

**UNIVERSITAS INDONESIA**

**ANALISIS PERFORMA VIDEO STREAMING SEMANTIK INTERAKTIF INVIDANCE**

**SKRIPSI**

**NADYA SYIFANA**

**0906638401**

**FAKULTAS TEKNIK UNIVERSITAS INDONESIA**

**DEPARTEMEN TEKNIK ELEKTRO**

**PROGRAM STUDI TEKNIK KOMPUTER**

**DEPOK**

**JULI 2013**



**UNIVERSITAS INDONESIA**

**ANALISIS PERFORMA VIDEO STREAMING SEMANTIK INTERAKTIF INVIDANCE**

**SKRIPSI**

Diajukan untuk melengkapi sebagian prasyarat menjadi Sarjana Teknik

**NADYA SYIFANA**

**0906638401**

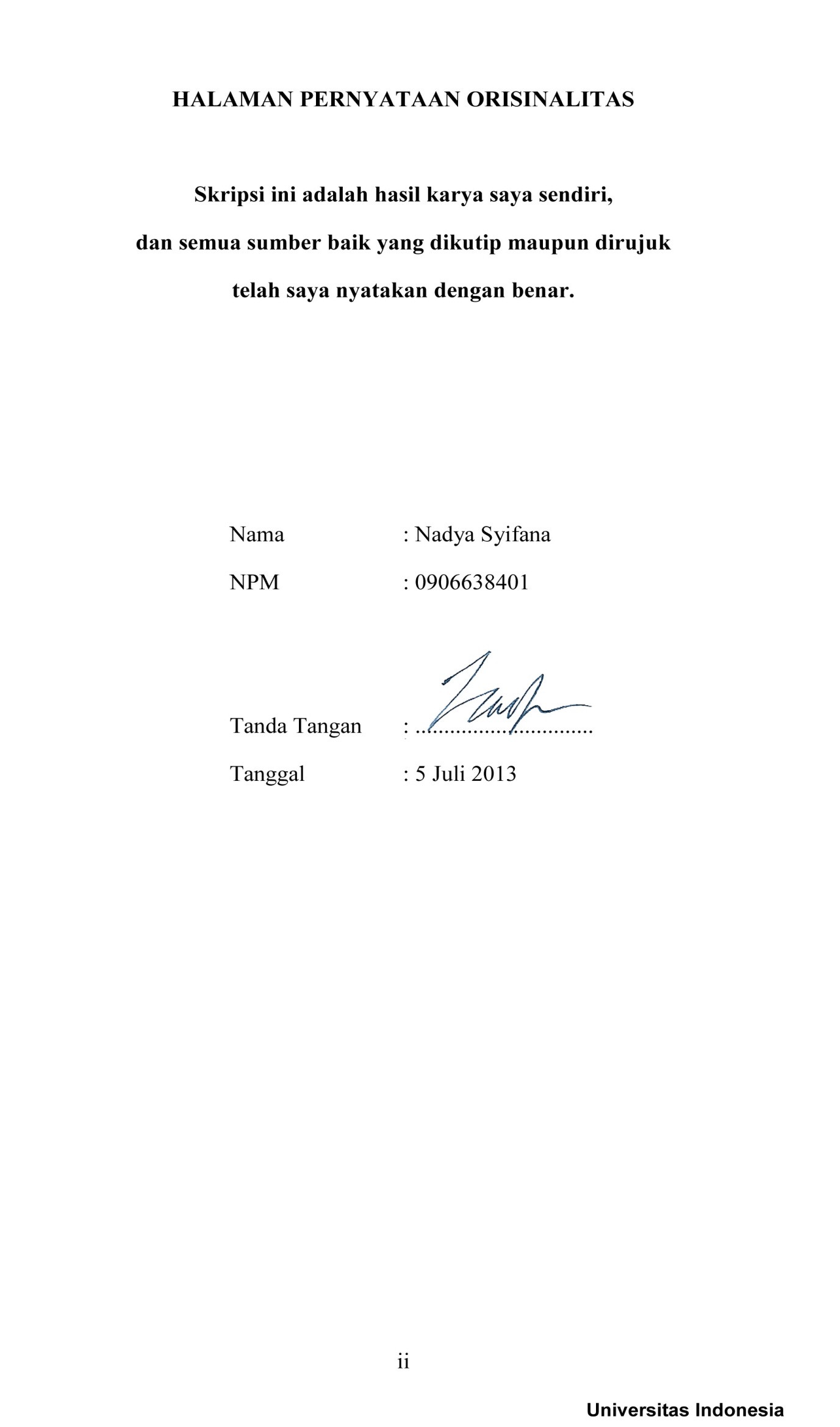
**FAKULTAS TEKNIK UNIVERSITAS INDONESIA**

**DEPARTEMEN TEKNIK ELEKTRO**

**PROGRAM STUDI TEKNIK KOMPUTER**

**DEPOK**

**JULI 2013**

HALAMAN PERNYATAAN ORISINALITAS

**Skripsi ini adalah hasil karya saya sendiri,**

**dan semua sumber baik yang dikutip maupun dirujuk**

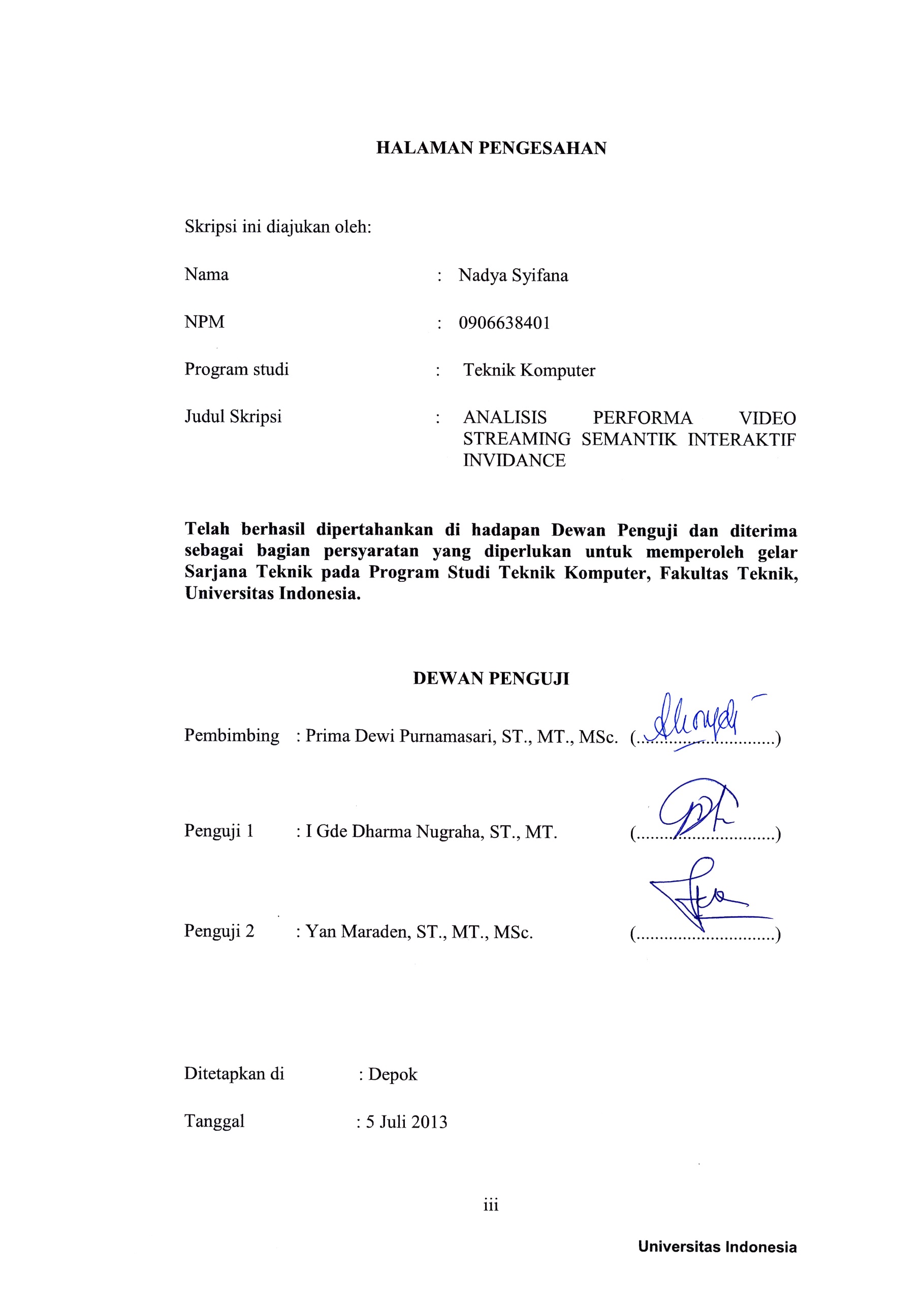
**telah saya nyatakan dengan benar.**

Nama : Nadya Syifana

NPM : 0906638401

Tanda Tangan : ...............................

Tanggal : 5 Juli 2013

HALAMAN PENGESAHAN

Skripsi ini diajukan oleh:

Nama : Nadya Syifana

NPM : 0906638401

Program studi : Teknik Komputer

Judul Skripsi : ANALISIS PERFORMA VIDEO STREAMING SEMANTIK INTERAKTIF INVIDANCE

**Telah berhasil dipertahankan di hadapan Dewan Penguji dan diterima sebagai bagian persyaratan yang diperlukan untuk memperoleh gelar Sarjana Teknik pada Program Studi Teknik Komputer, Fakultas Teknik, Universitas Indonesia.**

**DEWAN PENGUJI**

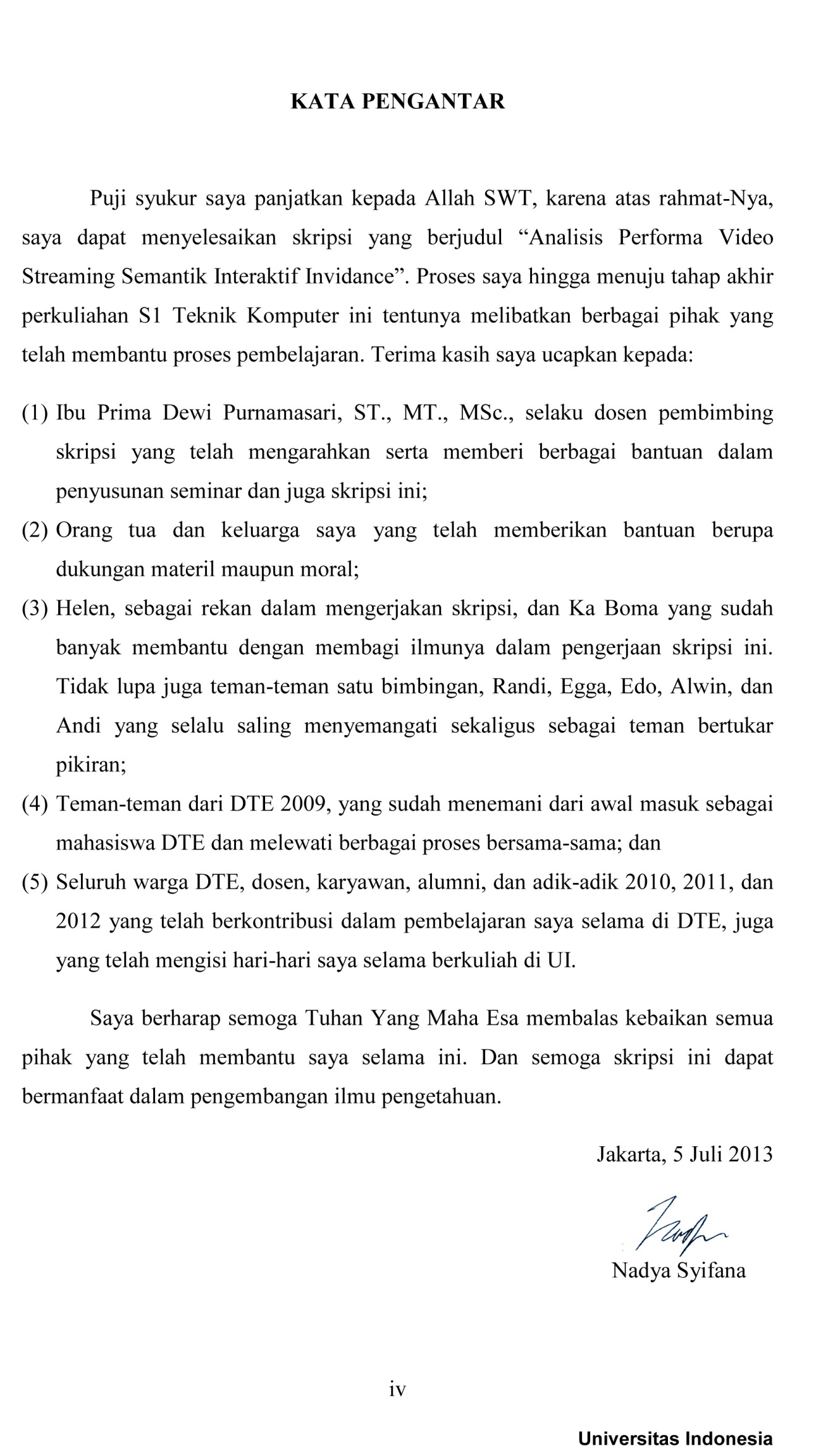
Pembimbing : Prima Dewi Purnamasari, ST., MT., MSc. (..............................)

