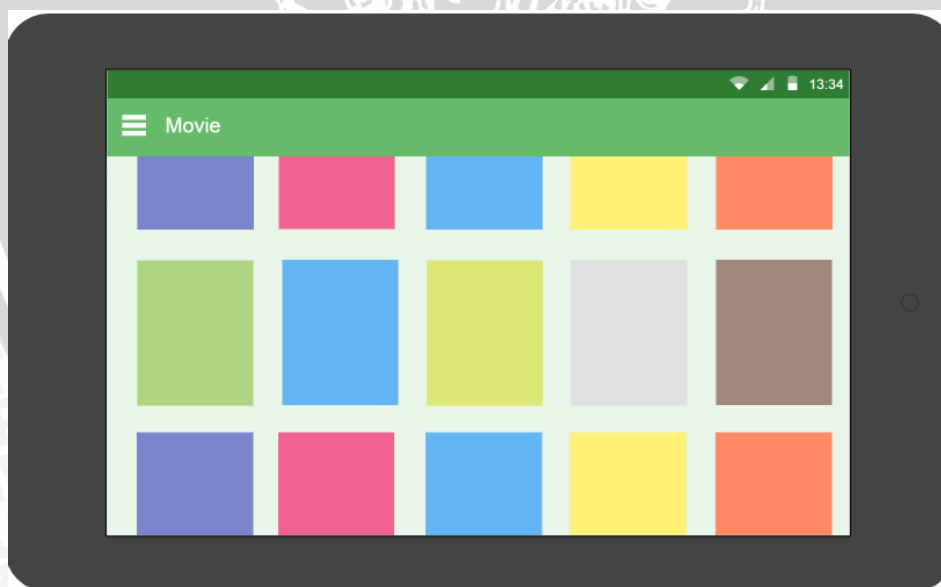
Gambar 4.13 merupakan desain *user experience* pada halaman home. Pada halaman home terdapat empat tombol menu yang digunakan untuk navigasi menuju ke halaman daftar konten sesuai dengan jenis hiburan yang dipilih. Desain *user experience* halaman home termasuk dalam skenario *use case* melihat list yang dijelaskan pada Tabel 4.4. Pada skenario *use case* melihat list, halaman yang diakses pertama kali oleh pengguna adalah halaman home.

Gambar 4.14 merupakan desain *user experience* pada halaman daftar konten. Pada halaman daftar konten, masing-masing konten direpresentasikan dalam bentuk *image* dan *title* yang membentuk sebuah *gridview* seperti dalam Gambar 4.14. Desain *user experience* halaman daftar konten termasuk dalam skenario *use case* melihat list yang dijelaskan pada Tabel 4.5. Pada skenario *use case* melihat list, halaman yang digunakan untuk menampilkan daftar konten adalah halaman daftar konten.

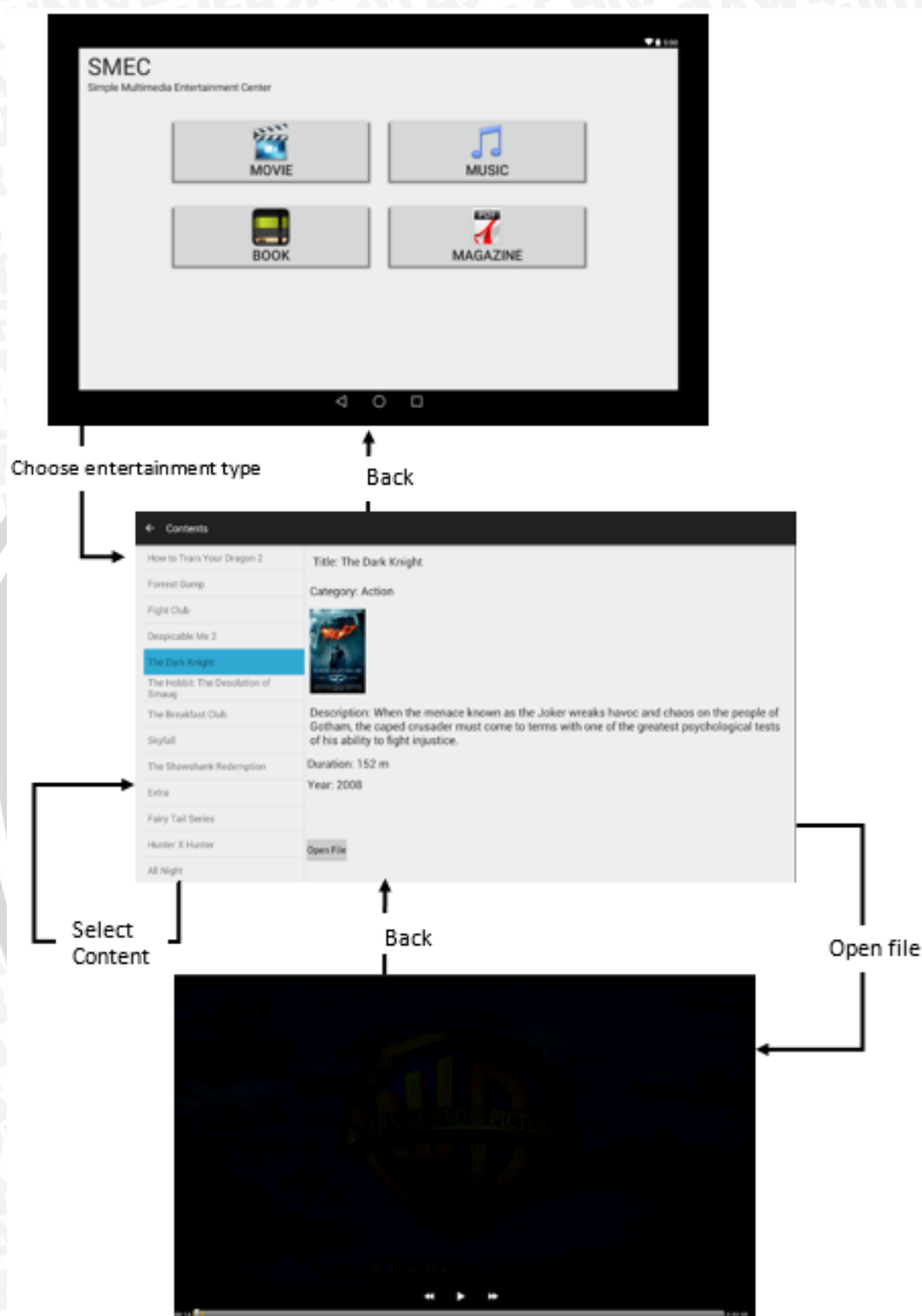


Gambar 4.13 Desain user experience halaman home

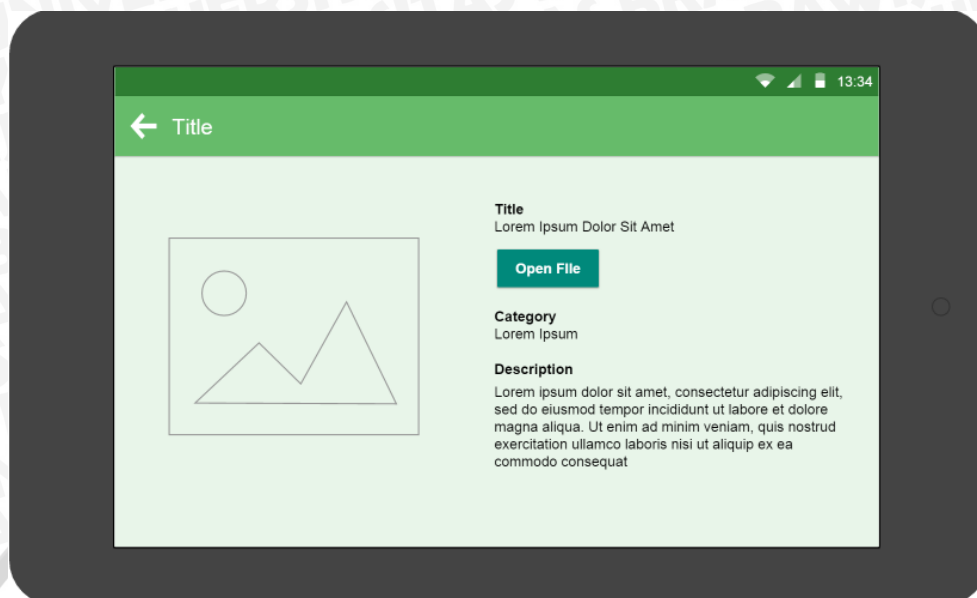
Gambar 4.16 merupakan desain *user experience* halaman detail konten. Pada halaman detail konten, konten dideskripsikan sesuai dengan data yang diterima dari *server* berupa gambar dan kata-kata. Terdapat sebuah tombol yang berfungsi sebagai pembuka konten sesuai dengan format konten yang dipilih. Desain *user experience* halaman *detail* konten termasuk dalam skenario *use case* membuka konten yang dijelaskan pada Tabel 4.6.



Gambar 4.14 Desain user experience halaman daftar konten



Gambar 4.15 Page flow aplikasi



Gambar 4.16 Desain user experience halaman *detail* konten

4.2.7 Multimedia Content Format

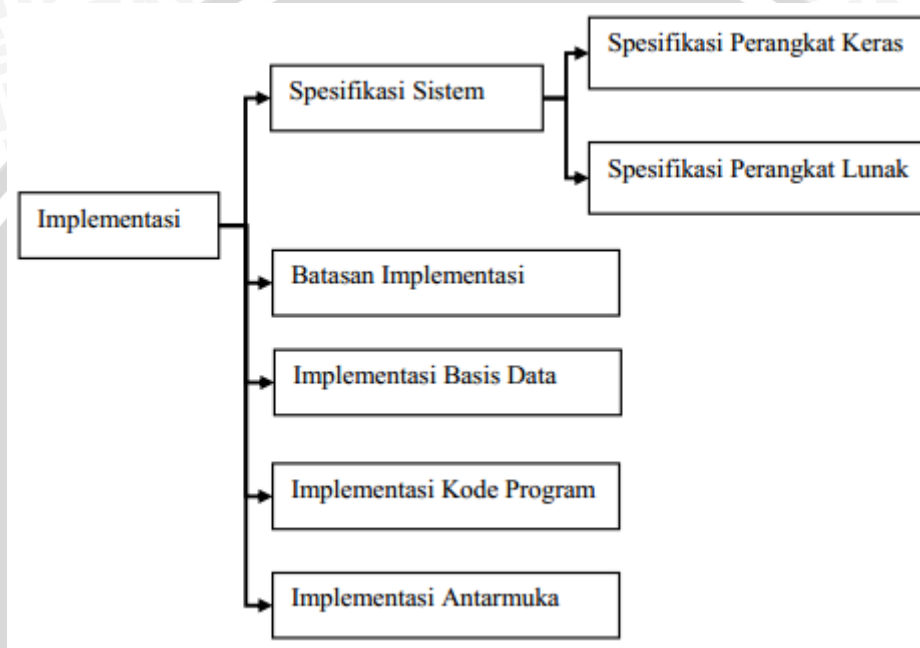
Perancangan format konten multimedia merupakan perancangan mengenai format-format konten multimedia yang dapat dibuka oleh aplikasi *entertainment access terminal*. Pada aplikasi terdapat empat buah tipe konten multimedia yakni *movie*, *music*, *book*, dan *magazine*. Format konten masing-masing tipe hiburan dijelaskan pada Tabel 4.7.