Penguji 1 : I Gde Dharma Nugraha, ST., MT. (..............................)

Penguji 2 : Yan Maraden, ST., MT., MSc. (..............................)

Ditetapkan di : Depok

Tanggal : 5 Juli 2013

KATA PENGANTAR

Puji syukur saya panjatkan kepada Allah SWT, karena atas rahmat-Nya, saya dapat menyelesaikan skripsi yang berjudul “Analisis Performa Video Streaming Semantik Interaktif Invidance”. Proses saya hingga menuju tahap akhir perkuliahan S1 Teknik Komputer ini tentunya melibatkan berbagai pihak yang telah membantu proses pembelajaran. Terima kasih saya ucapkan kepada:

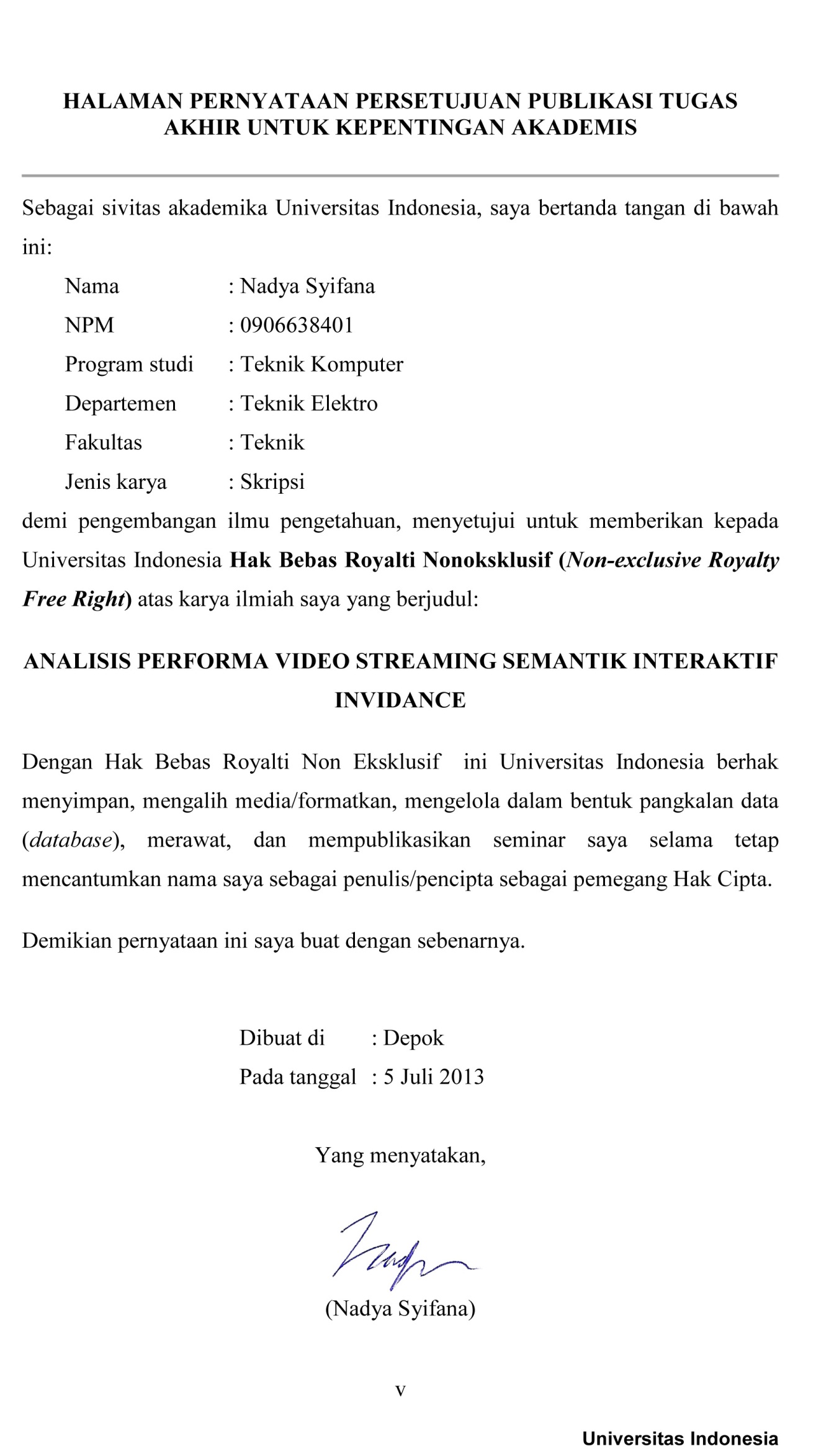
1. Ibu Prima Dewi Purnamasari, ST., MT., MSc., selaku dosen pembimbing skripsi yang telah mengarahkan serta memberi berbagai bantuan dalam penyusunan seminar dan juga skripsi ini;
2. Orang tua dan keluarga saya yang telah memberikan bantuan berupa dukungan materil maupun moral;
3. Helen, sebagai rekan dalam mengerjakan skripsi, dan Ka Boma yang sudah banyak membantu dengan membagi ilmunya dalam pengerjaan skripsi ini. Tidak lupa juga teman-teman satu bimbingan, Randi, Egga, Edo, Alwin, dan Andi yang selalu saling menyemangati sekaligus sebagai teman bertukar pikiran;
4. Teman-teman dari DTE 2009, yang sudah menemani dari awal masuk sebagai mahasiswa DTE dan melewati berbagai proses bersama-sama; dan
5. Seluruh warga DTE, dosen, karyawan, alumni, dan adik-adik 2010, 2011, dan 2012 yang telah berkontribusi dalam pembelajaran saya selama di DTE, juga yang telah mengisi hari-hari saya selama berkuliah di UI.

Saya berharap semoga Tuhan Yang Maha Esa membalas kebaikan semua pihak yang telah membantu saya selama ini. Dan semoga skripsi ini dapat bermanfaat dalam pengembangan ilmu pengetahuan.

Jakarta, 5 Juli 2013

Nadya Syifana

HALAMAN PERNYATAAN PERSETUJUAN PUBLIKASI TUGAS AKHIR UNTUK KEPENTINGAN AKADEMIS



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

Nama : Nadya Syifana

NPM : 0906638401

Program studi : Teknik Komputer

Departemen : Teknik Elektro

Fakultas : Teknik

Jenis karya : Skripsi

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

**ANALISIS PERFORMA VIDEO STREAMING SEMANTIK INTERAKTIF INVIDANCE**

Dengan Hak Bebas Royalti Non Eksklusif ini Universitas Indonesia berhak menyimpan, mengalih media/formatkan, mengelola dalam bentuk pangkalan data (*database*), merawat, dan mempublikasikan seminar saya selama tetap mencantumkan nama saya sebagai penulis/pencipta sebagai pemegang Hak Cipta.

Demikian pernyataan ini saya buat dengan sebenarnya.

Dibuat di : Depok

Pada tanggal : 5 Juli 2013

Yang menyatakan,

(Nadya Syifana)

ABSTRAK

|  |  |  |
| --- | --- | --- |
| Nama | : | Nadya Syifana |
| Program Studi | : | Teknik Komputer |
| Judul | : | Analisis Performa Video Streaming Semantik Interaktif Invidance |

Invidance merupakan sebuah aplikasi *video streaming* semantik interaktif berbasis Web bertemakan tari tradisional Indonesia. Sistem *database* Invidance terdiri dari dua bagian, yaitu sistem *admin* dan sistem *viewer*. Sistem *admin* berfungsi untuk mengelola *database* dalam pembuatan video interaktif dengan *clickable icon*, sekaligus menanamkan aspek semantik pada video. Sistem *viewer* berfungsi pada pengelolaan *database* untuk menampilkan *video streaming* semantik interaktif. Performa pengunggahan properti video dipengaruhi oleh *bitrate* video, dimana semakin kecil *bitrate* maka semakin baik. Panjang karakter teks informasi merupakan faktor paling berpengaruh dalam performa pengunggahan properti *clickable icon* dan informasi juga performa pencarian kata kunci. Uji performa pada subsistem sistem *viewer* memberikan rata-rata response time untuk penarikan properti video sebesar 2.72 ms, penarikan properti *clickable icon* sebesar 2.68 ms, dan penarikan properti informasi sebesar 3.42 ms.

Kata kunci: Sistem *Database*, Sistem interaktif, Web semantik

ABSTRACT

Name : Nadya Syifana

Study Program : Computer Engineering

Title : Performance Analysis of Invidance Interactive Semantic Video Streaming

Invidance is an Indonesian traditional dance themed interactive semantic video streaming application based on Web. Invidance database system consists of two parts, the admin system and viewer system. Admin system serves to manage the database on creating the interactive video with clickable icon, as well as embed semantic aspects of the video. Viewer system works on database management for displaying interactive semantic video streaming. Upload performance for video properties is affected by video’s bitrate, where the smaller bitrate gives a good result. Character length of information text is the most affecting factor for the performance of clickable icon and information properties upload, and for the performance of keyword search. The performance test on subsystem of viewer system gave the average response time of 2.72 ms for video properties retrieval, 2.68 ms for clickable icon properties retrieval, and 3.42 ms for information properties retrieval.

Keyword: Databases Systems, Interactive systems, Semantic Web

**DAFTAR ISI**

[HALAMAN PERNYATAAN ORISINALITAS ii](#_Toc360221295)

[HALAMAN PENGESAHAN iii](#_Toc360221296)

[KATA PENGANTAR iv](#_Toc360221297)

[HALAMAN PERNYATAAN PERSETUJUAN PUBLIKASI TUGAS AKHIR UNTUK KEPENTINGAN AKADEMIS v](#_Toc360221298)

[ABSTRAK vi](#_Toc360221299)

[ABSTRACT vii](#_Toc360221300)

[DAFTAR ISI viii](#_Toc360221301)

[DAFTAR TABEL x](#_Toc360221302)

[DAFTAR GAMBAR xi](#_Toc360221303)

[BAB I PENDAHULUAN 1](#_Toc360221304)

[1.1 Latar Belakang 1](#_Toc360221305)

[1.2 Tujuan Penelitian 2](#_Toc360221306)

[1.3 Batasan Masalah 2](#_Toc360221307)

[1.4 Metode Penulisan 2](#_Toc360221308)

[1.5 Sistematika Penulisan 2](#_Toc360221309)

[BAB II MULTIMEDIA, DATABASE, WEB SEMANTIK, DAN VIDEO INTERAKTIF 4](#_Toc360221310)

[2.1 Multimedia 4](#_Toc360221311)

[2.1.1 Karakteristik Data Multimedia 5](#_Toc360221312)

[2.2 Database 6](#_Toc360221313)

[2.2.1 Model Data 7](#_Toc360221314)

[2.2.2 Skema, Instance, dan Database State 11](#_Toc360221315)

[2.3 Database Management System 12](#_Toc360221316)

[2.3.1 DBMS Language 14](#_Toc360221317)

[2.3.2 Struktur Komponen DBMS 14](#_Toc360221318)

[2.3.3 Database Query Language 16](#_Toc360221319)

[2.4 Multimedia Database 18](#_Toc360221320)

[2.4.1 Syarat dan Arsitektur MMDBMS 19](#_Toc360221321)

[2.4.2 Query dalam Database Multimedia 19](#_Toc360221322)

[2.5 PHP dan Web 20](#_Toc360221323)

[2.6 Semantic Web 21](#_Toc360221324)

[2.7 Video Interaktif 23](#_Toc360221325)

[BAB III PERANCANGAN SISTEM UNTUK VIDEO STREAMING SEMANTIK INTERAKTIF 24](#_Toc360221326)

[3.1 Gambaran Umum Sistem Video Interaktif Berbasis Web 24](#_Toc360221327)

[3.1.1 Rancangan Sistem 25](#_Toc360221328)

[3.1.2 Alur Sistem 27](#_Toc360221329)

[3.2 Perancangan Sistem Database 30](#_Toc360221330)

[3.2.1 Normalisasi Database 30](#_Toc360221331)

[3.2.2 Struktur Database 32](#_Toc360221332)

[3.3 Algoritma Sistem 34](#_Toc360221333)

[3.3.1 Algoritma Sistem Admin 35](#_Toc360221334)

[3.3.2 Algoritma Sistem Viewer 37](#_Toc360221335)

[BAB IV IMPLEMENTASI, PENGUJIAN, DAN ANALISIS PERFORMA VIDEO STREAMING SEMANTIK INTERAKTIF 40](#_Toc360221336)

[4.1 Implementasi Sistem Database pada Invidance 40](#_Toc360221337)

[4.1.1 Halaman Admin 42](#_Toc360221338)

[4.1.2 Halaman Preview 45](#_Toc360221339)

[4.2 Uji Fungsionalitas Sistem Invidance 45](#_Toc360221340)

[4.2.1 Pengunggahan Video 46](#_Toc360221341)

[4.2.2 Pembuatan Clickable Icon dan Informasi 47](#_Toc360221342)

[4.2.3 Pencarian Kata Kunci dan Pencarian Informasi Terkait 49](#_Toc360221343)

[4.3 Uji Performa Sistem Invidance 52](#_Toc360221344)

[4.3.1 Pengunggahan Properti Video 53](#_Toc360221345)

[4.3.2 Pembuatan Clickable Icon dan Informasi 55](#_Toc360221346)

[4.3.3 Pencarian Kata Kunci 56](#_Toc360221347)

[4.3.4 Penarikan Properti Media dari Database pada Halaman Preview 57](#_Toc360221348)

[BAB V KESIMPULAN 59](#_Toc360221349)

[DAFTAR ACUAN 60](#_Toc360221350)

[DAFTAR PUSTAKA 62](#_Toc360221351)

[LAMPIRAN 64](#_Toc360221352)

DAFTAR TABEL

[Tabel 4.1 Spesifikasi Properti Video Pengujian 53](#_Toc360221386)

[Tabel 4.2 Spesifikasi Informasi Pengujian 55](#_Toc360221387)

[Tabel 4.3 Response Time Penarikan Properti Media 57](#_Toc360221388)

DAFTAR GAMBAR

Gambar 2.1 Relational Data Model [4] 8

Gambar 2.2 Object Data Model [4] 9

Gambar 2.3 Contoh XML [4] 10

Gambar 2.4 Network Data Model [4] 10

Gambar 2.5 Hierarchical Data Model [4] 11

Gambar 2.6 Struktur Komponen DBMS [4] 15

Gambar 2.7 Arsitektur Multimedia DBMS [2] 20

Gambar 3.1 Logo Invidance 24

Gambar 3.2 Gambaran Umum Sistem Invidance 25

Gambar 3.3 Use Case Diagram 26

Gambar 3.4 Activity Diagram Sistem *Admin* 28

Gambar 3.5 Activity Diagram Sistem *Viewer* 29

Gambar 3.6 *Raw Database* Invidance 30

Gambar 3.7 *First Normal Form* 31

Gambar 3.8 *Second Normal Form* 31

Gambar 3.9 *Third Normal Form* Tabel INFO 32

Gambar 3.10 ERD Database Invidance 33

Gambar 3.11 *Pseudocode* Sistem *Admin* Invidance 34

Gambar 3.12 *Pseudocode* Sistem *Viewer* Invidance 34

Gambar 3.13 *Pseudocode* Unggah Video 35

Gambar 3.14 *Pseudocode* Membuat *Clickable Icon* 35

Gambar 3.15 *Pseudocode* Membuat Informasi *Pop-up* 36

Gambar 3.16 *Pseudocode* Mencari Kata Kunci 36

Gambar 3.17 Ontologi Tari Tradisional pada Sistem Invidance 37

Gambar 3.18 *Pseudocode* Menampilkan Video 38

Gambar 3.19 *Pseudocode* Menarik Informasi Properti *Clickable Icon* 38

Gambar 3.20 *Pseudocode* Penarikan Properti Konten Informasi *Pop-up* 38

Gambar 3.21 *Pseudocode* untuk Mencari Informasi Terkait 39

Gambar 4.1 Tabel-tabel pada *Database* Invidance 41

Gambar 4.2 Tabel click\_area 41

Gambar 4.3 Tabel image 41

Gambar 4.4 Tabel info 42

Gambar 4.5 Tabel keyword 42

Gambar 4.6 Tabel text 42

Gambar 4.7 Tabel video 42

Gambar 4.8 Form Unggah Video 43

Gambar 4.9 Tampilan Form Input Informasi 44

Gambar 4.10 Halaman Preview Video Interaktif 44

Gambar 4.11 Mengunggah Video 46

Gambar 4.12 Notifikasi Video Berhasil Diunggah 46

Gambar 4.13 Hasil Unggah Video pada Tabel video 47

Gambar 4.14 Membuat *Clickable icon* dan Informasi 47

Gambar 4.15 Hasil Masukkan Informasi pada Tabel text 48

Gambar 4.16 Hasil Masukkan Informasi pada Tabel image 48

Gambar 4.17 Hasil Masukkan Informasi pada Tabel info 49

Gambar 4.18 Hasil Masukkan Informasi pada Tabel click\_area 49

Gambar 4.19 Tabel keyword 50

Gambar 4.20 Hasil Pencarian Kata Kunci pada Tabel info 50

Gambar 4.21 Penampilan Informasi terkait pada *Pop-up Window* 51

Gambar 4.22 Skrip PHP untuk Memulai Pengukuran *Response Time* [13] 52

Gambar 4.23 Skrip PHP untuk Mengakhiri Pengukuran *Response Time* [13] 53

Gambar 4.24 Grafik *Response Time* Pengunggahan Properti Video 54

Gambar 4.25 Grafik *Response Time* Pengunggahan Properti Video Tanpa Penyalinan *Temporary File* 54

Gambar 4.26 Grafik *Response Time* Input Informasi ke *Database* 55

Gambar 4.27 Grafik *Response Time* Pencarian Kata Kunci 56

# PENDAHULUAN

Pada bab ini akan dijabarkan mengenai latar belakang, tujuan, batasan masalah, metode penulisan, serta sistematika penulisan.

## Latar Belakang

Web bukan lagi hal yang asing bagi sebagian besar masyarakat. Semua orang yang pernah menggunakan komputer untuk mengakses Internet, tentunya akan berhubungan dengan Web. Kegunaan Web pun semakin beragam seiring meningkatnya minat masyarakat akan Internet. Teknologi-teknologi terkait Web yang terus berkembang juga telah membuat Web semakin mudah dan nyaman untuk dinikmati pengguna.

*Video streaming* merupakan aplikasi berbasis Web yang sangat terkenal saat ini. Dari sebuah halaman Web, seseorang dapat menyaksikan *video clip* musik terkini, film, hingga *live streaming* siaran berita. Dengan berkembangnya teknologi Web saat ini, sangat memungkinkan untuk memanipulasi sebuah video di halaman Web. Manipulasi ini tentunya bertujuan memaksimalkan video sehingga menjadi lebih interaktif dan informatif bagi pengguna.

Untuk membuat sebuah video interaktif berbasis Web, ada banyak hal yang perlu diperhatikan. Pertama, konten dari video tersebut. Animo masyarakat yang cukup tinggi akan *video streaming* melalui Web menjadikan video sarana yang strategis untuk menyebarkan informasi atau bahkan menciptakan sesuatu yang menjamur. Situasi ini seharusnya bisa dimanfaatkan untuk hal yang baik, seperti memberikan edukasi atau melestarikan kebudayaan.

Bahasa pemrograman serta *database* juga menjadi hal penting dalam membangun Web. Salah satu bahasa pemrograman yang sering digunakan yaitu *HyperText Markup Language* (HTML). Data-data mentah harus disimpan secara terstruktur di dalam sistem *database*, agar mudah diambil dan juga diolah. Untuk Web interaktif dengan konten multimedia dan informasi, manajemen *database* menjadi salah satu hal yang esensial. Manajemen *database* yang buruk dapat menurunkan performa Web dalam menyampaikan informasi, atau bahkan informasi tidak dapat dinikmati oleh pengguna.

## Tujuan Penelitian

Membuat dan menganalisis sistem *database* untuk memberikan aspek interaktif dan semantik pada *video streaming* dan juga menampilkan *video streaming* semantik interaktif dalam sebuah Web.

## Batasan Masalah

Sistem dirancang untuk aplikasi video semantik interaktif berbasis Web yang dibangun menggunakan bahasa pemrograman PHP. Untuk *database scripting* digunakan bahasa SQL.

## Metode Penulisan

Metode penulisan mengadopsi *software engineering cycle*, yaitu sebagai berikut:

1. *System Requirement*: menentukan spesifikasi sistem dengan melakukan studi literatur dan pengamatan pada aplikasi yang sudah ada
2. *System and Software Design*: merancang ontologi, struktur *database*, algoritma kerja sistem, serta perancangan *database* menggunakan bahasa SQL
3. *Implementation*: mengimplementasikan rancangan *database*, yaitu dengan menghubungkan *database* dengan sistem menggunakan bahasa pemrograman PHP
4. *Testing*: melakukan pengujian terhadap performa sistem
5. Analisis: analisis hasil pengujian sistem

## Sistematika Penulisan

*Bab 1 Pendahuluan*

Pada bab ini akan dijabarkan mengenai latar belakang, tujuan, batasan masalah, metode penulisan, serta sistematika penulisan.

*Bab 2 Multimedia, Database, Web Semantik, dan Video Interaktif*

Pada bab ini akan dijelaskan dasar teori yang berkaitan dengan multimedia, *database*, Web semantik dan juga video interaktif.

*Bab 3 Perancangan Sistem untuk Video Streaming Semantik Interaktif*

Pada bab ini akan dijelaskan mengenai perancangan sistem dan *database* dalam menunjang video streaming semantik interaktif.

*Bab 4 Implementasi, Pengujian, dan Analisis Performa Video Streaming Semantik Interaktif*

Pada bab ini akan dijelaskan mengenai implementasi sistem *database* untuk *video streaming* semantik interaktif, serta pengujian dan analisis fungsionalitas sistem.

*Bab 5 Kesimpulan*

Pada bab ini akan disampaikan kesimpulan dari implementasi, pengujian, serta analisis sistem *video streaming* semantik interaktif.

# MULTIMEDIA, DATABASE, WEB SEMANTIK, DAN VIDEO INTERAKTIF

Pada bab ini akan dibahas mengenai teori-teori yang berkaitan dengan topik penelitian. Adapun yang akan dibahas yaitu mengenai multimedia, *database*, Web semantik, dan video interaktif.

## Multimedia

Multimedia dapat diartikan sebagai presentasi dari campuran jenis informasi dalam bentuk sinyal digital. Informasi-informasi tersebut dapat berupa teks, data, gambar, audio, dan video. Beberapa aplikasi multimedia di antaranya adalah *video conferencing*, perpustakaan digital, dan *telemedicine*. Komunikasi multimedia berkaitan dengan teknologi untuk memanipulasi, transmisi dan mengendalikan sinyal audiovisual melewati jaringan kanal telekomunikasi.

Media dibagi dalam dua kelas, yaitu kontinu dan diskrit. Media kontinu, yang berubah sesuai waktu, contohnya seperti audio dan video. Sementara media diskrit, yang bersifat *time independent*, contohnya adalah teks (*formatted* dan *unformatted*), gambar, dan grafik.

Data multimedia adalah data yang tidak dapat direpresentasikan atau diindekskan dengan struktur *database* konvensional, dengan demikian sering pula disebut sebagai data tak berstruktur. Data tak berstruktur dapat didefinisikan sesuai jenis media spesifiknya. Data dapat direpresentasikan dari segi dimensi ruang data [1]:

* 0-dimensional data: Jenis data reguler yang berupa data alfanumerik. Contohnya adalah data teks.
* 1-dimensional data: Jenis data dengan satu dimensi ruang. Contohnya adalah data audio.
* 2-dimensional data: Jenis data dengan dua dimensi ruang. Contohnya adalah data imajiner dan data grafik.
* 3-dimensional data: Jenis data dengan tiga dimensi ruang. Contohnya adalah video dan animasi.

### Karakteristik Data Multimedia

Karakteristik dari data multimedia dapat dijelaskan sebagai berikut [2]:

* *Lack of structure*: Data multimedia cenderung tidak terstruktur. Pengelolaan data standar seperti *indexing* dan pencarian dan pengambilan berbasis konten tidak tersedia.
* *Temporality*: Beberapa jenis data multimedia seperti video, audio dan urutan animasi memiliki persyaratan sementara yang berimplikasi pada tempat penyimpanan, manipulasi, dan presentasi.
* *Massive Volume*: Data multimedia seperti video dan audio sering kali membutuhkan sebuah perangkat penyimpanan yang besar.
* *Logistics*: *Non-standard media* dapat mempersulit pemrosesan. Sebagai contoh, sebuah aplikasi *database* multimedia membutuhkan algoritma kompresi.

Dalam *data mining*, multimedia data direpresentasikan sebagai *feature*. Terdapat beberapa representasi *feature* dari media spesifik, yaitu [1]:

* TF-IDF: TF-IDF merupakan *feature* untuk data teks. Misalnya N adalah dokumen, dan M adalah total kosakata. Pemrosesan didasari asumsi *bag-of-words*, dimana dokumen hanya dianggap sebagai kumpulan kata terisolasi. TF, atau *Term Frequency Matrix*, merupakan matriks N×M yang menjadi representasi dari *database* teks. *Document frequency* (DF) merupakan banyaknya dokumen dimana sebuah kata muncul. Untuk memindahkan kata-kata yang muncul terlalu banyak, dibutuhkan *inverse document frequency* (IDF).
* *Cepstrum*: *Cepstrum* sering digunakan untuk data media satu dimensi, seperti data audio. Misalnya ada sebuah data media yang direpresentasikan sebagai sebuah sinyal satu dimensi, *cepstrum* didefinisikan sebagai transformasi Fourier dari spektrum desibel sinyal. Karena spektrum desibel dari sebuah sinyal merupakan logaritma dari transformasi Fourier sinyal asli, *cepstrum* terkadang disebut juga spektrum dari spektrum.
* *Fundamental Frequency*: *Fundamental frequency* mangacu pada frekuensi terendah dalam sebuah serangkaian harmonik yang dimiliki suara audio. Fundamental frequency biasa digunakan sebagai *feature* untuk data mining pada audio.
* *Audio Sound Attributes*: Atribut suara audio biasanya terdiri dari *pitch*, kenyaringan, dan timbre. *Pitch* adalah sensasi dari “ketinggian” yang berkaitan dengan frekuensi dari suara, yang juga berkaitan dengan fundamental *frequency* dari suara. Kenyaringan adalah sensasi “kekuatan” atau “intensitas” dari nada suara, terkait dengan intensitas energi. Timbre adalah sensasi dari “kualitas” suara audio, biasa terkait dengan spektrum suara audio.
* *Optical Flow*: *Optical flow* merupakan *feature* pada data media tiga dimensi seperti video dan animasi. *Optical flow* didefinisikan sebagai perubahan *brightness* pada lokasi spesifik dalam sebuah citra seiring berjalannya waktu.