Tabel 4.7 Format konten multimedia

No	Tipe Konten	Format
1	<i>Movie</i>	MKV, MP4
2	<i>Music</i>	MP3
3	<i>Book</i>	PDF
4	<i>Magazine</i>	PDF

BAB 5 IMPLEMENTASI

Bab ini menjelaskan tentang implementasi pada aplikasi *entertainment access terminal* pada tablet Android. Implementasi didasarkan pada hasil yang didapat dari proses analisis kebutuhan dan perancangan aplikasi. Pembahasan terdiri dari penjelasan tentang spesifikasi sistem, batasan implementasi, implementasi basis data, implementasi kode program, dan implementasi antarmuka. Struktur bab implementasi ditunjukkan oleh Gambar 5.1.



Gambar 5.1 Struktur bab implementasi

5.1 Spesifikasi Sistem

Implementasi rancang bangun aplikasi *entertainment access terminal* pada tablet Android didasarkan pada hasil dari tahap analisis kebutuhan dan tahap perancangan yang berupa *activity diagram*, *sequence diagram*, *class diagram*, diagram logis *database*, *page flow* aplikasi, dan *user experience* aplikasi. Spesifikasi sistem yang digunakan untuk implementasi aplikasi dibagi menjadi dua yaitu spesifikasi perangkat lunak dan spesifikasi perangkat keras.

5.1.1 Spesifikasi Perangkat Keras

Perangkat keras yang digunakan dalam proses pengembangan sistem terbagi menjadi dua yaitu perangkat keras untuk *server* dan perangkat keras untuk klien. Perangkat keras yang digunakan sebagai *server* berupa *notebook* dengan spesifikasi yang dijelaskan dalam Tabel 5.1. Perangkat keras yang digunakan sebagai klien berupa tablet dengan spesifikasi perangkat keras yang dijelaskan dalam Tabel 5.2.

Tabel 5.1 Spesifikasi perangkat keras server

Perangkat Keras	Spesifikasi
Type	ASUS X455LD
Processor	Intel Core i3 @1.90 GHz
Memory	4.00 GB RAM

Tabel 5.2 Spesifikasi perangkat keras klien

Perangkat Keras	Spesifikasi
Type	Samsung Galaxy Tab 3 8.0
Processor	dual-core 1.5GHz
Memory	1.5 GB RAM
Layar	TFT <i>capacitive touchscreen</i> , 16M colors, 8.0 inches, 800 x 1280 pixels

5.1.2 Spesifikasi Perangkat Lunak

Spesifikasi perangkat lunak yang digunakan dalam proses pengembangan *server* dan klien dijelaskan pada Tabel 5.3 dan Tabel 5.4. Spesifikasi perangkat lunak server menggunakan server apache yang telah terpasang pada sistem operasi Windows 8.1. Spesifikasi perangkat lunak klien menggunakan sistem operasi Android.

Tabel 5.3 Spesifikasi perangkat lunak server

Komponen	Spesifikasi
OS	Windows 8.1 64-bit
Bahasa Pemrograman	HTML, PHP, CSS
Server	Apache

Tabel 5.4 Spesifikasi perangkat lunak klien

Komponen	Spesifikasi
OS	Android OS v4.2.2
Bahasa Pemrograman	JAVA, XML
IDE (Integrated Development Environment)	Android Studio 1.2.2

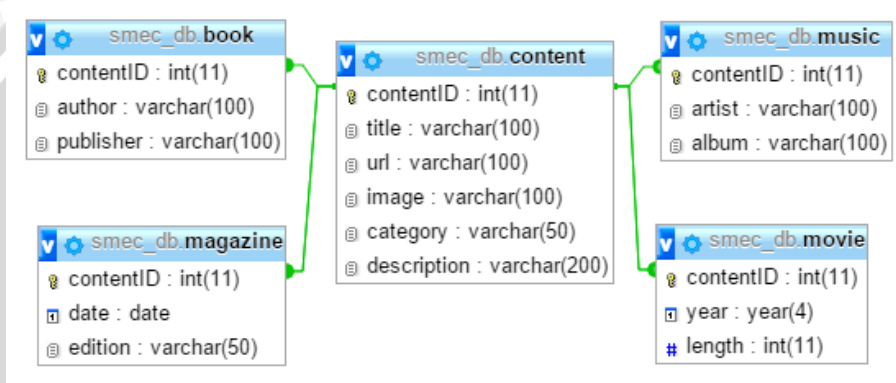
5.2 Batasan Implementasi

Batasan-batasan dalam implementasi rancang bangun aplikasi *entertainment access terminal* pada tablet Android adalah sebagai berikut:

1. Database Management System pada server menggunakan MySQL phpMyAdmin.
2. Perangkat lunak merupakan aplikasi Android *mobile native*.

5.3 Implementasi Basis Data

Implementasi basis data yang dilakukan berdasar pada model data logis dalam Gambar 4.12 menjadi. Implementasi basis data menggunakan MySQL dengan model data fisik sesuai dalam Gambar 5.2. Pada model data logis dalam Gambar 4.12 terdapat relasi IS-A. Relasi IS-A diimplementasikan dengan cara memberikan atribut *primary key* yang ada pada tabel yang bersifat *general* yaitu tabel content ke tabel yang bersifat spesifik yaitu movie, music, book, dan magazine.



Gambar 5.2 Implementasi basis data

a. Tabel content

Tabel 5.5 merupakan deskripsi dari tabel content yang digunakan untuk menyimpan keterangan konten secara umum. Pada tabel content terdapat enam atribut yaitu contentID, title, url, image, category, dan description. Atribut contentID merupakan *primary key* dari tabel content dan berguna untuk menyimpan id unik tiap konten. Atribut title berguna untuk menyimpan judul dari konten. Atribut url berguna untuk menyimpan lokasi file dari konten pada *server*. Atribut image berguna untuk menyimpan lokasi gambar dari konten pada *server*. Contoh nilai dari masing-masing atribut dijelaskan pada Tabel 5.5.

b. Tabel movie

Tabel 5.6 merupakan deskripsi dari tabel movie yang digunakan untuk menyimpan keterangan konten berjenis film. Pada tabel movie terdapat tiga atribut yaitu contenID, year, dan length. Atribut contentID merupakan *foreign key* dari tabel content sekaligus merupakan *primary key* dari tabel movie. Atribut year digunakan untuk menyimpan keterangan tahun pada konten. Atribut length digunakan untuk menyimpan keterangan durasi film dalam satuan menit. Contoh nilai dari masing-masing atribut pada tabel movie dijelaskan pada Tabel 5.6.

Tabel 5.5 Contoh nilai tiap atribut pada tabel content

Atribut	Tipe	Contoh Nilai
contentID	integer	32
title	varchar	The Dark Knight
url	varchar	res/movie/150413The.Dark_.Knight_.2008_.mkv
image	varchar	res/image/150413the_dark_knight_-_cover.jpg
category	varchar	Action
description	varchar	When the menace known as the Joker wreaks havoc and chaos on the people of Gotham, the caped crusader must come to terms with one of the greatest psychological tests of his ability to fight injustice.

Tabel 5.6 Deskripsi tabel movie

Atribut	Tipe	Contoh Nilai
contentID	integer	32
year	year	2008
length	integer	152

c. Tabel music

Tabel 5.7 merupakan deskripsi dari tabel music yang digunakan untuk menyimpan keterangan konten berjenis musik. Pada tabel music terdapat tiga atribut yaitu contentID, artist, dan album. Atribut contentID merupakan *foreign key* dari tabel content sekaligus merupakan *primary key* dari tabel music. Atribut artist digunakan untuk menyimpan keterangan *artist*. Atribut album digunakan untuk menyimpan keterangan album. Contoh nilai dari masing-masing atribut pada tabel music dijelaskan pada Tabel 5.7.

Tabel 5.7 Contoh nilai tiap atribut pada tabel music

Atribut	Tipe	Contoh Nilai
contentID	integer	4
artist	varchar	Lenka
album	varchar	Lenka

d. Tabel book

Tabel 5.6 merupakan deskripsi dari tabel book yang digunakan untuk menyimpan keterangan konten berjenis buku. Pada tabel book terdapat tiga atribut yaitu contentID, author, dan publisher. Atribut contentID merupakan *foreign key* dari tabel content sekaligus merupakan *primary key* dari tabel book.

Atribut author digunakan untuk menyimpan keterangan penulis buku. Atribut publisher digunakan untuk menyimpan keterangan penerbit buku. Contoh nilai dari masing-masing atribut pada tabel book dijelaskan pada Tabel 5.8.

e. Tabel magazine

Tabel 5.9 merupakan deskripsi tabel magazine yang digunakan untuk menyimpan keterangan konten berjenis majalah. Pada tabel magazine terdapat tiga atribut yaitu contentID, date, dan edition. Atribut contentID merupakan *foreign key* dari tabel content sekaligus merupakan *primary key* dari tabel magazine. Atribut date digunakan untuk menyimpan tanggal majalah. Atribut edition digunakan untuk menyimpan keterangan edisi majalah. Contoh nilai dari masing-masing atribut pada tabel magazine dijelaskan pada Tabel 5.9.