## Database

*Database* adalah sekumpulan data terstruktur yang memiliki keterhubungan. Data sendiri merupakan fakta-fakta yang terekam dan memiliki arti secara implisit. *Database* memiliki properti-properti sebagai berikut [3]:

* *Database* merepresentasikan aspek-aspek yang ada di dunia nyata yang disebut *miniworld* atau *universe of discourse* (UoD). Perubahan yang terjadi pada *miniworld* akan terefleksi pada *database*.
* *Database* merupakan kumpulan data koheren secara logika yang memiliki arti inheren. Campuran data acak tidak dapat disebut sebagai *database*.
* Sebuah *database* didesain, dibangun, dan diisi dengan data untuk tujuan khusus.

Perancangan sebuah *database* dimulai dengan spesifikasi dan analisis kebutuhan. Spesifikasi rancangan ini didokumentasikan secara terperinci dan dibentuk ke dalam desain konseptual yang dapat dibuat dan dimanipulasi menggunakan program komputer agar mudah diatur, diubah, dan dibentuk menjadi *database*. Desain konseptual kemudian diubah menjadi desain logika yang diekspresikan dalam sebuah data model pada DBMS. Terakhir adalah desain fisik untuk spesifikasi penyimpanan dan pengaksesan *database*. Desain *database* diimplementasikan, diisi dengan data aktual, dan dijaga agar tetap merefleksikan keadaaan *miniworld*.

### Model Data

Salah satu karakteristik utama dari pendekatan *database* adalah tingkat abstraksi data. Abstraksi data secara umum adalah rincian organisasi dan penyimpanan data, serta penandaan hal khusus untuk membuat data lebih dimengerti [2]. Dengan abstraksi data, pengguna dapat mengerti data pada tingkat rincian yang diinginkan. Data model mendeskripsikan data, keterhubungan antar data, semantik data, dan konsistensi dari batasan data [4].

Terdapat banyak model data yang dapat dikategorikan berdasarkan konsep yang digunakan untuk mendeskripsikan struktur *database*. Model data konseptual, atau *high-level*, memiliki konsep yang mendekati cara kebanyakan pengguna dalam memahami data, sementara model data fisik, atau *low-level*, memiliki konsep yang menjelaskan secara rinci bagaimana data disimpan dalam media penyimpanan. Di antara kedua model ini terdapat model data representasional, atau implementasi, yang lebih mudah dimengerti oleh pengguna, namun juga masih dekat dengan cara data diorganisir dalam penyimpanan. Model data representasional menyembunyikan rincian mengenai penyimpanan data tetapi dapat diimplementasikan pada sistem komputer secara langsung [3].

Model data konseptual menggunakan konsep seperti entitas, atribut, dan *relationship*. Entitas menggambarkan objek atau konsep dalam dunia nyata. Atribut menjabarkan properti-properti yang menjelaskan entitas. *Relationship* menggambarkan keterhubungan antar dua atau lebih entitas.

Model data fisik menjelaskan bagaimana data disimpan sebagai berkas di dalam komputer dengan merepresentasikan informasi seperti format rekam, susunan rekam, dan *access path*. *Access path* merupakan struktur yang mengefisienkan pencarian rekaman *database*. Contoh *access path* yaitu indeks yang memungkinkan akses langusng ke data menggunakan sebuah aturan indeks atau kata kunci.

Model data representasional adalah model yang digunakan pada DBMS. Model data yang umum digunakan adalah model data *relational*. Model data lainnya yaitu *object*, *hierarchical*, dan *network*. DBMS *relational* telah berkembang pesat dan saat ini sudah dapat menyertakan beberapa konsep yang dikembangkan dalam *object database*. Hal ini memunculkan sebuah kelas baru dari DBMS, yaitu *object-relational* DBMS. Selain model data yang telah disebutkan, ada pula DBMS yang berbasis *eXtended Markup Language* (XML). XML memiliki model data *tree-structured*, atau hierarki. Berikut penjelasan pada masing-masing model:

1. Relational Data Model

Model data *relational* menggambarkan data dan hubungan antar data dalam kumpulan tabel, dimana tiap tabel disimpan dalam berkas yang berbeda. Tiap tabel memiliki sejumlah kolom dengan nama yang unik. *Relational database* menggunakan *query language* yang disebut SQL. Gambar 2.1 merupakan contoh model data *relational*.

|  |  |  |  |
| --- | --- | --- | --- |
| *name* | *street* | *city* | *number* |
| Lowery  Shiver  Shiver  Hodges  Hodges | Maple  North  North  Sidehill  Sidehill | Queens  Bronx  Bronx  Brooklyn  Brooklyn | 900  556  647  801  647 |

|  |  |
| --- | --- |
| *number* | *balance* |
| 900  556  647  801 | 55  100000  105366  10533 |

Gambar 2.1 Relational Data Model [4]

1. Object Data Model

Model data *object* mendefinisikan *database* sebagai objek-objek, properti objek, dan operasinya. Objek dengan struktur dan sifat yang sama digolongkan dalam sebuah kelas, dan kelas diatur dalam hierarki. Operasi pada masing-masing kelas ditentukan dalam sebuah prosedur yang disebut *method*. Contoh model data *object* ditunjukkan pada Gambar 2.2.

computer system

board

bus

device

instr-set

chips

interfaces

Gambar 2.2 Object Data Model [4]

1. XML Model

XML menggunakan struktur hierarki pohon, dimana data direpresentasikan sebagai elemen. XML menggabungkan konsep *database* dengan konsep representasi dokumen. Dengan penggunaan *tag*, dapat dibentuk data bersarang, atau *nested*, untuk membuat struktur hierarki yang rumit. Gambar 2.3 merupakan contoh kode XML.

1. Network Model

Data dalam *network model* direpresentasikan dengan kumpulan *record*. *Relationship* di antara data direpresentasikan dengan *link.* *Record* dalam *database* ditata sebagai kumpulan grafik yang berubah-ubah [4]. Gambar 2.4 memberikan contoh model data *network*.

<?xml version= “1.0” standalone=“yes”?>

<Projects>

<Project>

<Name>ProductX</Name>

<Number>1</Number>

<Location>Bellaire</Location>

<Dept\_no>5</Dept\_no>

<Worker>

<Ssn>123456789</Ssn>

<Last\_name>Smith</Last\_name>

<Hours>32.5</Hours>

</Worker>

<Worker>

<Ssn>453453453</Ssn>

<First\_name>Joyce</First\_name>

<Hours>20.0</Hours>

</Worker>

</Project

</Projects>

Gambar 2.3 Contoh XML [4]

Shiver

Lowery

Hodges

Maple

North

Sidehill

Queens

Bronx

Brooklyn

55

100 000

105 366

10 533

900

556

647

801

Gambar 2.4 Network Data Model [4]

1. Hierarchical Model

*Hierarchical model* merepresentasikan data dalam bentuk struktur pohon. Dalam setiap hierarki terdapat beberapa *record*. Gambar 2.5 merupakan contoh *hierarchical*.

Shiver

Lowery

Hodges

Maple

North

Sidehill

Queens

Bronx

Brooklyn

55

100 000

105 366

10 533

900

556

647

801

105 366

647

Gambar 2.5 Hierarchical Data Model [4]

### Skema, Instance, dan Database State

Skema *database* adalah deskripsi dari *database*, dispesifikasikan pada saat perancangan *database* dan tidak akan banyak berubah [3]. Perubahan skema, atau evolusi skema, bisa saja terjadi ketika kebutuhan aplikasi berubah. Skema dapat ditampilkan dalam bentuk diagram skema. Diagram menampilkan struktur setiap jenis record, tetapi bukan instance dari record. Tiap objek dalam skema disebut schema construct.

Data aktual dalam *database* dapat berubah cukup sering. Data di dalam *database* pada saat tertentu disebut *database* state. Setiap kali nilai data di dalam *database* ditambah, dihapus, atau diubah, state dari *database* berubah ke state lainnya. Dalam sebuah *database* state, setiap schema construct memiliki serangkaian instance sendiri. Sejumlah *database* state dapat dibuat berdasarkan skema *database* tertentu [3].

Ketika membuat sebuah *database* baru, DBMS hanya akan menerima skema *database* saja. Pada saat ini, *database* state kosong tanpa data. State awal akan diperoleh ketika *database* diisi dengan data. Seterusnya, setiap kali terjadi pembaruan, akan didapatkan *database* state lainnya. DBMS bertanggung jawab untuk memastikan setiap state dari *database* valid, atau sesuai struktur dan batasan yang dispesifikasikan dalam skema.

## Database Management System

*Database management system* (DBMS) adalah program yang memungkinkan seseorang untuk membuat dan mengatur sebuah *database*. DBMS memberikan fasilitas dalam proses pendefinisian, konstruksi, manipulasi, dan penyediaan *database* untuk berbagai pengguna dan aplikasi, atau disebut *sharing*. Proses pendefinisian melibatkan spesifikasi jenis data, struktur, dan batasan data untuk disimpan dalam *database*. Informasi definisi *database* ini disimpan oleh DBMS dalam bentuk katalog *databse*, yang disebut juga meta-data. Proses konstruksi *database* yaitu proses penyimpanan data dalam beberapa media penyimpanan yang diatur oleh DBMS. Proses manipulasi melibatkan beberapa fungsi seperti melakukan *query* pada *database* untuk mengambil suatu data, memperbarui *database*, dan membuat laporan dari data. *Sharing* memungkinkan pengguna atau program untuk mengakses *database* secara terus-menerus [3].

Sebuah program yang menggunakan *database* melakukan proses transaksi dengan DBMS. Program mengakses *database* dengan mengirimkan *query* atau permintaan data pada DBMS. Transaksi antara program dan DBMS memungkinkan terjadinya pembacaan data dari *database*, ataupun penulisan data ke dalam *database*.

Fungsi penting lain dari DBMS adalah perlindungan dan pemeliharaan *database* untuk jangka panjang. Perlindungan melibatkan perlindungan sistem dari kerusakan perangkat keras maupun lunak, dan pengamanan dari akses yang tidak disetujui. *Database* berukuran besar umumnya digunakan untuk bertahun-tahun, sehingga DBMS harus dapat memelihara sistem *database* dengan melakukan perubahan terhadap sistem sesuai kebutuhan yang berubah-ubah seiring waktu.

Ada banyak jenis DBMS yang saat ini beredar. Mulai dari yang bersifat komersial atau berpaten, maupun yang bersifat *open source*. DBMS berpaten misalnya saja Oracle. Sementara untuk DBMS *open source* yang paling banyak digunakan adalah MySQL dan PostgreSQL. Kedua DBMS ini memiliki kelebihan tersendiri untuk aplikasi-aplikasi tertentu, dan kemampuannya pun cukup memadai.

MySQL pertama kali dibuat dan dikembangkan di Swedia oleh David Axmark, Allan Larsson dan Michael “Monty” Widenius yang bekerjasama sejak tahun 1980 [10]. Dengan kecepatan, keandalan, dan kemudahan akses, MySQL telah menjadi pilihan dalam pembuatan Web, Web2.0, SaaS, ISV, dan juga bagi perusahaan-perusahaan telekomunikasi.

MySQL merupakan bagian dari LAMP. LAMP merupakan akronim dari Linux, Apache, MySQL, PHP/Perl/Python, yaitu *enterprise* *software stack* yang bersifat *open source*. Banyak perusahaan yang meggunakan LAMP sebagai alternatif dari *software* *stack* berpaten karena biayanya yang rendah dan juga tidak terbatas *platform*.

Organisasi besar dan berkembang banyak yang menggunakan MySQL untuk menghemat waktu dan biaya yang diperlukan dalam menyediakan Website bervolume besar, sistem bisnis kritikal, dan juga *packaged software* untuk industri besar seperti Yahoo!, Alcatel-Lucent, Google, Nokia, YouTube, Wikipedia, dan Booking.com [10].

PostgreSQL merupakan *object-relational* DBMS (ORDBMS) yang awalnya dikembangkan oleh University of California, Berkley, dengan nama proyek POSTGRES. Proyek ini dimulai oleh Profesor Michael Stonebraker pada tahun 1986 untuk menggantikan DBMS Ingres. Proyek POSTGRES digunakan untuk berbagai keperluan dalam organisasi yang berbeda-beda, seperti untuk *database* pelacakan asteroid, sistem analisis data finansial, dan juga sarana belajar [11].

PostgreSQL merupakan ORDBMS *opensource* yang *powerful* dan memiliki arsitektur yang diakui untuk keandalan, integritas data, dan juga tingkat kebenarannya. PostgreSQL telah mendapat pengakuan dari pengguna dan juga industri, seperti penghargaan sebagai Best *Database* System dari Linux New Media Award dan juga lima kali mendapat penghargaan sebagai DBMS terbaik dari The Linux Journal Editor’s Choice Award [12].

MySQL selama ini dikenal sebagai DBMS dengan kecepatan yang baik namun memiliki fitur yang kurang lengkap. Sementara, PostgreSQL dikenal dengan fitur DBMS yang lebih padat, bahkan dianggap sebagai versi *open source* dari Oracle.

MySQL dan PostgreSQL menawarkan jenis fleksibilitas yang berbeda. MySQL didesain dengan konsep bahwa aplikasi merupakan logika, dan *database* menyediakan tempat penyimpanan untuk aplikasi. Sementara PostgreSQL didesain dengan konsep bahwa *database* itu sendiri merupakan alat untuk membuat model, dan aplikasi berinteraksi dengannya menggunakan API yang didefinisikan oleh SQL.

### DBMS Language

Kerja DBMS diatur oleh tiga bahasa berbeda, yaitu *schema language* yang terdiri dari [5]:

* *Data definition language* (DDL): digunakan untuk membangun model informasi untuk dikelola oleh *database*. Artinya, DDL adalah bahasa yang mendeskripsikan skema-skema yang berbeda, terutama skema konseptual.
* *Data manipulation language* (DML): digunakan untuk memasukkan dan memanipulas data yang dikelola DBMS. Termasuk pembuatan dataset baru, memasukkan informasi baru ke dalam dataset, dan mengubah data di dalam dataset.
* *Data query language*: merupakan bagian dari DML, dan digunakan untuk memfasilitasi query. Pengguna atau aplikasi dapat membuat query pada data di dalam DBMS untuk mendapatkan informasi dari *database*. Query biasanya melibatkan lebih dari satu dataset di dalam DBMS, dan mungkin saja memberikan informasi yang tidak secara langsung ada di salah satu dataset.

### Struktur Komponen DBMS

Sistem *database* terbagi atas beberapa modul yang memiliki tanggungjawab tersendiri terhadap keseluruhan sistem. Gambar 2.6 menunjukkan gambaran struktur dari DBMS. Desain dari sistem *database* harus mempertimbangkan antarmuka di antara sistem *database* dengan sistem operasi. Komponen fungsional dari sistem *database* mencakup [4]:

data files

data dictionary

file manager

database manager

application programs object coder

data manipulation language precompiler

query processor

data definition language compiler

*database* scheme

query

application programs

application interface

naïve users

application programmers

sophisticated users

*database* administrator

users

disk storage

database management system

Gambar 2.6 Struktur Komponen DBMS [4]

* *File manager*: mengelola alokasi ruang pada disk storage dan mengelola data struktur yang digunakan untuk merepresentasikan informasi yang tersimpan dalam disk.
* *Database manager*: menyediakan antarmuka antara data low-level yang tersimpan dalam *database* dengan program dari aplikasi dan query yang disampaikan ke sistem.
* *Query processor*: menerjemahkan pernyataan pada query language menjadi instruksi low-level yang dapat dimengerti *database* manager.
* *Data manipulation language (DML) precompiler*: konversi pernyataan DML yang tertanam dalam suatu program aplikasi menjadi panggilan prosedur normal dalam bahasa host. Precompiler berinteraksi dengan query processor untuk menghasilkan kode yang tepat.
* *Data definition language (DDL) compiler*: konversi pernyataan DDL menjadi serangkaian tabel yang berisikan metadata, atau “data tentang data”.
* *Data file*: menyimpan *database*.
* *Data dictionary*: menyimpan metadata mengenai struktur *database*.
* *Indices*: menyediakan akses cepat ke data dengan informasi tertentu.

### Database Query Language

Bahasa *query* merupakan sebuah bahasa komputer yang digunakan untuk meminta, atau membuat *query*, informasi dari *database*.

#### SQL

SQL merupakan kependekan dari *Structured Query Language*. SQL didesain dan diimplementasikan di IBM Research sebagai antarmuka untuk eksperimen sistem *relational database* yang disebut SYSTEM R. Saat ini SQL merupakan bahasa standar untuk *relational* DBMS komersial.

SQL memiliki pernyataan untuk pendefinisian, *query*, dan pembaruan data. Selain itu, juga terdapat fasilitas untuk mendefinisikan tampilan pada *database*, mengatur keamananan dan *authorization*, memberikan batasan integritas, dan untuk mengatur transaksi. SQL juga memiliki aturan sendiri untuk ditanam pada bahasa pemrograman seperti Java, Cobol, atau C/C++ [3].

Dalam SQL terdapat tabel, baris, dan kolom yang merepresentasikan hubungan, *tuple*, dan atribut pada model *relational*. Perintah utama dalam SQL untuk mendefinisikan data adalah pernyataan CREATE. CREATE dapat digunakan untuk membuat skema, tabel, dan domain.

Perintah CREATE TABLE digunakan untuk membuat sebuah hubungan baru dengan memberi nama, atribut, dan batasan. Setiap atribut diberikan nama, jenis data, dan batasan-batasan atribut. *Key*, *entity* *integrity*, dan *referential integrity constraint* dapat dibuat di dalam CREATE TABLE setelah atribut-atribut dideklarasikan, atau menggunakan perintah ALTER TABLE.

SQL memiliki pernyataan dasar untuk mengambil informasi dari *database*, yaitu pernyataan SELECT. *Retrieval query* pada SQL dapat terdiri dari empat klausa, namun hanya dua, yaitu SELECT dan FROM, yang harus digunakan. Klausa SELECT berisikan atribut yang ingin diambil, dan klausa FROM berisikan tabel dimana atribut berada. Klausa WHERE digunakan untuk memilih *tuple* dengan kondisi tertentu, dan ORDER BY mengurutkan hasil dari *query*.

Dalam SQL terdapat tiga perintah untuk memodifikasi *database*, yaitu INSERT, DELETE, dan UPDATE. INSERT digunakan untuk menambahkan sebuah *tuple* ke dalam sebuah tabel. Pada perintah INSERT harus diberikan nama tabel dan juga nilai *tuple* yang sesuai dengan atribut dari tabel. DELETE digunakan untuk menghapus *tuple* dari sebuah tabel. Klausa WHERE juga digunakan pada perintah ini untuk memilih *tuple* yang akan dihapus. Jika tidak menggunakan WHERE, maka semua *tuple* dalam tabel akan terhapus dan hanya tersisa tabel kosong. UPDATE digunakan untuk memodifikasi nilai atribut dari satu atau lebih tuple. Klausa WHERE juga digunakan pada perintah ini untuk memilih tuple yang akan dimodifikasi dalam sebuah tabel. Klausa SET digunakan dalam perintah UPDATE untuk memberikan nilai baru pada sebuah atribut.

#### OQL

*Object Query Language* (OQL) merupakan bahasa *query* yang didesain untuk bekerja dengan bahasa pemrograman untuk *Object Data Management Group* (ODMG) seperti C++, Smalltalk, dan Java. *Syntax query* OQL mirip dengan *syntax* standar pada bahasa *query* SQL, dengan tambahan fitur untuk konsep ODMG, seperti *object identity*, *complex object*, *operation, inheritance*, *polymorphism*, dan *relationship* [3].

#### XPath dan XQuery

*Query* XML memiliki dua bahasa standar, yaitu XPath dan XQuery. XPath menyediakan konstruksi bahasa yang memberikan ekspresi *path* untuk mengidentifikasi elemen atau atribut tertentu dalam dokumen XML. XQuery merupakan bahasa query yang lebih umum. XQuery menggunakan ekspresi XPath dengan beberapa konstruksi tambahan [3].

Sebuah ekspresi XPath memberikan urutan *item* yang sesuai dengan pola yang dispesifikasikan pada ekspresi. *Item* ini dapat berupa nilai dari elemen atau atribut. Nama pada ekspresi XPath adalah nama-nama *node* dalam pohon XML baik nama *tag* (elemen) ataupun nama atribut, dengan tambahan kondisi untuk membuat node sesuai dengan pola. Terdapat dua buah tanda baca utama dalam penulisan *path*, yaitu *single slash* (/) dan *double slash* (//). *Single slash* sebelum sebuah *tag* menandakan bahwa *tag* tersebut merupakan *direct child* dari *tag* sebelumnya, sedangkan *double slash* menandakan sebuah *tag* merupakan *descendant* dari *tag* sebelumnya.

Bentuk umum dari *query* pada XQuery dikenal sebagai ekspresi FLWR, yang terdiri dari empat klausa, yaitu FOR, LET, WHERE, dan RETURN [3]. Dalam sebuah XQuery, klausa FOR dapat berisi beberapa *instance* atau kosong, begitu pula LET. Klausa WHERE merupakan klausa tambahan tetapi dapat muncul paling tidak satu kali, sementara klausa RETURN harus muncul satu kali.

XQuery memiliki konstruksi yang kuat untuk membentuk *query* yang kompleks. Lebih jelasnya, XQuery dapat membuat bilangan universal dan eksistensial dalam sebuah kondisi dari *query*, fungsi *aggregate*, mengurutkan hasil query, memilih berdasarkan posisi dalam sebuah urutan, bahkan membuat kondisi bercabang.

## Multimedia Database

*Multimedia database* adalah kumpulan data multimedia yang terkendali. *Multimedia database* terdiri dari jenis data yang berbeda-beda seperti teks, gambar, objek grafik, urutan animasi, video dan audio. Perbedaan jenis data tentunya membutuhkan cara khusus untuk mengoptimalkan penyimpanan, akses, *indexing*, dan pengambilan. Sebuah *multimedia database management system* (MMDBMS) menyediakan lingkungan yang sesuai untuk menggunakan dan mengolah *database* multimedia [2].

Objek multimedia memiliki hubungan yang bersifat temporal dan spasial yang harus diperhitungkan untuk sinkronisasi dan penampilan informasi. Hubungan ini harus dimodelkan secara eksplisit sebagai bagian dari data yang tersimpan. Dengan demikian, meskipun data multimedia tersimpan dalam *file*, hubungan antar data multimedia harus disimpan sebagai bagian dari metadata dalam DBMS.

Hubungan spasial antar objek individual dalam gambar atau *frame* video menghasilkan beberapa batasan ketika mencari objek dalam *database*. Model data *relational* ataupun *object-oriented* telah diajukan untuk mengatasi batasan dalam *database* multimedia ini.

### Syarat dan Arsitektur MMDBMS

Arsitektur umum MMDBMS ditunjukkan pada Gambar 2.7. Untuk mengembangkan beragam aplikasi *multimedia database* dibutuhkan lingkungan pengembangan *hypermedia* yang kuat dan fleksibel. Ada beberapa syarat khusus untuk *database* multimedia, seperti [2]:

* Kemampuan dasar DBMS
* Manajemen penyimpanan berkapasitas besar
* Kemampuan mengambil informasi
* Mendukung query multimedia
* Intergrasi media, komposisi dan presentasi
* Antaramuka multimedia dan *interactivity*

### Query dalam Database Multimedia

*Query* merupakan salah satu hal paling penting dalam DBMS. *Query* *language* untuk multimedia harus dapat mengatasi hubungan spasial dan temporal kompleks yang diwarisi ke dalam jenis data multimedia. *Query language* yang baik dapat membantu memanipulasi MMDBMS dan menjaga independensi antara *database* dan aplikasi. *Query* data multimedia dapat dibagi dalam beberapa jenis [2]:

* *Keyword querying*
* *Semantic querying*
* *Visual querying*
* *Video querying*

Presentation Manager

Interface Manager

Configuration Manager

Query Manager

Media Manager

Transaction Manager

Context base manager

Semantic Information manager

Multimedia DBMS Engine

Data Migration Manager

*Database*

Gambar 2.7 Arsitektur Multimedia DBMS [2]

## PHP dan Web

PHP (PHP: *Hypertext Preprocessor*) adalah bahasa pemrograman yang berfungsi untuk membuat Website dinamis maupun aplikasi Web [6]. PHP dapat berinteraksi dengan *database*, *file* dan *folder*, sehingga membuat PHP bisa menampilkan konten yang dinamis dari sebuah Website. PHP adalah bahasa *scripting* yang dapat berjalan pada sistem operasi yang berbeda-beda (*cross-platform*) seperti Windows, Linux, dan Mac. Saat ini PHP merupakan *server-side scripting* yang paling banyak digunakan. Program PHP ditulis dalam berkas *plain text* dengan format \*.php.

Untuk dapat berjalan, PHP membutuhkan *web server*. *Web server* bertugas memproses berkas PHP dan mengirimkan hasil proses untuk keperluan halaman Web, ataupun sebaliknya menerima masukan dari halaman Web kemudian mengirimkan hasil proses ke *server*. Oleh karena itu, PHP termasuk *server-side scripting*.

*Web server* adalah perangkat lunak yang berfungsi untuk melayani berbagai permintaan Web dari *client* yang berada dalam jangkauan jaringannya. Selain *Web server*, dalam membangun sebuah Web juga dibutuhkan *database server*, yaitu *server* yang berisikan perangkat lunak *database* (DBMS) untuk menyimpan data-data yang dibutuhkan Web.

## Semantic Web

*Semantic Web* merupakan sebuah peningkatan mutu dari Web sehingga komputer dapat memproses informasi yang ada di *World Wide Web* (WWW), menginterpretasi dan menghubungkannya untuk membantu manusia menemukan pengetahuan yang dibutuhkannya. *Semantic Web* dimaksudkan untuk membentuk suatu pendistribusian pengetahuan berbasis sistem. Tujuan dari *semantic Web* adalah untuk berbagi data, bukan dokumen. Dengan kata lain *Semantic Web* menyediakan *framework* yang memungkinkan data untuk dibagi dan digunakan kembali tanpa batasan aplikasi, perusahaan, ataupun komunitas. *Semantic Web* merupakan upaya kolaboratif dari *World Wide Web Consortium* (W3C) [7].