Tabel 5.8 Deskripsi tabel book

Atribut	Tipe	Contoh Nilai
contentID	Integer	57
author	Varchar	Kathleen Hayes
publisher	Varchar	Kathleen Hayes

Tabel 5.9 Deskripsi tabel magazine

Atribut	Tipe	Contoh Nilai
contentID	Integer	45
date	Date	2015-04-15
edition	Varchar	Weekly

5.4 Implementasi Kode Program

Kode program yang dicantumkan pada penulisan skripsi ini merupakan implementasi berdasarkan fitur yang ada pada *use case*. Implementasi kode program didapat dari hasil pada tahap perancangan yang berupa *class diagram* dan *sequence diagram*. Implementasi kode program menggunakan bahasa JAVA pada klien dan bahasa PHP pada server.

5.4.1 Kode Program Mengelola Konten Pada Server

Implementasi kode program mengelola konten pada server dibuat berdasarkan perancangan kelas dalam Gambar 4.11. Kode program mengelola konten dibagi menjadi empat bagian yaitu melihat konten yang dijelaskan pada Kode 5.1, menambah konten yang dijelaskan pada Kode 5.2, mengubah konten yang dijelaskan pada Kode 5.3, dan menghapus konten yang dijelaskan pada Kode 5.4.

```
public function showlist($type='') {
    $this->load->model('m_content');
    $result = $this->m_content->select($type);
    return $result; }

```

Kode 5.1 Kode program melihat konten

Kode 5.1 merupakan kode program pada *server* untuk melihat konten. Kode program melihat konten ditunjukkan dengan fungsi `showlist`. Fungsi `showlist` menerima *input* bertipe string dengan variabel `type`. Fungsi `select` pada model `m_content` digunakan untuk melakukan *read* pada database. *Output* dari fungsi `showlist` berupa hasil keluaran dari fungsi `select` pada model `m_content`.

```
public function add($type='')
{
    $date=date('ymd');
    $data = array('title' => $this->input->post('title'),
        'category'=> $this->input->post('category'),
        'description'=> $this->input->post('description'));
    #image and file upload
    if($_FILES['image']['tmp_name'] != NULL)
    {
        unset($config);
        #upload image
        $configImage['upload_path'] = './res/image/';
        $configImage['allowed_types'] = 'jpg|png';
        $configImage['max_size'] = '1000';
        $configImage['remove_spaces'] = TRUE;
        $configImage['file_name'] = $date.$_FILES['image']['name'];
        $configImage['overwrite'] = FALSE;
        $this->load->library('upload', $configImage);
        $this->upload->initialize($configImage);
        if ( ! $this->upload->do_upload('image')){
            $error = array('error' => $this->upload->display_errors());
        }
        else{
            $dataUpload = $this->upload->data();
            $data['image']= 'res/image/'.$dataUpload['orig_name'];
        }
    }
    if($_FILES['file']['tmp_name'] != NULL)
    {
        unset($config);
        #upload file
        if($type == 'movie'){
            $configFile['upload_path'] = './res/movie/';
            $configFile['allowed_types'] = 'mkv|mp4';
        }
    }
}

```

```

}if($type == 'music'){
    $configFile['upload_path']      = './res/music/';
    $configFile['allowed_types'] = 'mp3';
}if($type == 'book'){
    $configFile['upload_path']      = './res/book/';
    $configFile['allowed_types'] = 'pdf';
}if($type == 'magazine'){
    $configFile['upload_path']      = './res/magazine/';
    $configFile['allowed_types'] = 'pdf';
}
$configFile['max_size']      = '1500000';
$configFile['remove_spaces'] = TRUE;
$configFile['file_name']     = $date.$_FILES['file']['name'];
$configFile['overwrite']     = FALSE;
$this->load->library('upload', $configFile);
$this->upload->initialize($configFile);
if ( ! $this->upload->do_upload('file')){
    $error = array('error' => $this->upload->display_errors());
}
else{
    $dataUpload=$this->upload->data();
    $data['url']='res/'.$type.'/'.$dataUpload['orig_name'];
}
}
#data for entertainment specific type
if($type=='movie'){
    $spec = array('year' => $this->input->post('year'),
        'length'=>$this->input->post('length'));
} elseif ($type=='music') {
    $spec = array('artist' => $this->input->post('artist'),
        'album'=> $this->input->post('album'));
} elseif ($type=='book') {
    $spec = array('author' => $this->input->post('author'),
        'publisher'=> $this->input->post('publisher'));
} elseif ($type=='magazine') {
    $spec = array('date' => date('Y-m-d', strtotime(str_replace('-',
    '/', $this->input->post('date')))),
        'edition'=> $this->input->post('edition'));
}
$this->load->model('m_content'); #model
$result=$this->m_content->insert($data,$type,$spec);
redirect(base_url().'home/show/'.$type);
}

```

Kode 5.2 Kode program menambah konten

Kode 5.2 merupakan kode program pada *server* untuk menambah konten. Kode program untuk menambah konten ditunjukkan oleh fungsi `add`. Fungsi `add` menerima masukan berupa string dengan variabel `type`. Kemudian, *variable* `date` diisi dengan *output* dari fungsi `date` dengan format 'ymd'. Hasil *input* dari form penambahan konten dimasukkan ke dalam array dengan *variable* `data`. *Input* gambar diperiksa ekstensi dan ukurannya serta ditentukan namanya. Kemudian, gambar diunggah ke direktori yang telah ditentukan sesuai dengan tipe masing-masing konten. *Input* file konten diperlakukan sama dengan *input* gambar. Selanjutnya, data yang berhubungan dengan tipe konten disimpan dalam array `spec`. Fungsi `insert` pada model `m_content` dipanggil dengan parameter `data`, `type`, dan `spec` untuk melakukan proses *insert* ke *database*.

```
public function edit($type='', $id='')
{
    #data content
    $data = array('title' => $this->input->post('title'),
        'category'=> $this->input->post('category'),
        'contentID'=> $id,
        'description'=> $this->input->post('description'));
    #data for entertainment specific type
    if($type=='movie'){
        $spec = array('year' => $this->input->post('year'),
            'length'=>$this->input->post('length'));
    } elseif ($type=='music') {
        $spec = array('artist' => $this->input->post('artist'),
            'album'=> $this->input->post('album'));
    }elseif ($type=='book') {
        $spec = array('author' => $this->input->post('author'),
            'publisher'=> $this->input->post('publisher'));
    }elseif ($type=='magazine') {
        $spec = array('date' => date('Y-m-d',
            strtotime(str_replace('-', '/', $this->input->post('date'))),
            'edition'=> $this->input->post('edition'));
    }
    $this->load->model('m_content');
    $result=$this->m_content->update($data,$type,$spec);
    redirect(base_url(). 'home/show/'. $type);
}
```

Kode 5.3 Kode program mengubah konten

Kode 5.3 merupakan kode program pada *server* untuk mengubah konten. Kode program untuk mengubah konten ditunjukkan oleh fungsi `edit`. Fungsi `edit` menerima parameter `type` dan `id`. `Type` digunakan untuk menentukan tipe konten yang akan diubah. Sedangkan `id` digunakan untuk menentukan id dari konten yang akan diubah. Hasil *input* dari *form edit* konten dimasukkan dalam *variable* `data` yang bertipe array. Kemudian Fungsi `update` pada model `m_content` dipanggil dengan parameter `data`, `type`, dan `spec` untuk melakukan proses *update* ke *database*.

Kode 5.4 merupakan kode program pada *server* untuk menghapus konten. Kode program untuk mengubah konten ditunjukkan oleh fungsi *delete*. Fungsi *delete* menerima parameter *type* dan *id*. *Type* digunakan untuk menentukan tipe konten yang akan diubah. Sedangkan *id* digunakan untuk menentukan id dari konten yang akan dihapus. Fungsi *delete* pada model *m_content* dipanggil dengan parameter *type* dan *id* untuk melakukan proses *delete* ke *database*.

```
public function delete($type='', $id='')
{
    $this->load->model('m_content');
    $this->m_content->delete($type, $id);
    redirect(base_url(). 'home/show/'. $type);
}
```

Kode 5.4 Kode program menghapus konten

5.4.2 Kode Program Melihat Daftar Konten Pada Klien

Implementasi kode program melihat daftar konten pada klien dibuat berdasarkan perancangan kelas dalam Gambar 4.10. Kode program melihat daftar konten yang dituliskan hanya konten yang bertipe film. Kode 5.5 adalah implementasi dari kelas *ContentData* yang merupakan kelas konten secara umum. Kode 5.6 adalah implementasi kode program dari kelas *MovieItem* yang merupakan kelas turunan dari kelas *ContentData*. Kode 5.7 adalah implementasi kode program untuk mengambil data dari *server* dan pembuatan objek *MovieItem*.

```
public class ContentData {
    public String id;
    public String title;
    public String category;
    public String image;
    public String url;
    public String description;
    public String getId() {
        return id;
    }
    public void setId(String id) {
        this.id = id;
    }
    public String getTitle() {
        return title;
    }
    public void setTitle(String title) {
        this.title = title;
    }
    public String getCategory() {
        return category;
    }
    public void setCategory(String category) {
        this.category = category;
    }
    public String getImage() {
        return image;
    }
}
```

```

    }
    public void setImage(String image) {
        this.image = image;
    }
    public String getUrl() {
        return url;
    }
    public void setUrl(String url) {
        this.url = url;
    }
    public String getDescription() {
        return description;
    }
    public void setDescription(String description) {
        this.description = description;
    }
}

```

Kode 5.5 Kode program kelas ContentData

Kode 5.5 merupakan kode program untuk kelas ContentData. Pada kelas ContentData terdapat enam *variable* yang masing-masing berpasangan dengan fungsi get dan set. *Variable* tersebut adalah id, title, category, image, url, dan description.

```

public class MovieItem extends ContentData {
    public String year;
    public String length;
    public String getYear() {
        return year;
    }
    public void setYear(String year) {
        this.year = year;
    }
    public String getLength() {
        return length;
    }
    public void setLength(String length) {
        this.length = length;
    }
}

```

Kode 5.6 Kode program kelas MovieItem

Kode 5.6 merupakan kode program untuk kelas MovieItem. Kelas MovieItem merupakan kelas turunan dari kelas ContentData. Terdapat dua *variable* yang masing-masing berpasangan dengan fungsi get dan set. *Variable* tersebut adalah year dan length.

```

protected String doInBackground(String... params) {
    ServiceHandler sh = new ServiceHandler();
    // Making a request to url and getting response
    String jsonStr = sh.makeServiceCall(url, ServiceHandler.GET);
    Log.d("Response: ", "> " + jsonStr);
    if (jsonStr != null) {
        try {
            // Getting JSON Array node
            contentJson = new JSONArray(jsonStr);
            // looping through All category
            for (int i = 0; i < contentJson.length(); i++) {

```

```

        JSONObject c = contentJson.getJSONObject(i);
        MovieItem movie = new MovieItem();
        movie.setId(c.getString("contentID"));
        movie.setTitle(c.getString("title"));
        movie.setCategory(c.getString("category"));
        movie.setImage(server + c.getString("image"));
        movie.setDescription(c.getString("description"));
        movie.setUrl(server + c.getString("url"));
        movie.setYear(c.getString("year"));
        movie.setLength(c.getString("length"));
        movieItems.add(movie);
    }
} catch (JSONException e) {
    e.printStackTrace();
}
} else {
    Log.e("ServiceHandler", "Couldn't get any data from the url");
}

return null;
}

```

Kode 5.7 Kode program mengambil konten bertipe film

Kode 5.7 merupakan kode program untuk mengambil konten bertipe film. Fungsi untuk mengambil konten menggunakan teknik `asyncTask` yakni `doInBackground`. *Input* yang diperlukan adalah URL sesuai *request* ke *server*. Konten diambil dari *server* menggunakan protokol HTTP dengan *method* GET. Hasil dari *request* berupa objek JSON yang kemudian diurai secara berulang sesuai dengan panjang JSON yang diterima. Kemudian objek `movieItems` dibuat dari JSON yang telah diurai.