Konsep pengetahuan yang ada dalam *semantic* *Web* dijabarkan oleh ontologi. Ontologi mencoba menjelaskan mengenai struktur dan hubungan yang mungkin dalam realita melalui beberapa kosakata biasa. Dapat pula diartikan sebagai sebuah cara untuk menggambarkan pengetahuan dari sebuah komunitas tertentu tentang realita. Ontologi berasal dari bidang filosofi dan metafisik. Salah satu definisi umum dari ontologi adalah spesifikasi dari konseptualisasi.

Konseptualisasi merupakan serangkaian konsep yang digunakan untuk menunjukkan bagian dari realita atau pengetahuan yang merupakan hal menarik bagi suatu komunitas pengguna. Spesifikasi merujuk pada bahasa dan kosakata yang digunakan untuk menentukan konseptualisasi. Ontologi terdiri dari spesifikasi dan konseptualisasi. Sebagai contoh, konseptualisasi yang sama dapat dispesifikasikan dalam dua bahasa berbeda, memberikan dua ontologi terpisah. Berdasarkan gambaran umum ini, tidak ada kesepakatan apakah ontologi sebenarnya. Beberapa cara yang mungkin dapat menjelaskan ontologi adalah sebagai berikut[3]:

* *Thesaurus*, atau kamus istilah, menjelaskan hubungan antara kata yang menunjukkan berbagai konsep.
* *Taxonomy* menjelaskan bagaimana konsep dari sebuah area pengetahuan khusus terhubung menggunakan struktur seperti yang digunakan dalam pengkhususan atau generalisasi.
* Sebuah skema *database* yang detil dipertimbangkan sebagian orang sebagai sebuah ontologi yang menjelaskan konsep, entitas dan atribut, dan hubungan dari sebuah *miniworld* dengan realita.
* Sebuah teori logika menggunakan konsep dari logika matematika untuk menjelaskan konsep dan hubungan timbal baliknya.

Biasanya konsep yang digunakan untuk menggambarkan ontologi hampir sama dengan konsep yang dibahas dalam pemodelan konseptual dalam  *database*, seperti entitas, atribut, hubungan, pengkhususan, dan sebagainya. Perbedaan utama antara ontologi dan skema *database* adalah, skema biasanya terbatas hanya untuk menggambarkan bagian kecil sebuah *miniworld* yang bertujuan untuk menyimpan dan mengolah data. Ontologi bertindak lebih umum dalam menggambarkan sebuah bagian dari realita atau area kekhususan selengkap-lengkapnya.

Dalam pengembangan *semantic Web* dikenal sebuah istilah *Resource Description Framework* (RDF). RDF adalah sebuah bahasa untuk merepresentasikan informasi mengenai sumber dalam WWW. Lebih jelasnya, RDF dimaksudkan untuk merepresentasikan metadata dari sumber Web, seperti judul, penulis, tanggal modifikasi Web, informasi *copyright* dan lisensi tentang sebuah dokumen Web, atau jadwal ketersediaan untuk beberapa *shared resource* [8].

RDF digunakan ketika informasi perlu diproses lebih lanjut oleh sebuah aplikasi, bukan sekadar ditampilkan saja. RDF memiliki *framework* umum untuk mengekspresikan informasi sehingga antar aplikasi dapat saling bertukar informasi tanpa kehilangan arti informasi. Kemampuan untuk bertukar informasi antar aplikasi ini memungkinkan sebuah informasi untuk dapat digunakan pada aplikasi mana saja, bukan hanya untuk aplikasi asal informasi.

RDF didasari dari ide untuk mengidentifikasi suatu hal menggunakan *Web identifier*, yang disebut *Uniform Resource Identifier* (URI), dan mendeskripsikan sumber dalam bentuk properti sederhana dan nilai properti. Hal ini memungkinkan RDF untuk memberikan pernyataan sederhana mengenai sumber dalam bentuk grafik *node* dan *arc* yang merepresentasikan sumber, serta properti dan nilainya. RDF memiliki *syntax* berbasis XML untuk merekam dan bertukar sumber.

## Video Interaktif

Video interaktif adalah video yang dapat dimainkan seperti video biasa, tetapi memiliki *clickable area*, atau “hotspot”, yang dapat melakukan sesuatu ketika area tersebut diklik [9]. Video interaktif telah banyak digunakan untuk berbagai keperluan, seperti dalam edukasi, iklan, presentasi, katalog belanja, dan dunia hiburan. Fasilitas klik pada video merupakan salah satu aspek interaktif yang membuat penonton dapat berinteraksi, memberikan suatu informasi atau respon kepada sistem video. Video kemudian akan melakukan hal sesuai dengan respon yang dimasukkan oleh penonton, sehingga terjadilah komunikasi dua arah antara penonton dengan sistem video.

Jenis interaktif yang ditawarkan oleh aplikasi video interaktif saat ini sangat beragam. Yang paling sederhana misalnya pemberian catatan, atau *annotation*, pada saat-saat tertentu dalam video. *Clickable area* dapat divariasikan untuk banyak hal, misalnya memberikan sebuah informasi tambahan mengenai *hotspot* dalam video ketika *clickable area* diklik. *Clickable* *area* juga dapat dijadikan *link* menuju suatu halaman Web, jika aplikasi video interaktif bersifat *online*, atau bisa juga untuk melakukan *jump* ke waktu-waktu tertentu di video.

# PERANCANGAN SISTEM UNTUK VIDEO STREAMING SEMANTIK INTERAKTIF

Pada bab ini akan dijabarkan perancangan sistem mulai dari diagram sistem, ontologi, dan perancangan *database*.

## Gambaran Umum Sistem Video Interaktif Berbasis Web

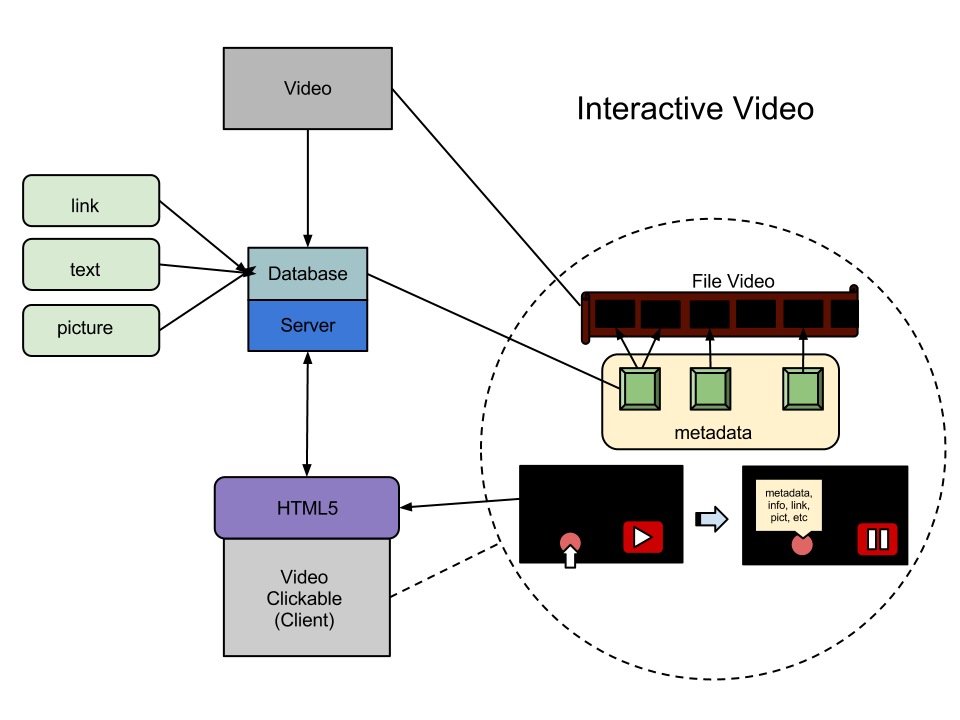
Sistem video semantik interaktif yang dirancang ini bernama Invidance, yang berasal dari *Interactive Video of Indonesian Traditional Dance*. Logo Invidance ditunjukkan oleh Gambar 3.1 Invidance merupakan sebuah Web yang di dalamnya menyajikan video interaktif bertemakan tari tradisional Indonesia. Interaksinya berupa *clickable icon* yang muncul pada video untuk memberi penonton informasi yang sesuai dengan *scene* video ketika diklik.



Gambar 3.1 Logo Invidance

Gambar 3.2 menunjukkan gambaran umum dari keseluruhan sistem video interaktif berbasis Web, yang diberi nama Invidance. Video disimpan di dalam sebuah sistem *server*, yang mana terdapat *database* untuk menyimpan berbagai informasi mengenai video. Video disajikan kepada pengguna melalui sebuah halaman Web. Dari halaman Web, pengguna dapat memainkan ataupun menghentikan video. Pada saat menonton video, pengguna dapat menemukan beberapa *icon*, yang mana area ini dapat diklik kemudian memunculkan informasi mengenai hal yang diwakilkan oleh *icon*.

Tiap *icon* yang muncul pada video menyimpan kode informasi yang berkaitan dengan hal yang ditampilkan pada video pada waktu tertentu. Pada saat diklik, sistem akan melakukan cek ke *database*, informasi apakah yang harus dimunculkan, dan juga mencari informasi terkait menggunakan kata kunci. Saat sudah ditemukan, informasi tersebut kemudian disajikan ke pengguna dalam berbagai bentuk, seperti teks, gambar, video, maupun *link*.



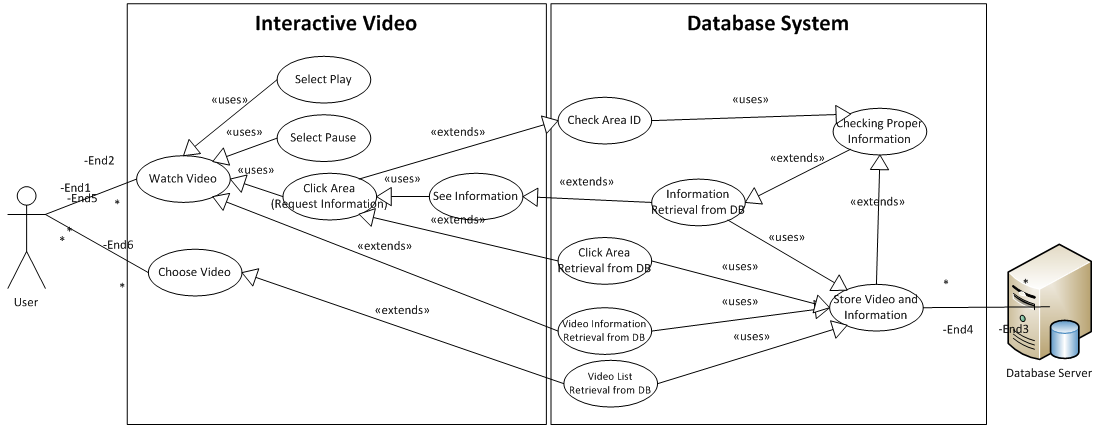
Gambar 3.2 Gambaran Umum Sistem Invidance

### Rancangan Sistem

Secara umum sistem Invidance terdiri dari dua bagian, yaitu sistem *admin* untuk membuat video interaktif, dan sistem *viewer* untuk menampilkan aplikasi video interaktif. Di dalam Invidance terdapat sistem *database* untuk menyimpan dan mengatur informasi yang harus ditampilkan pada video.

*Database* Invidance menyimpan data informasi yang akan ditampilkan pada halaman Web Invidance. Untuk berkas multimedia seperti video dan gambar tidak disimpan langsung di dalam *database*, tapi diletakkan di *file system*. Data yang memiliki ukuran besar seperti video dan gambar kurang cocok jika dimasukkan ke dalam *database*, karena akan memperlambat proses pengambilan dan pengolahan data.

*Database* menyimpan properti dari berkas multimedia, seperti nama berkas, ukuran, jenis berkas, lokasi berkas, dan kode identitas informasi. Properti multimedia ini digunakan sebagai referensi untuk penyajian berkas multimedia pada Web. *Database* juga menyimpan kata kunci sebagai referensi informasi terkait. Gambar 3.3 menunjukkan diagram *use case* untuk sistem Invidance.



Gambar 3.3 Use Case Diagram

### Alur Sistem

Sistem *admin* dan sistem *viewer* memiliki mekanisme manipulasi *database* yang berbeda. Sistem *admin* bersifat *push*, dalam artian pengguna atau *admin* berperan sebagai pengisi informasi, kemudian sistem akan memasukkan informasi ke dalam *database* sesuai tempatnya. Sementara sistem *viewer* bersifat *pull*, dimana ketika pengguna menjalankan halaman *viewer*, sistem akan menarik berbagai informasi yang diperlukan dari *database* untuk menyajikan video interaktif.

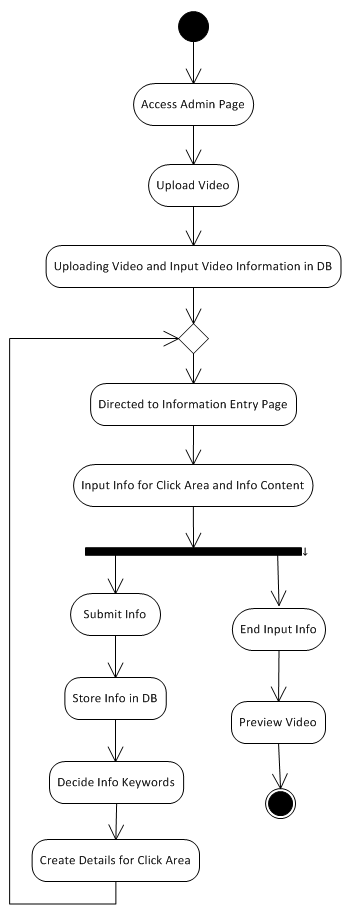
#### Sistem Admin

Gambar 3.4 menunjukkan diagram dari kerja sistem *admin*. Halaman *admin* diawali dengan form untuk mengunggah video. Video yang diunggah merupakan video yang ingin dibuat menjadi video clickable, atau video interaktif. Ketika *admin* mengunggah video, sistem akan mengambil informasi properti video seperti format, ukuran, dan juga memberikan nama video. Properti-properti video ini kemudian dimasukkan ke dalam *database*.

Setelah video selesai diunggah, masuk ke tahap memasukkan informasi. *Admin* dapat memasukkan informasi melalui *form* *input* informasi. Melalui *form* ini *admin* dapat membuat *clickable icon* pada video, dan juga mengisi informasi ketika *icon* diklik.

Untuk membuat *clickable icon*, diperlukan properti waktu muncul, koordinat, dan juga *layer* *clickable icon*. Pada sistem Invidance *admin* dimungkinkan untuk memasukkan tiga *clickable icon* pada waktu yang bersamaan, yaitu dengan membedakan *layer* *clickable icon*. Untuk konten informasi yang akan muncul di *pop-up window*, *admin* dapat mengisi judul informasi, isi informasi, dan juga gambar.

Jika *admin* sudah selesai memasukkan satu informasi, maka sistem akan memproses *input* dari *admin* dan meletakkannya pada lokasi tertentu di dalam *database*. Pada saat ini juga terjadi proses pencarian kata kunci. Pencarian dilakukan dengan membandingkan kata kunci yang disediakan sistem dengan kata per kata dalam isi informasi yang dimasukkan *admin*.



Gambar 3.4 Activity Diagram Sistem *Admin*

#### Sistem Viewer

Saat halaman Web Invidance dibuka, berkas video interaktif akan diambil dari *server* untuk ditampilkan. Pengguna kemudian dapat memulai menonton video. Di dalam video terdapat beberapa *clickable icon* pada waktu-waktu tertentu.

Pada saat pengguna mengklik *clickable icon*, sistem akan mengirimkan informasi identitas *clickable icon* ke sistem *database*. Identitas ini kemudian digunakan untuk mengecek konten informasi apakah yang harus ditampilkan untuk *clickable icon* tersebut. Sistem akan mengirim kembali konten informasi yang harus ditampilkan, untuk kemudian disajikan melalui *pop-up window* informasi. *Pop-up* *window* ini berisikan judul informasi, teks yang menjelaskan informasi, dan bisa juga terdapat gambar yang terkait dengan informasi. Gambar 3.5 merupakan *activity diagram* dari sistem *viewer* Invidance.



Gambar 3.5 Activity Diagram Sistem *Viewer*

## Perancangan Sistem Database

*Database* dirancang menggunakan model data relasional. Pada perancangan *database* digunakan PostgreSQL sebagai DBMS. PostgreSQL merupakan DBMS yang cukup tangguh untuk menangani *database* dengan *entry* yang banyak, dimana pada *database* video interaktif ini melibatkan properti berkas-berkas multimedia untuk ditampilkan pada Web.

### Normalisasi Database

Normalisasi adalah sebuah proses atau serangkaian aturan yang digunakan untuk merancang *database* secara optimal untuk mengurangi data yang mubazir [13]. Pada *database* yang belum dinormalisasi mungkin saja terdapat data yang dimiliki oleh lebih dari satu tabel berbeda tanpa alasan yang jelas. Hal ini kurang baik untuk keamanan, penggunaan ruang penyimpanan, kecepatan *query*, efisiensi pembaharuan *database*, dan yang paling penting ialah integritas data.

*Database* yang belum dinormalisasi adalah *database* yang belum dipisah secara logika menjadi tabel yang lebih kecil dan mudah dikelola. *Raw database* Invidance ditunjukkan oleh Gambar 3.6.

INVIDANCE\_DATABASE

video\_id info\_id

video\_title info\_title

video\_format info\_keywords

video\_height image\_id

video\_width image\_title

video\_dir image\_format

area\_id image\_height

pos\_x image\_width

pos\_y image\_dir

s\_time text\_id

e\_time text\_title

layer text\_content

icon keyword

Gambar 3.6 *Raw Database* Invidance

Proses normalisasi pertama, atau untuk mencapai *first normal form* (1NF) dilakukan dengan memecah *database* menjadi beberapa bagian yang memiliki keterkaitan informasi. Tiap tabel memiliki *primary key*, dan tidak ada sekelompok atribut yang mengulang di tabel manapun. Gambar 3.7 menunjukkan kondisi 1NF dari *database* Invidance.

INVIDANCE\_DATABASE

video\_id info\_id

video\_title info\_title

video\_format info\_keywords

video\_height image\_id

video\_width image\_title

video\_dir image\_format

area\_id image\_height

pos\_x image\_width

pos\_y image\_dir

s\_time text\_id

e\_time text\_title

layer text\_content

icon keyword

CLICKABLE\_VIDEO

video\_id

video\_title

video\_format

video\_height

video\_width

video\_dir

area\_id

pos\_x

pos\_y

s\_time

e\_time

layer

icon

INFO

info\_id

info\_title

keywords

image\_id

image\_title

image\_format

image\_height

image\_width

image\_dir

text\_id

text\_title

text\_content

Gambar 3.7 *First Normal Form*

Normalisasi ke-dua, atau *second normal form* (2NF), bertujuan untuk memisahkan data yang hanya tergantung sebagian pada *primary key*, dan meletakannya di tabel lain. Kondisi 2NF ditunjukkan oleh Gambar 3.8.

CLICKABLE\_VIDEO

video\_id

video\_title

video\_format

video\_height

video\_width

video\_dir

area\_id

pos\_x

pos\_y

s\_time

e\_time

layer

icon

INFO

info\_id

info\_title

info\_keywords

image\_id

image\_title

image\_format

image\_height

image\_width

image\_dir

text\_id

text\_title

text\_content

keyword

CLICK\_AREA

area\_id

pos\_x

pos\_y

s\_time

e\_time

layer

icon

video\_id

VIDEO

video\_id

video\_title

video\_format

video\_height

video\_width

video\_dir

INFO

info\_id

info\_title

info\_keywords

image\_id

text\_id

keyword

IMAGE

image\_id

image\_title

image\_format

image\_height

image\_width

image\_dir

TEXT

text\_id

text\_title

text\_content

Gambar 3.8 *Second Normal Form*

Normalisasi ke-tiga, atau *third normal form* (3NF), bertujuan untuk memindahkan data yang tidak terkait sama sekali dengan *primary key*. Dari hasil 2NF, tabel CLICK\_AREA, VIDEO, IMAGE, dan TEXT sudah tidak memiliki atribut yang sama sekali tidak bergantung pada *primary key*-nya. Pada tabel INFO masih terdapat atribut keyword yang tidak bergantung pada *primary key* info\_id. Gambar 3.9 menunjukkan kondisi 3NF untuk tabel INFO.

INFO

info\_id

info\_title

info\_keywords

image\_id

text\_id

keyword

INFO

info\_id

info\_title

info\_keywords

image\_id

text\_id

KEYWORD

keyword\_id

keyword

Gambar 3.9 *Third Normal Form* Tabel INFO

### Struktur Database

Setelah dilakukan normalisasi tabel dari entitas-entitas pada sistem Invidance, maka didapatkan enam tabel untuk sistem *database* Invidance. Tabel-tabel tersebut yaitu click\_area, info, video, text, image, dan keyword. Tabel click\_area mendefinisikan *clickable area* yang dimunculkan pada saat video dimainkan. *Clickable area* yang dimunculkan dalam video memiliki atribut pos\_x dan pos\_y yang memberikan posisi area, s\_time dan e\_time yang menentukan kapan *clickable icon* akan muncul dan hilang, dan juga *icon* yang berisikan lokasi gambar *icon* pada sistem. Tiap *clickable icon* menyimpan info\_id yang merujuk pada informasi mengenai hal yang disorot oleh *clickable icon*.

Tabel info mendefinisikan konten informasi untuk setiap set informasi yang ditampilkan pada *pop-up window*. Atribut info\_id memberikan set informasi sebuah identitas unik sebagai pengenalnya dalam pengolahan informasi, sementara info\_title merupakan judul dari set informasi yang juga akan ditampilkan pada *pop-up window*. Atribut image\_id dan text\_id berisikan kode untuk konten berupa gambar dan teks yang merupakan bagian dari set informasi. Atribut keyword berisikan kata kunci dari informasi yang didapatkan berdasarkan pencarian per kata dari konten informasi teks.

Fungsi pencarian keyword dari konten informasi teks menggunakan tabel keyword sebagai referensi kosakata yang harus dicocokkan. Tabel ini berisikan kata kunci yang sudah disesuaikan dengan tema informasi Invidance, yaitu mengenai tari tradisional. Setiap set informasi yang didefinisikan pada tabel info dapat memiliki beberapa kata kunci dari tabel keyword ini. Keterkaitan informasi akan dilihat dari seberapa cocok kata kunci antar set informasi.

Tabel untuk objek media terdiri dari video, text, dan image. Tabel objek media dapat dikatakan sebagai sumber informasi yang dimunculkan kepada pengguna. Sumber informasi ini menyimpan properti-properti yang mendukung informasi untuk dapat ditampilkan oleh sistem. Misalnya untuk tabel image, terdapat atribut image\_dir yang berisi lokasi gambar di dalam *server*, sehingga gambar dapat di*retrieve* dari lokasi yang tepat.

1

1

1

image\_id

image\_title

image\_format

image\_height

image\_width

image\_dir

text\_id

text\_title

text\_content

video\_id

video\_title

video\_format

video\_height

video\_width

video\_dir

area\_id

info\_id

pos\_x

pos\_y

s\_time

e\_time

layer

icon

keywords

keyword\_id

1

1

1

1

M

Gambar 3.10 ERD Database Invidance

Keterhubungan antar tabel entitas pada *database* ditunjukkan oleh Gambar 3.10. Entitas info terhubung dengan entitas click\_area, text, dan image melalui atribut info\_id, text\_id, dan image\_id dengan keterhubungan *one-to-one*. Entitas video terhubung dengan entitas click\_area melalui video\_id dengan keterhubungan *one-to-many­*. Sementara entitas info sebenarnya tidak memiliki keterhubungan langsung dengan entitas keyword, karena entitas keyword hanya berupa tabel daftar kosakata untuk mengisi atribut keywords pada entitas info.

## Algoritma Sistem

Sistem *admin* dan sistem *viewer* Invidance terdiri dari beberapa modul. Secara umum, kerja sistem *admin* digambarkan oleh *pseudocode* pada Gambar 3.11, dan kerja sistem *viewer* digambarkan oleh *pseudocode* pada Gambar 3.12.

choose a video

upload video

insert video’s attributes into video table in database

redirected to input form

while not choose end input

input properties for clickable icon

input information for pop-up

submit form

insert click icon properties into table click\_area

insert text properties into table text

insert image properties into table image

search keywords

insert information set’s properties into database

end while

Gambar 3.11 *Pseudocode* Sistem *Admin* Invidance

get video\_id

retrieve video’s attributes

load video to Web

while video's time not end

while video played

if clickable icon available

show icon

if clickable icon clicked

pause video

get area\_id

get information contents

show information in pop-up

end if

end if

end while

end while

Gambar 3.12 *Pseudocode* Sistem *Viewer* Invidance

### Algoritma Sistem Admin

Sistem *admin* terdiri dari empat bagian penting, yaitu pengunggahan dan pendefinisian properti video, pendefinisian properti *clickable icon*, pendefinisian informasi *pop-up window*, dan pencarian kata kunci untuk set informasi.

#### Mengunggah Video

Seperti terlihat pada Gambar 3.13. untuk mengunggah video, *admin* pertama-tama harus memilih video. Ketika proses unggah dilakukan, sistem akan mengambil properti-properti dari video seperti format video dan dimensi video. Kemudian video diberi nama serta lokasi untuk menempatkan *file* video. *Temporary file* video lalu disalin ke direktori yang diberikan dengan nama baru. Setelah itu, semua properti video dimasukkan ke tabel video dalam *database*.

choose a video

upload video

get video’s properties (format, dimension)

define video’s name

define video’s directory

copy temporary video file into defined directory

insert all video’s properties into table video

Gambar 3.13 *Pseudocode* Unggah Video

#### Membuat Clickable Icon

get input icon’s position (x and y)

get input icon’s start time

get input icon’s duration

count end time (start time + duration)

get input icon’s layer

get input icon’s graphic

insert all icon properties into table click\_area

Gambar 3.14 *Pseudocode* Membuat *Clickable Icon*

Gambar 3.14 menunjukkan algoritma pembuatan *clickable icon* dalam *database*. Ketika user melakukan submit form input informasi, sistem akan mengambil *input* posisi x, posisi y, waktu muncul *icon*, durasi muncul *icon*, *layer icon*, dan gambar *icon* untuk dimasukkan ke dalam *database* sebagai properti dari *clickable icon*. Untuk waktu berakhir *clickable*, dihitung dari waktu awal munculnya *icon* ditambah dengan durasi *icon*, kemudian dimasukkan ke dalam *database* sebagai waktu akhir *icon*.