5.4.3 Kode Program Membuka Konten Pada Klien

Implementasi kode program membuka konten pada klien dibuat berdasarkan perancangan *sequence* diagram yang terdapat dalam Gambar 4.8. Kode program membuka konten dijelaskan pada Kode 5.8. Kode 5.8 merupakan kode program untuk membuka konten. Kode program diawali dengan memeriksa tipe konten yang akan dibuka. Jika tipe konten adalah `movie`, maka objek uri akan dibuat sesuai url yang dimiliki oleh konten tersebut. Kemudian sebuah intent video akan dibuat dengan parameter data dan tipe berupa objek uri dan tipe video. Jika tipe konten adalah `music`, maka objek uri akan dibuat sesuai url yang dimiliki oleh konten tersebut. Kemudian sebuah intent audio akan dibuat dengan parameter data dan tipe berupa objek uri dan tipe audio. Jika tipe konten berupa `book` atau `magazine`, maka akan dibuat intent untuk membuka file bertipe PDF.

```

if (type.equals("movie")) {
    openBtn.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            Uri uri =
            Uri.parse(getIntent().getExtras().getString("url"));

```

```

        Intent intent = new Intent(Intent.ACTION_VIEW, uri);
        intent.setDataAndType(uri, "video/*");
        startActivity(intent);
    }
    });
} else if (type.equals("music")) {
    openBtn.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            Uri myUri =
            Uri.parse(getIntent().getExtras().getString("url"));
            Intent intent = new
            Intent(Android.content.Intent.ACTION_VIEW);
            intent.setDataAndType(myUri, "audio/*");
            startActivity(intent);
        }
    });
} else if (type.equals("book") || type.equals("magazine")) {
    openBtn.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            download(v);
        }
    });
}
}

```

Kode 5.8 Kode program membuka konten

5.5 Implementasi Antarmuka

Pada sub bab implementasi antarmuka akan dijelaskan mengenai antarmuka dari aplikasi *entertainment access terminal* pada tablet Android. Implementasi antarmuka dibuat menggunakan format XML dan berdasar *user experience* pada tahap perancangan.

5.5.1 Implementasi Antarmuka Halaman Home

Implementasi antarmuka halaman home dibuat berdasarkan perancangan *user experience* dalam Gambar 4.13. Implementasi kode XML yang disusun untuk membuat tampilan sesuai dengan perancangan dijelaskan pada Kode 5.9. Gambar 5.3 merupakan hasil antarmuka halaman home.

```

<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:background="#E8F5E9"

```

```
android:gravity="center">
<LinearLayout
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:gravity="center">
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textAppearance="?android:attr/textAppearanceLarge"
        android:text="WELCOME"
        android:id="@+id/textView"
        android:gravity="center"
        android:textSize="50dp"
        android:textStyle="bold"
        android:editable="true" />
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:gravity="center"
        android:orientation="horizontal"
        android:layout_alignParentTop="false">
        <Button
            android:layout_width="150dp"
            android:layout_height="150dp"
            android:text="@string/movies"
            android:textSize="75sp"
            android:textColor="@color/text_btn"
            android:id="@+id/movies"
            android:background="@drawable/roundbackground"
            android:scaleType="fitCenter"
            android:gravity="center"
            android:clickable="true"
            android:layout_centerVertical="true"
            android:layout_centerHorizontal="true"
            android:layout_gravity="center"
            android:layout_margin="5dp" />
        <Button
            android:layout_width="150dp"
            android:layout_height="150dp"
            android:text="@string/musics"
            android:textSize="75sp"
            android:textColor="@color/text_btn"
            android:id="@+id/music"
            android:background="@drawable/roundbackground"
            android:scaleType="fitCenter"
            android:gravity="center"
            android:layout_gravity="center"
            android:clickable="true"
```

```

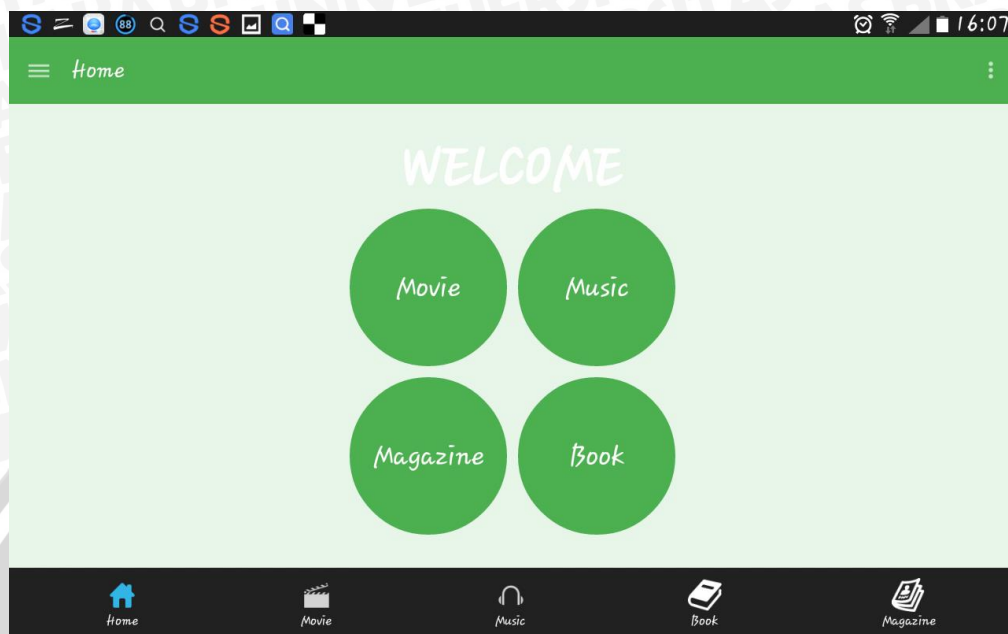
        android:layout_alignLeft="@+id/movies"
        android:layout_alignStart="@+id/movies"
        android:layout_centerVertical="true"
        android:layout_margin="5dp" />
    </LinearLayout>
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="horizontal"
        android:gravity="center"
        android:layout_alignWithParentIfMissing="false">
        <Button
            android:layout_width="150dp"
            android:layout_height="150dp"
            android:text="@string/magz"
            android:textSize="75sp"
            android:textColor="@color/text_btn"
            android:id="@+id/magz"
            android:background="@drawable/roundbackground"
            android:scaleType="fitCenter"
            android:layout_gravity="center"
            android:clickable="true"
            android:gravity="center"
            android:layout_centerVertical="true"
            android:layout_centerHorizontal="true"
            android:layout_margin="5dp" />
        <Button
            android:layout_width="150dp"
            android:layout_height="150dp"
            android:text="@string/books"
            android:textSize="75sp"
            android:textColor="#fff"
            android:id="@+id/books"
            android:background="@drawable/roundbackground"
            android:scaleType="fitCenter"
            android:gravity="center"
            android:clickable="true"
            android:layout_centerVertical="true"
            android:layout_centerHorizontal="true"
            android:layout_gravity="center"
            android:layout_margin="5dp" />
    </LinearLayout>
</LinearLayout>
</RelativeLayout>

```

Kode 5.9 Kode XML tampilan halaman home

Pada halaman home terdapat empat buah tombol menu yang masing-masing akan menuju ke halaman daftar konten sesuai dengan jenis hiburan yang dipilih. Tampilan tombol menu movie dibuat menggunakan kode XML. Tombol dibuat berukuran 150 dp dengan bentuk lingkaran berwarna hijau. Tombol menu hiburan

lain dibuat dengan cara yang sama. Keempat tombol menu hiburan disusun sesuai dengan yang terdapat dalam Gambar 5.3.



Gambar 5.3 Antarmuka halaman home

5.5.2 Implementasi Antarmuka Halaman Daftar Konten

Implementasi antarmuka halaman daftar konten dibuat berdasarkan perancangan *user experience* dalam Gambar 4.14. Antarmuka halaman daftar konten menggunakan susunan *gridview* untuk menampilkan susunan konten. Implementasi kode XML untuk membuat satu tampilan konten dijelaskan pada Kode 5.10. Gambar 5.4 merupakan hasil implementasi antarmuka halaman daftar konten dengan konten berjenis buku.

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android=http://schemas.android.com/apk/res/android
    xmlns:card_view=http://schemas.android.com/apk/res-auto
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">
    <android.support.v7.widget.CardView
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        card_view:contentPadding="3dp"
        card_view:cardCornerRadius="3dp"
        android:layout_margin="10dp">
        <ImageView
            android:id="@+id/img_thumbnail"
            android:layout_width="fill_parent"
            android:layout_height="200dp" />
        <TextView
            android:id="@+id/tv_movie"
```

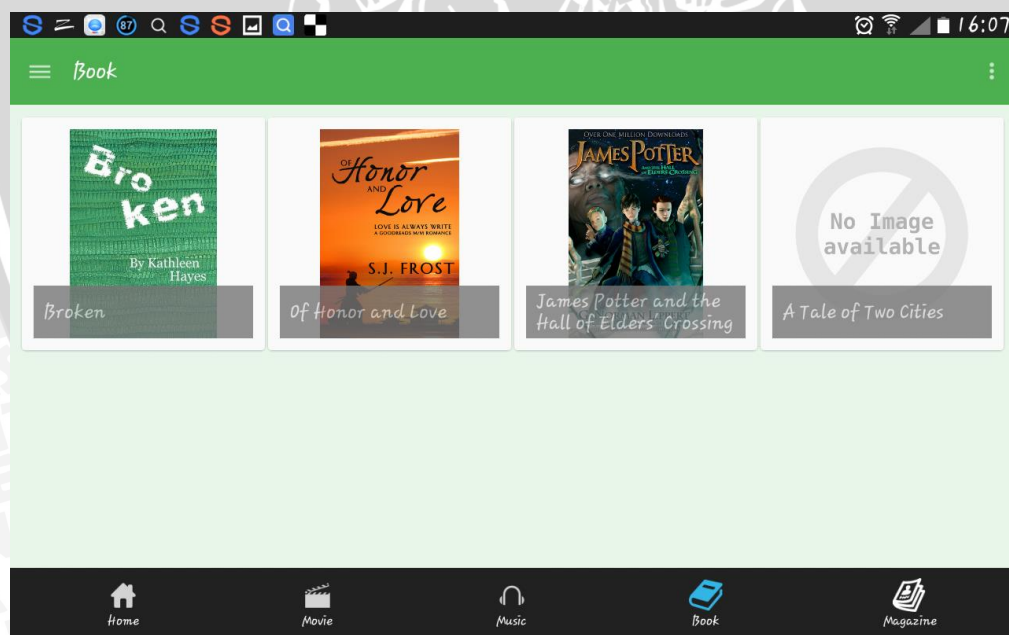
```

        Android:layout_width="fill_parent"
        Android:layout_height="50dp"
        Android:layout_gravity="bottom"
        Android:alpha="0.8"
        Android:background="#757575"
        Android:gravity="center_vertical"
        Android:paddingLeft="5dp"
        Android:paddingRight="5dp"
        Android:text="Test"
        Android:textSize="@dimen/abc_text_size_medium_material" />
    </Android.support.v7.widget.CardView>
</LinearLayout>

```

Kode 5.10 Kode XML satu item dalam *gridview*

Halaman daftar konten merupakan halaman untuk menampilkan konten-konten hiburan dalam bentuk *gridview*. Pembuatan *gridview* menggunakan tampilan *cardview* yang merupakan widget yang telah disediakan oleh Android. Satu *item* dalam *gridview* merupakan representasi dari satu konten. Dalam satu item terdapat sebuah gambar dan teks untuk mengidentifikasi tiap konten. Kode 5.10 merupakan kode XML yang menggambarkan satu *item* pada *gridview*. Kemudian disusun sehingga sesuai dengan tampilan dalam Gambar 5.4.



Gambar 5.4 Antarmuka halaman daftar konten

5.5.3 Implementasi Antarmuka Halaman Detail Konten

Implementasi antarmuka halaman detail konten dibuat berdasarkan perancangan *user experience* dalam Gambar 4.16. Implementasi kode XML untuk

membuat antarmuka halaman detail konten dijelaskan pada Kode 5.11. Gambar 5.5 merupakan hasil antarmuka halaman detail konten.

```
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:background="#E8F5E9" >
    <include
        android:id="@+id/app_bar"
        layout="@layout/app_bar" />
    <LinearLayout
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:orientation="horizontal"
        android:paddingTop="40dp"
        android:gravity="center">
        <ImageView
            android:id="@+id/imageView"
            android:layout_width="fill_parent"
            android:layout_height="400dp"
            android:layout_weight="1"
            android:paddingLeft="50dp"
            android:layout_marginLeft="50dp"
            android:layout_marginRight="50dp"
            android:layout_marginBottom="50dp"
            android:layout_gravity="right" />
        <ScrollView
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:paddingLeft="50dp"
            android:layout_gravity="center_vertical|left">
            <LinearLayout
                android:layout_width="fill_parent"
                android:layout_height="wrap_content"
                android:orientation="vertical">
                <TextView
                    android:layout_width="fill_parent"
                    android:layout_height="wrap_content"
                    android:padding="3dip"
                    android:text="Title"

                    android:textSize="@dimen/abc_text_size_large_material"
                    android:autoText="false"
                    android:textColor="#2E2E2E" />
                <TextView
                    android:layout_width="fill_parent"
                    android:layout_height="wrap_content"
```

```

        Android:padding="10dip"
        Android:text="Lorem Ipsum"
        Android:textSize="@dimen/abc_text_size_body_1_material"
        Android:id="@+id/detail_title"
        Android:textColor="#2E2E2E" />
    <Button
        Android:layout_width="100dp"
        Android:padding="3dip"
        Android:layout_height="wrap_content"
        Android:text="Open File"
        Android:id="@+id/openbutton"
        Android:background="#00796B" />
    <TextView
        Android:layout_width="fill_parent"
        Android:layout_height="wrap_content"
        Android:padding="3dip"
        Android:text="Category"
        Android:textSize="@dimen/abc_text_size_large_material"
        Android:textColor="#2E2E2E" />
    <TextView
        Android:layout_width="fill_parent"
        Android:layout_height="wrap_content"
        Android:padding="10dip"
        Android:text="Lorem"
        Android:textSize="@dimen/abc_text_size_body_1_material"
        Android:id="@+id/detail_category"
        Android:textColor="#2E2E2E" />
    <TextView
        Android:layout_width="fill_parent"
        Android:layout_height="wrap_content"
        Android:padding="3dip"
        Android:text="Description"
        Android:textSize="@dimen/abc_text_size_large_material"
        Android:textColor="#2E2E2E" />

    <TextView
        Android:layout_width="350dp"
        Android:layout_height="wrap_content"
        Android:padding="10dip"
        Android:text="Lorem"
        Android:textSize="@dimen/abc_text_size_body_1_material"
        Android:id="@+id/detail_description"
        Android:textColor="#2E2E2E" />
</LinearLayout>
</ScrollView>

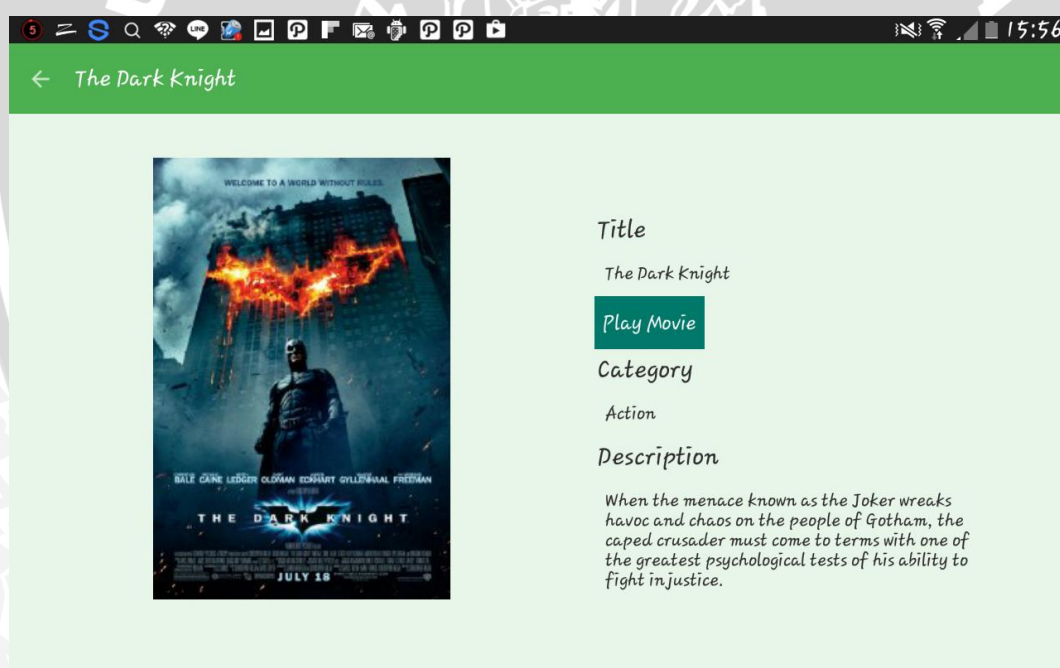
```

```
</LinearLayout>
</LinearLayout>
```

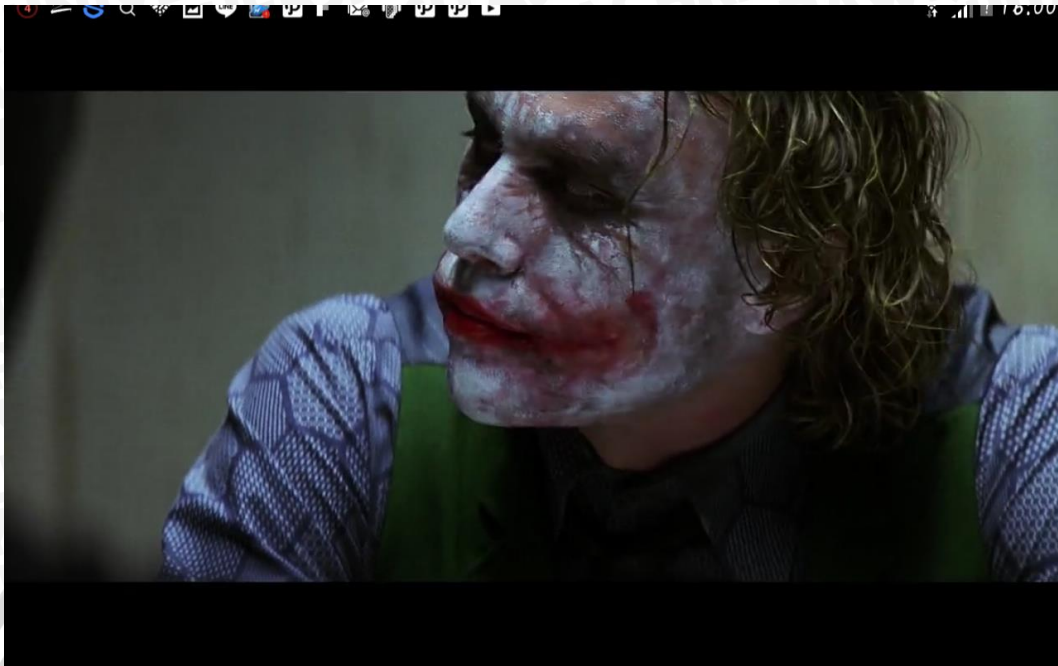
Kode 5.11 Kode XML antarmuka halaman detail konten

Halaman detail konten merupakan halaman yang memuat mengenai detail dari konten yang telah dipilih. Detail dari konten berupa judul konten, kategori, deskripsi konten, dan gambarnya. Terdapat sebuah tombol yang berfungsi untuk membuka konten. Kode XML dari halaman detail konten dijelaskan pada Kode 5.11. Pada halaman detail konten terdapat tiga textview yang nilainya akan berubah sesuai dengan data konten yang dipilih yaitu textview dengan id detail_title, detail_category, dan detail_description. Gambar yang dimuat pada halaman detail konten telah diatur ulang dengan ukuran panjang vertikal maksimal 400 dp.

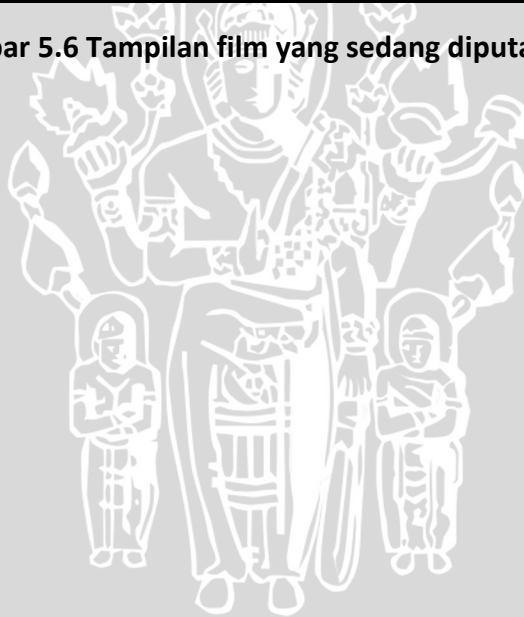
Gambar 5.5 merupakan antarmuka halaman detail konten dengan kode XML sesuai dengan Kode 5.11. Dalam Gambar 5.5 terdapat sebuah tombol yang berfungsi untuk membuka *file*. Ketika tombol ditekan, maka kode program yang ada pada Kode 5.8 akan dijalankan sesuai dengan objek konten yang dipilih. Gambar 5.6 merupakan antarmuka ketika tombol dalam Gambar 5.5 yang bertuliskan 'Play Movie' telah ditekan dan konten yang berupa film telah terbuka.



Gambar 5.5 Antarmuka halaman detail konten



Gambar 5.6 Tampilan film yang sedang diputar



BAB 6 PENGUJIAN

Pada bab ini akan dilakukan tahap pengujian dari aplikasi *entertainment access terminal* pada tablet Android. Pengujian dilakukan dengan tiga metode yaitu pengujian usabilitas, pengujian reliabilitas, dan pengujian fungsionalitas. Pengujian usabilitas merupakan pengujian yang dilakukan untuk mengukur tingkat kemudahan penggunaan sistem oleh pengguna. Pengujian reliabilitas merupakan pengujian yang dilakukan untuk mengukur tingkat konsistensi sistem terhadap banyaknya klien serta *request* data. Pengujian fungsionalitas merupakan pengujian yang dilakukan untuk mengukur tingkat kesesuaian sistem yang telah dibuat dengan kebutuhan fungsional.

6.1 Pengujian Usabilitas

Pengujian usabilitas dilakukan dengan tujuan untuk mengetahui tingkat kemudahan penggunaan sistem oleh responden. Pengujian dilakukan dengan menggunakan kuesioner yang terdiri dari beberapa pertanyaan beserta jawaban pada masing-masing pertanyaan. Kuesioner ini didasarkan pada *System Usability Scale* (SUS) yang telah dijelaskan pada landasan kepustakaan.

Skenario pengujian usabilitas yang dilakukan yaitu: memberikan penjelasan secara umum mengenai penelitian yang dilakukan, meminta responden untuk mencoba aplikasi yang telah dibuat, dan meminta responden untuk mengisi kuesioner. Setelah responden mengisi kuesioner yang diberikan, hasil dari kuesioner dikumpulkan dan dihitung. Contoh kuesioner yang telah diisi oleh responden dapat dilihat pada LAMPIRAN A.

Tabel 6.1 Hasil pengujian usabilitas

	Kuesioner	STS	TS	N	S	SS	Total
1	I think that I would like to use this system frequently. (Saya berpikir bahwa saya ingin menggunakan sistem ini dengan sering)	0	0	4	7	1	33
2	I found the system unnecessarily complex. (Saya menemukan sistem ini kompleks yang tidak perlu)	2	1	4	5	0	24
3	I thought the system was easy to use. (Saya pikir sistem ini mudah digunakan)	0	0	1	7	4	39
4	I think that I would need the support of a technical person to be able to use this system. (Saya berpikir bahwa saya akan membutuhkan dukungan dari orang teknis untuk dapat menggunakan sistem ini)	3	9	0	0	0	39

5	I found the various functions in this system were well integrated. (Saya menemukan berbagai fungsi dalam sistem ini yang terintegrasi dengan baik)	1	1	2	8	0	29
6	I thought there was too much inconsistency in this system. (Saya pikir ada terlalu banyak inkonsistensi dalam sistem ini)	1	9	2	0	0	35
7	I would imagine that most people would learn to use this system very quickly. (Saya akan membayangkan bahwa kebanyakan orang akan belajar untuk menggunakan sistem ini dengan sangat cepat.)	0	0	0	7	5	41
8	I found the system very cumbersome to use. (Saya menemukan sistem ini sangat rumit untuk digunakan)	5	7	0	0	0	41
9	I felt very confident using the system. (Saya merasa sangat yakin menggunakan sistem ini)	0	0	7	3	2	31
10	I needed to learn a lot of things before I could get going with this system. (Saya perlu belajar banyak hal sebelum terbiasa dengan sistem ini)	4	8	0	0	0	40
		Jumlah * 2.5					880
		Rata-rata skor					73.33

Tabel 6.1 merupakan hasil dari pengujian usabilitas dengan 12 responden. Berdasarkan Tabel 6.1, hasil pengujian usabilitas didapatkan rata-rata skor sebesar 73.33 yang berarti termasuk dalam klasifikasi di atas rata-rata.

6.2 Pengujian Reliabilitas

Pengujian reliabilitas dilakukan untuk mengukur tingkat konsistensi sistem terhadap banyaknya *user* serta *request* data. Pengujian reliabilitas dilakukan dengan menggunakan apache benchmark tools. Pengujian dilakukan dengan mengatur jumlah *request* yang berbeda-beda ke *server* serta banyaknya *user* yang melakukan *request* ke *server* secara bersamaan (*concurrent*).

Pengujian reliabilitas dilakukan dengan jumlah *request* yang berbeda-beda ke *server* dan jumlah *concurrent user* yang berbeda-beda. Skenario pengujian yang dilakukan yaitu: menjalankan apache bench tools, mengatur jumlah *request* dan *concurrent user*, mencatat hasil keluaran dari apache bench tools yang telah

dijalankan. Hasil keluaran dari apache bench tools yang diperlukan antara lain: *concurrency level*, *time for request*, *request per session*, *complete request*, dan *failed request*. Pengujian dilakukan hingga ditemukan *failed request* pada saat melakukan *request* ke server.

Tabel 6.2 Hasil pengujian reliabilitas web service

Concurrency Level	Time Per Request	Request Per Second	Complete Request	Failed Request	Percentage
1	35.74 ms	27.98	10	0	100%
5	56.25 ms	88.89	50	0	100%
10	121.6 ms	82.22	100	0	100%
25	297.39 ms	84.06	250	0	100%
50	590.62 ms	84.66	500	0	100%
100	1131.27 ms	88.4	1000	0	100%
200	2309.39 ms	86.6	2000	0	100%
300	3424.71 ms	87.6	3000	0	100%
500	6756.03 ms	74.01	5000	2	99.96%
1000	13566.15 ms	73.71	10000	2	99.98%

Tabel 6.2 merupakan hasil pengujian reliabilitas *web service* yang dilakukan sebanyak sepuluh kali dengan *concurrency user* 1, 5, 10, 25, 50, 100, 200, 300, 500, dan 1000. Hasil pengujian reliabilitas berdasarkan Tabel 6.2 menunjukkan bahwa sistem dapat menangani hingga 1000 user dengan *request* mencapai 10000 *request* dengan persentase keberhasilan mencapai 99.98%.

Tabel 6.3 merupakan hasil pengujian reliabilitas konten film yang dilakukan sebanyak lima kali dengan *concurrency user* 1, 5, 10, 25, dan 50. Hasil pengujian reliabilitas berdasarkan Tabel 6.3 menunjukkan bahwa sistem dapat menangani hingga 50 user dengan *request* konten film mencapai 50 *request* dengan persentase keberhasilan mencapai 60%.

Tabel 6.4 merupakan hasil pengujian reliabilitas konten musik yang dilakukan sebanyak enam kali dengan *concurrency user* 1, 5, 10, 25, 50, dan 100. Hasil pengujian reliabilitas berdasarkan Tabel 6.4 menunjukkan bahwa sistem dapat menangani hingga 100 user dengan *request* konten musik mencapai 100 *request* dengan persentase keberhasilan mencapai 96%.

Tabel 6.5 merupakan hasil pengujian reliabilitas konten buku yang dilakukan sebanyak sembilan kali dengan *concurrency user* 1, 5, 10, 25, 50, 100, 200, 300, dan 500. Hasil pengujian reliabilitas berdasarkan Tabel 6.5 menunjukkan bahwa sistem dapat menangani hingga 500 user dengan *request* konten buku mencapai 500 *request* dengan persentase keberhasilan mencapai 99.4%.

Tabel 6.6 merupakan hasil pengujian reliabilitas konten majalah yang dilakukan sebanyak enam kali dengan *concurrency user* 1, 5, 10, 25, 50, dan 100. Hasil pengujian reliabilitas berdasarkan Tabel 6.6 menunjukkan bahwa sistem dapat menangani hingga 100 user dengan *request* konten majalah mencapai 100 *request* dengan persentase keberhasilan mencapai 82%.

Tabel 6.3 Hasil pengujian reliabilitas konten film

Concurrency Level	Time Per Request	Request Per Second	Complete Request	Failed Request	Percentage
1	33854 ms	0.03	1	0	100%
5	2956 ms	0.34	5	0	100%
10	3891 ms	0.26	10	0	100%
25	7200 ms	0.14	25	0	100%
50	6001 ms	0.17	50	20	60%

Tabel 6.4 Hasil pengujian reliabilitas konten musik

Concurrency Level	Time Per Request	Request Per Second	Complete Request	Failed Request	Percentage
1	33.03 ms	30.27	1	0	100%
5	33.62 ms	29.74	5	0	100%
10	32.12 ms	31.13	10	0	100%
25	29.94 ms	33.4	25	0	100%
50	37.93 ms	26.37	50	0	100%
100	3001.34 ms	0.33	100	4	96%

Tabel 6.5 Hasil pengujian reliabilitas konten buku

Concurrency Level	Time Per Request	Request Per Second	Complete Request	Failed Request	Percentage
1	31.24 ms	32.01	1	0	100%
5	28.12 ms	35.56	5	0	100%
10	28.12 ms	35.56	10	0	100%
25	27.69 ms	36.12	25	0	100%
50	23.54 ms	42.48	50	0	100%
100	21.4 ms	46.73	100	0	100%
200	20.72 ms	48.26	200	0	100%
300	20.59 ms	48.56	300	0	100%
500	33.85 ms	29.54	500	3	99.4%

Tabel 6.6 Hasil pengujian reliabilitas konten majalah

Concurrency Level	Time Per Request	Request Per Second	Complete Request	Failed Request	Percentage
1	156 ms	6.4	1	0	100%
5	146 ms	6.81	5	0	100%
10	156.25 ms	6.4	10	0	100%
25	162.5 ms	6.15	25	0	100%
50	157.05 ms	6.37	50	0	100%
100	3000.221ms	0.3	100	18	82%

6.3 Pengujian Fungsionalitas

Pengujian fungsionalitas dilakukan setelah sistem selesai dibangun. Pengujian dilakukan sesuai dengan kasus uji pada masing-masing kebutuhan fungsional. Hasil dari pengujian fungsionalitas berupa tabel yang mengindikasikan apakah kebutuhan fungsional telah terpenuhi oleh sistem (valid) atau tidak (tidak valid).

Tabel 6.7 Kasus uji fungsional Melihat list

Nomor Kasus Uji	UF_01
Nama Kasus Uji	Melihat list
Nomor <i>Use Case</i>	F01
Tujuan Pengujian	Pengujian yang dilakukan untuk memastikan bahwa sistem dapat menampilkan daftar konten
Prosedur Uji	1. Memilih jenis hiburan musik dengan cara menekan menu berlabel musik pada beranda.
Hasil yang Diharapkan	Sistem menampilkan daftar konten musik berupa gambar dan judul sesuai dengan data musik yang diterima dari <i>server</i> . Konten musik yang termuat yaitu: Roll with the punches, See You Again, Sugar – Marron 5, I really like you, I'm yours, dan Locked Away

Tabel 6.7 menjelaskan kasus uji dari pengujian fungsionalitas pada kebutuhan fungsional melihat list. Pengujian dilakukan dengan cara membuka halaman daftar konten yang berisi konten-konten dalam bentuk *gridview* sesuai dalam Gambar 5.4.

Tabel 6.8 Kasus uji fungsional membuka konten berjenis musik

Nomor Kasus Uji	UF_02
Nama Kasus Uji	Membuka konten musik
Nomor <i>Use Case</i>	F02
Tujuan Pengujian	Pengujian yang dilakukan untuk memastikan bahwa sistem dapat membuka konten musik sesuai dengan pilihan pengguna
Prosedur Uji	1. Memilih menu berlabel musik pada beranda 2. Memilih konten yang berjudul 'I really like you' dengan cara menekan gambar atau judul dari konten tersebut pada daftar konten musik 3. Menekan tombol 'Play Music' untuk membuka konten
Hasil yang Diharapkan	Sistem membuka konten musik yang berjudul 'I really like you'

Tabel 6.9 Kasus uji fungsional membuka konten berjenis film

Nomor Kasus Uji	UF_03
Nama Kasus Uji	Membuka konten film
Nomor <i>Use Case</i>	F02
Tujuan Pengujian	Pengujian yang dilakukan untuk memastikan bahwa sistem dapat membuka konten film sesuai dengan pilihan pengguna
Prosedur Uji	1. Memilih menu berlabel movie pada beranda 2. Memilih konten yang berjudul 'Hitman' dengan cara menekan gambar atau judul dari konten tersebut pada daftar konten film 3. Menekan tombol 'Play Movie' untuk membuka konten
Hasil yang Diharapkan	Sistem membuka konten film yang berjudul 'Hitman'

Tabel 6.10 Kasus uji fungsional membuka konten berjenis buku

Nomor Kasus Uji	UF_04
Nama Kasus Uji	Membuka konten buku
Nomor <i>Use Case</i>	F02
Tujuan Pengujian	Pengujian yang dilakukan untuk memastikan bahwa sistem dapat membuka konten buku sesuai dengan pilihan pengguna
Prosedur Uji	1. Memilih menu berlabel Book pada beranda 2. Memilih konten yang berjudul 'Broken' dengan cara menekan gambar atau judul dari konten tersebut pada daftar konten buku 3. Menekan tombol 'Read Book' untuk membuka konten
Hasil yang Diharapkan	Sistem membuka konten buku yang berjudul 'Broken'

Tabel 6.11 Kasus uji fungsional membuka konten berjenis majalah

Nomor Kasus Uji	UF_05
Nama Kasus Uji	Membuka konten majalah
Nomor <i>Use Case</i>	F02
Tujuan Pengujian	Pengujian yang dilakukan untuk memastikan bahwa sistem dapat membuka konten majalah sesuai dengan pilihan pengguna
Prosedur Uji	1. Memilih menu berlabel Magazine pada beranda 2. Memilih konten yang berjudul 'Cuisine' dengan cara menekan gambar atau judul dari konten tersebut pada daftar konten majalah

	3. Menekan tombol 'Read Magazine' untuk membuka konten majalah tersebut
Hasil yang Diharapkan	Sistem membuka konten majalah yang berjudul 'Cuisine'

Tabel 6.8 menjelaskan kasus uji dari pengujian fungsionalitas pada kebutuhan fungsional membuka konten yang berjenis musik. Tabel 6.9 menjelaskan kasus uji dari pengujian fungsionalitas pada kebutuhan fungsional membuka konten yang berjenis film. Tabel 6.10 menjelaskan kasus uji dari pengujian fungsionalitas pada kebutuhan fungsional membuka konten yang berjenis buku. Tabel 6.11 menjelaskan kasus uji dari pengujian fungsionalitas pada kebutuhan fungsional membuka konten yang berjenis majalah.

Tabel 6.12 Kasus uji fungsional mengelola data konten

Nomor Kasus Uji	UF_06
Nama Kasus Uji	Mengelola data konten
Nomor <i>Use Case</i>	F03
Tujuan Pengujian	Pengujian yang dilakukan untuk memastikan bahwa sistem dapat melakukan proses kelola data konten yang dilakukan oleh admin
Prosedur Uji	1. Membuka halaman web yang digunakan sebagai <i>server</i> dari aplikasi 2. Memilih jenis hiburan musik dengan cara menekan tombol menu musik. 3. Melakukan pengubahan data konten yang berjudul 'Roll With the punches' dengan cara menekan tombol edit 4. Mengganti input description menjadi 'edited'.
Hasil yang Diharapkan	Data konten musik yang berjudul 'Roll With the punches' telah berubah pada kolom description menjadi 'edited'.

Tabel 6.12 menjelaskan kasus uji dari pengujian fungsionalitas pada kebutuhan fungsional mengelola data konten. Pengelolaan data konten dilakukan pada *server*. Kebutuhan fungsional mengelola data konten berhubungan dengan data konten yang diubah melalui *database*.

Tabel 6.13 Hasil pengujian fungsionalitas

No Kasus Uji	Nama Kasus Uji	Hasil yang diharapkan	Hasil yang diperoleh	Status
UF_01	Melihat list	Sistem menampilkan daftar konten musik berupa gambar dan judul	Sistem menampilkan daftar konten musik berupa gambar dan judul	Valid

		sesuai dengan data musik yang diterima dari <i>server</i> . Konten musik yang termuat yaitu: Roll with the punches, See You Again, Sugar – Marron 5, I really like you, I'm yours, dan Locked Away	sesuai dengan data musik yang diterima dari <i>server</i> . Konten musik yang termuat yaitu: Roll with the punches, See You Again, Sugar – Marron 5, I really like you, I'm yours, dan Locked Away	
UF_02	Membuka konten musik	Sistem membuka konten musik yang berjudul 'I really like you'	Sistem membuka konten musik yang berjudul 'I really like you'	Valid
UF_03	Membuka konten film	Sistem membuka konten film yang berjudul 'Hitman'	Sistem membuka konten film yang berjudul 'Hitman'	Valid
UF_04	Membuka konten buku	Sistem membuka konten buku yang berjudul 'Broken'	Sistem membuka konten buku yang berjudul 'Broken'	Valid
UF_05	Membuka konten majalah	Sistem membuka konten majalah yang berjudul 'Cuisine'	Sistem membuka konten majalah yang berjudul 'Cuisine'	Valid
UF_06	Mengelola konten	Data konten musik yang berjudul 'Roll With the punches' telah berubah pada kolom description menjadi 'edited'	Data konten musik yang berjudul 'Roll With the punches' telah berubah pada kolom description menjadi 'edited'	Valid

Tabel 6.13 merupakan hasil dari pengujian fungsionalitas tiap-tiap kebutuhan fungsional terhadap sistem. Pada Tabel 6.13 terdapat enam buah kasus uji beserta hasil pengujian yang telah dilakukan. Hasil yang didapat setelah melakukan pengujian adalah keenam kasus uji menunjukkan status valid yang berarti hasil yang diperoleh telah sesuai dengan hasil yang diharapkan.

BAB 7 PENUTUP

7.1 Kesimpulan

Berdasarkan hasil perancangan, implementasi, dan pengujian yang dilakukan, maka diambil kesimpulan sebagai berikut:

1. Aplikasi *entertainment access terminal* yang dibangun berdasarkan perancangan dan kebutuhan pengguna dengan hasil *activity diagram*, *sequence diagram*, *class diagram*, diagram logis *database*, *page flow* aplikasi, dan *user experience* aplikasi, berhasil diimplementasikan pada tablet Android.
2. Berdasarkan hasil pengujian fungsionalitas, aplikasi yang dibangun telah memenuhi kriteria yang ada pada spesifikasi kebutuhan fungsional. Berdasarkan hasil pengujian usabilitas, hasil perhitungan rata-rata skor yang didapatkan sebesar 73.33 yang berarti tingkat usabilitas sistem termasuk dalam klasifikasi di atas rata-rata. Berdasarkan hasil pengujian reliabilitas, sistem dapat menangani *request web service* mencapai 1000 user dengan persentase keberhasilan 99.98%, *request* konten film mencapai 50 user dengan persentase keberhasilan 60%, *request* konten musik mencapai 100 user dengan persentase keberhasilan 96%, *request* konten buku mencapai 500 user dengan persentase keberhasilan 99.4%, dan *request* konten majalah mencapai 100 user dengan persentase keberhasilan 82%.

7.2 Saran

Saran yang dapat diberikan setelah menyelesaikan penelitian skripsi ini yaitu:

1. Pengembangan aplikasi dapat dilakukan dengan penambahan fitur yang memungkinkan aplikasi melakukan beberapa kegiatan dalam satu waktu (*multitasking*) dengan cara menggunakan alir program (*thread*) yang berbeda pada tiap konten yang dieksekusi.
2. Pengembangan sistem dapat dilakukan dengan penambahan fitur *playlist* pada konten berjenis film dan musik dengan cara membuat kelas *playlist* yang berisi daftar judul dan URL konten yang dipilih oleh pengguna agar pengguna dapat menikmati konten film atau musik dengan sedikit interaksi pada aplikasi.

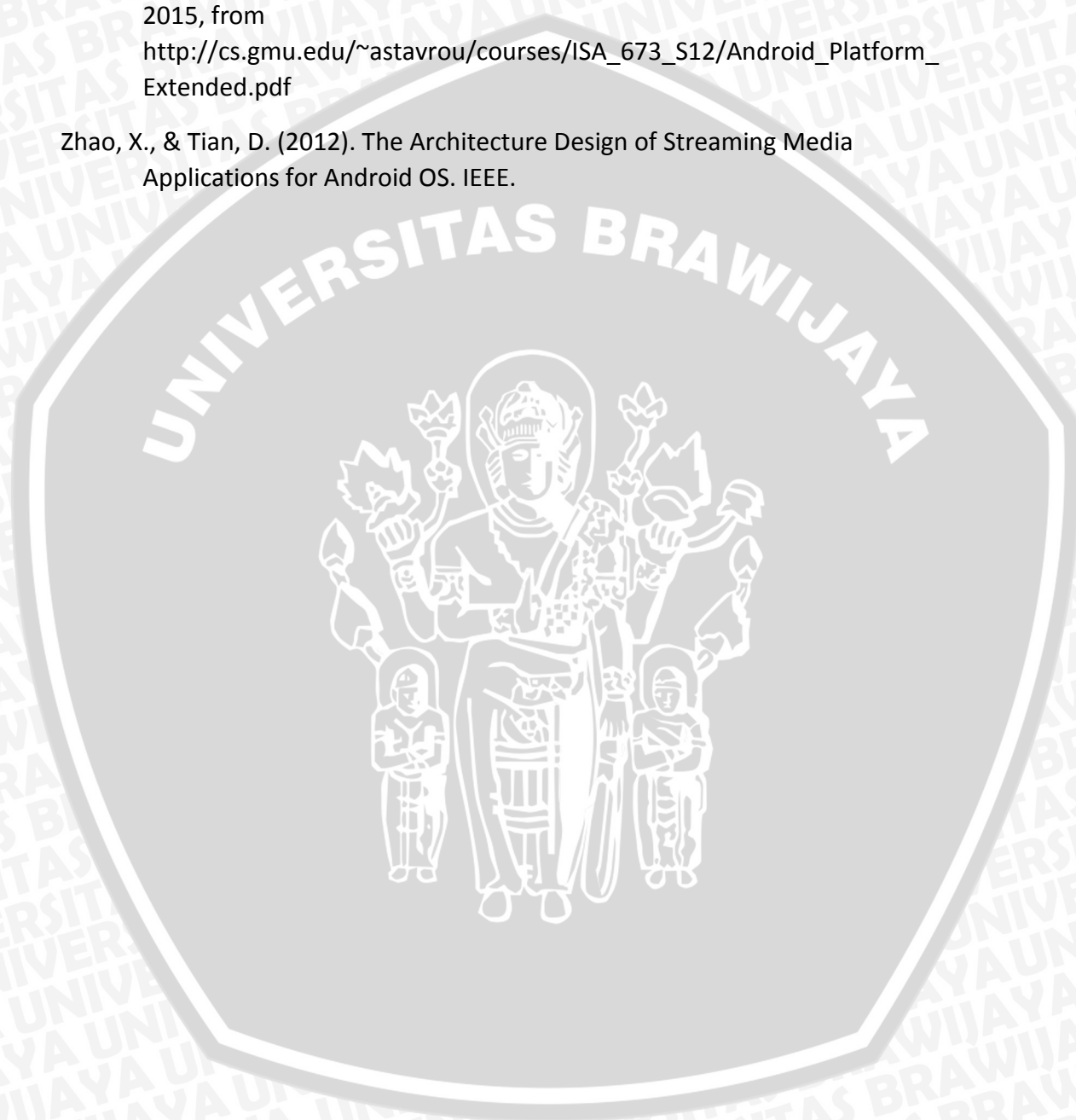
DAFTAR PUSTAKA

- Admob. (2011). *Admob Tablet Survey*. Google.
- Booth, D. (2004). *Web Services Architecture*. Retrieved Januari 24, 2015, from <http://www.w3.org/TR/2004/NOTE-ws-arch-20040211/wsa.pdf>
- BUMN. (2014). *PT KAI, Semakin Prima dalam Pelayanan*. Retrieved Desember 30, 2015, from <http://www.bumn.go.id/keretaapi/berita/1877/PT.KAI,.Semakin.Prima.dalam.Pelayanan>
- Burke, B. (2013). *RESTful Java with JAX-RS 2.0* (2nd ed.). O'Reilly Media.
- Developer, A. (n.d.). *Connecting to the Network*. Retrieved Agustus 2, 2015
- Developer, A. (n.d.). *Supported Media Formats*. Retrieved Juni 30, 2015, from <http://developer.Android.com/guide/appendix/media-formats.html>
- Felix, R. (2013). *Tablets Are Used Mainly For Entertainment*. Retrieved September 17, 2015, from <http://www.statista.com/chart/1482/activities-conducted-on-tablets>
- International, E. (2013). *The JSON Data Interchange Format*. Retrieved Januari 26, 2015, from <http://www.ecma-international.org/publications/files/ECMA-ST/ECMA-404.pdf>
- JSON. (n.d.). *Introducing JSON*. Retrieved Januari 27, 2015, from <http://www.json.org>
- Komputer, W. (2012). *Membuat Aplikasi Android untuk Tablet dan Handphone*. PT Elex Media Komputindo.
- Liu, H. (2007). In-flight Entertainment System: State of the Art and Research Directions. *Semantic Media Adaptation and Personalization*, 4.
- S, R., C, P., & S, P. (2013). *Android Operating Systems*. Takshshila Institute of Engineering and Technology.
- Sauro, J. (2011). *Measuring Usability with the System Usability Scale (SUS)*. Retrieved Nopember 10, 2015, from <http://www.measuringu.com/sus.php>
- Schuberth, P. A. (2011). In-Flight Entertainment & Communications Systems 101. THALES.
- Utama, Y. (n.d.). *Teknik Pemrograman Web Service PHP Dengan Menggunakan SOAP dan WSDL*. Yogyakarta: Andi Offset.

Vaqqas, M. (2014). *RESTful Web Services: A Tutorial*. Retrieved Februari 1, 2015, from <http://www.drdoobs.com/web-development/restful-web-services-a-tutorial/240169069>

Wang, Z., & Stavrou, A. (n.d.). *Google Android Platform*. Retrieved Januari 24, 2015, from http://cs.gmu.edu/~astavrou/courses/ISA_673_S12/Android_Platform_Extended.pdf

Zhao, X., & Tian, D. (2012). The Architecture Design of Streaming Media Applications for Android OS. IEEE.



LAMPIRAN A CONTOH KUESIONER PENGUJIAN USABILITAS

KUISIONER

Evaluasi *usability* aplikasi *entertainment access terminal* pada tablet Android. Berilah tanda silang (X) pada kolom yang anda anggap sesuai.

- 1 Strongly Disagree (sangat tidak setuju)
- 2 Disagree (tidak setuju)
- 3 Neutral (netral)
- 4 Agree (setuju)
- 5 Strongly Agree (sangat setuju)

	Kuisiонер	1	2	3	4	5
1	I think that I would like to use this system frequently. (Saya berpikir bahwa saya ingin menggunakan sistem ini dengan sering)			X		
2	I found the system unnecessarily complex. (Saya menemukan sistem ini tidak seharusnya rumit)				X	
3	I thought the system was easy to use. (Saya pikir sistem ini mudah digunakan)					X
4	I think that I would need the support of a technical person to be able to use this system. (Saya berpikir bahwa saya akan membutuhkan dukungan dari orang teknis untuk dapat menggunakan sistem ini)		X			
5	I found the various functions in this system were well integrated. (Saya menemukan berbagai fungsi dalam sistem ini yang terintegrasi dengan baik)				X	
6	I thought there was too much inconsistency in this system. (Saya pikir ada terlalu banyak inkonsistensi dalam sistem ini)			X		

This questionnaire is based on the System Usability Scale (SUS), which was developed by John Brooke while working at Digital Equipment Corporation. © Digital Equipment Corporation, 1986.

7	I would imagine that most people would learn to use this system very quickly. (Saya akan membayangkan bahwa kebanyakan orang akan belajar untuk menggunakan sistem ini dengan sangat cepat.)					X	
8	I found the system very cumbersome to use. (Saya menemukan sistem ini sangat rumit untuk digunakan)		X				
9	I felt very confident using the system. (Saya merasa sangat yakin menggunakan sistem ini)					X	
10	I needed to learn a lot of things before I could get going with this system. (Saya perlu belajar banyak hal sebelum terbiasa dengan sistem ini)		X				

This questionnaire is based on the System Usability Scale (SUS), which was developed by John Brooke while working at Digital Equipment Corporation. © Digital Equipment Corporation, 1986.