#### Membuat Informasi untuk Pop-up

Dapat dilihat pada Gambar 3.15, sistem akan memasukkan input judul dan konten informasi sebagai properti untuk text, kemudian mengambil text\_id yang baru saja dimasukkan. Jika *admin* mengunggah gambar, maka properti-properti gambar ini akan diletakkan di tabel image, kemudian diambil image\_id. Setelah selesai memasukkan properti text dan image ke dalam *database* dan mendapatkan kode untuk masing-masing objek media, sistem akan membentuk set informasi pada tabel info. Kode set informasi (info\_id) ini kemudian diambil dan dimasukkan sebagai salah satu properti *clickable icon* pada tabel click\_area, sebagai kode informasi yang akan keluar bila *icon* diklik.

get input info’s title

get input info’s content (text)

insert text pproperties into table text

get text\_id

upload image

get image’s properties (format, dimension)

define image’s name

define image’s directory

copy temporary image file into defined directory

insert all image’s properties into table image

get image\_id

insert text and image’s code into table info

Gambar 3.15 *Pseudocode* Membuat Informasi *Pop-up*

#### Mencari Kata Kunci Set Informasi

get input info’s content (text)

select all keyword from table keyword

while keyword available

match keyword with words in text

if keyword match

put keyword in array

end if

end while

insert keywords into table info

Gambar 3.16 *Pseudocode* Mencari Kata Kunci

Pencarian informasi terkait untuk tiap set informasi yang dimasukkan dilakukan dengan algoritma seperti terlihat pada Gambar 3.16. Setiap kata per kata dari isi informasi text yang dimasukkan di *database* akan dicocokkan dengan daftar kata kunci yang ada dalam tabel keyword. Jika satu atau beberapa kata kunci terdeteksi di dalam isi informasi teks, makan kata kunci ini diambil sebagai kata kunci dari set informasi, dan dimasukkan sebagai properti set informasi ke dalam tabel info.



Gambar 3.17 Ontologi Tari Tradisional pada Sistem Invidance

Daftar kata kunci yang ada di dalam *database* dibuat berdasarkan ontologi informasi tari tradisional yang ditunjukkan oleh Gambar 3.17. Penerapan ontologi seperti ini dalam sistem memungkinkan penggalian informasi yang lebih mudah dan “cerdas”. Hanya dari satu objek informasi yang dilihat, dapat digali informasi-informasi lebih mendalam mengenai objek tersebut. Selain itu, dapat pula ditemukan objek informasi lain yang memiliki kesamaan dengan objek informasi yang sedang dilihat.

### Algoritma Sistem Viewer

Pada sistem *viewer* dilakukan penarikan data-data dari *database* untuk dimanipulasi dan disajikan pada Web Invidance dalam bentuk informasi utuh. Proses yang terdapat pada sistem *viewer* ialah penarikan properti video untuk menampilkan video, penarikan properti *clickable icon* untuk memunculkan icon, penarikan properti konten informasi untuk ditampilkan pada *pop-up*, dan juga pencarian set informasi terkait berdasarkan kecocokan kata kunci set informasi.

#### Menampilkan Video

Gambar 3.18 menunjukkan *pseudocode* untuk menampilkan video. Pertama, sistem akan mencari video\_id di dalam tabel video. Properti-properti penting video seperti format, ukuran, dan letak video akan diambil untuk keperluan menampilkan video dalam Web.

find video\_id in table video

get video format, dimension, and directory

load video

Gambar 3.18 *Pseudocode* Menampilkan Video

#### Penarikan Properti Clickable Icon

Gambar 3.19 menunjukkan algoritma untuk menarik properti *clickable icon* dari *database* untuk ditampilkan pada video selama video berjalan. *Clickable icon* hanya akan muncul pada interval-interval waktu tertentu dan maksimum dapat muncul tiga *icon* sekaligus. Selama video dimainkan, sistem akan mencocokkan waktu berjalan dengan interval-interval *clickable icon*. Jika waktu berjalan sesuai, makan sistem akan mengambil properti *clickable icon* untuk kemudian dimunculkan pada video.

compare current time with clickable icon interval time

if current time >= start time AND < end time

get x and y position

get area layer (1, 2, or 3)

get icon graphic

get info\_id

end if

Gambar 3.19 *Pseudocode* Menarik Informasi Properti *Clickable Icon*

#### Penarikan Properti Konten Informasi pada Pop-up

get text\_id and image\_id with info\_id

get info\_title

get text’s properties

get image’s properties

get keywords

Gambar 3.20 *Pseudocode* Penarikan Properti Konten Informasi *Pop-up*

Saat sebuah *clickable* *icon* diklik, sistem akan menggunakan info\_id yang dibawa *icon* untuk mencari properti set informasi dari tabel info pada *database*. Properti set informasi terdiri dari judul, kode teks, kode gambar, dan kata kunci. Sistem kemudian mencari properti untuk masing-masing objek informasi teks dan gambar, juga kata kunci. Algoritma sistem dapat dilihat pada Gambar 3.20.

#### Pencarian Informasi Terkait

Pencarian informasi terkait dilakukan dengan membandingkan kata kunci pada satu set informasi, dengan kata kunci dari set-set informasi lainnya. Jika ditemukan minimal satu kata kunci yang sama, set informasi dengan kata kunci yang mirip tersebut menjadi informasi terkait. Keterkaitan informasi dikelompokkan dalam tingkatan tinggi, sedang, dan rendah untuk mengetahui mana informasi yang paling terkait. Gambar 3.21 menunjukkan algoritma pencarian informasi terkait.

compare keywords with another keywords in table info

if keywords similarity found

count keywords similarity percentage

group similar information(high, medium, low)

end if

Gambar 3.21 *Pseudocode* untuk Mencari Informasi Terkait

# IMPLEMENTASI, PENGUJIAN, DAN ANALISIS PERFORMA VIDEO STREAMING SEMANTIK INTERAKTIF

Pada bab ini akan dibahas mengenai implementasi, pengujian, serta analisis dari sistem video interaktif berbasis Web (Invidance) yang telah dibuat. Pengujian meliputi pengujian terhadap fungsi bagian-bagian dari sistem yang telah dibuat, dan pengujian performa sistem.

Dalam implementasi serta pengujian sistem Invidance digunakan *hardware* dan *software* dengan spesifikasi sebagai berikut:

1. Hardware

Untuk pengujian sistem, digunakan sebuah *laptop* dengan spesifikasi:

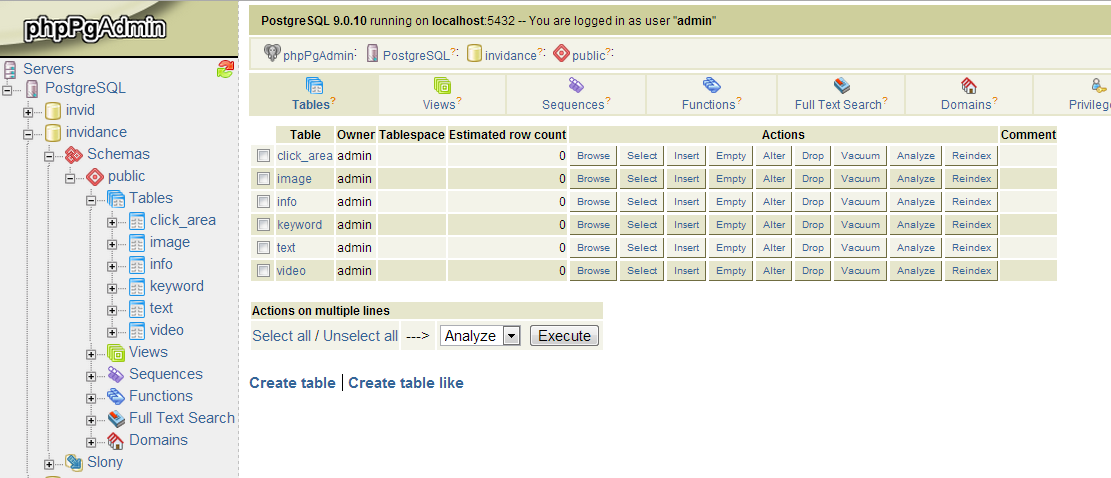
* *Processor* : Intel® Core™ i3 CPU M 370 @ 2.40GHz
* RAM : 2.00 GB
* Sistem Operasi: Windows 7 Home Premium 32-bit

1. Software

* XAMPP (*Basis Package*) version 1.7.3, dengan:
  + Apache 2.2.14
  + PHP 5.3.1
* *Database* PostgreSQL 9.0 dan *interface* phpPgAdmin 5.0.4
* *Browser* Google Chrome Version 27.0.1453.94 m

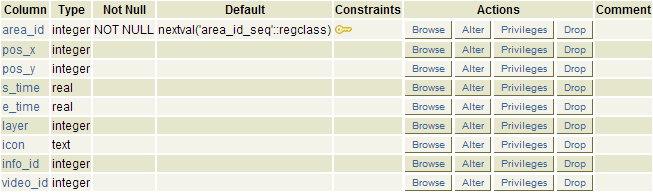
## Implementasi Sistem Database pada Invidance

*Database* Invidance dibuat menggunkan *database management system* PostgreSQL 9.0. *Database* terdiri dari enam tabel, yaitu click\_area, image, info, keyword, text, dan video. Gambar 4.1 menunjukkan *screenshot* daftar tabel pada *database* Invidance yang diambil dari halaman phpPgAdmin pada *browser*.

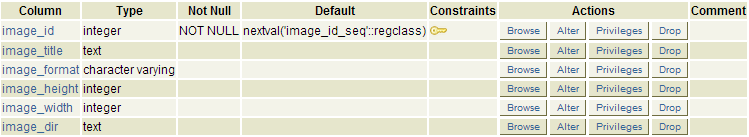


Gambar 4.1 Tabel-tabel pada *Database* Invidance

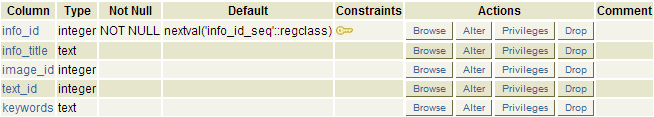
Gambar 4.2 hingga Gambar 4.7 menunjukkan atribut-atribut untuk setiap tabel pada *database*. Setiap tabel memiliki sebuah kolom yang mendefinisikan kode identitas untuk setiap baris pada tabel, seperti area\_id pada tabel click\_area atau image\_id pada tabel image. Masing-masing kode identitas ini bersifat *auto increment*, yaitu nilainya akan bertambah satu secara otomatis tiap kali memasukkan baris baru.



Gambar 4.2 Tabel click\_area



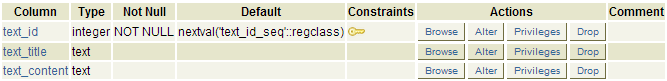
Gambar 4.3 Tabel image



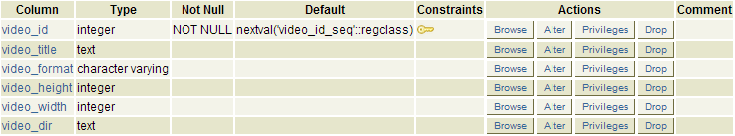
Gambar 4.4 Tabel info



Gambar 4.5 Tabel keyword



Gambar 4.6 Tabel text

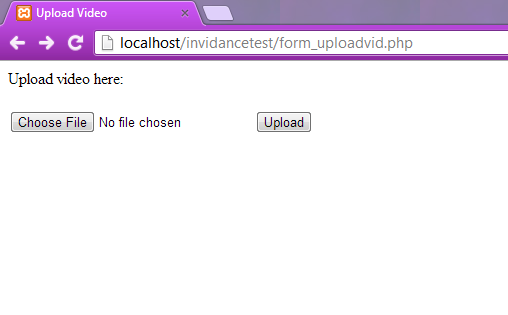


Gambar 4.7 Tabel video

### Halaman Admin

Sistem *admin* Invidance diterapkan untuk halaman *admin*. Halaman *admin* diawali dengan *form* untuk meng*upload* video yang ingin dibuat menjadi *clickable video*. Gambar 4.8 menunjukkan halaman unggah video pada Web Invidance yang terdapat pada localhost dengan URL: localhost/invidancetest/form\_uploadvid.php. Pada halaman ini, *user*, atau *admin*, dapat memilih video dari direktori komputer dengan mengklik tombol “Choose File”, kemudian mengklik tombol “Upload” untuk mengunggah video ke sistem Invidance.

Setelah *admin* mengklik tombol “Upload”, skrip upload\_video.php akan dijalankan. Skrip ini akan melakukan proses pengunggahan video, yaitu menyalin video ke sistem *file server* sekaligus mengisi poperti objek video pada tabel video di *database*.

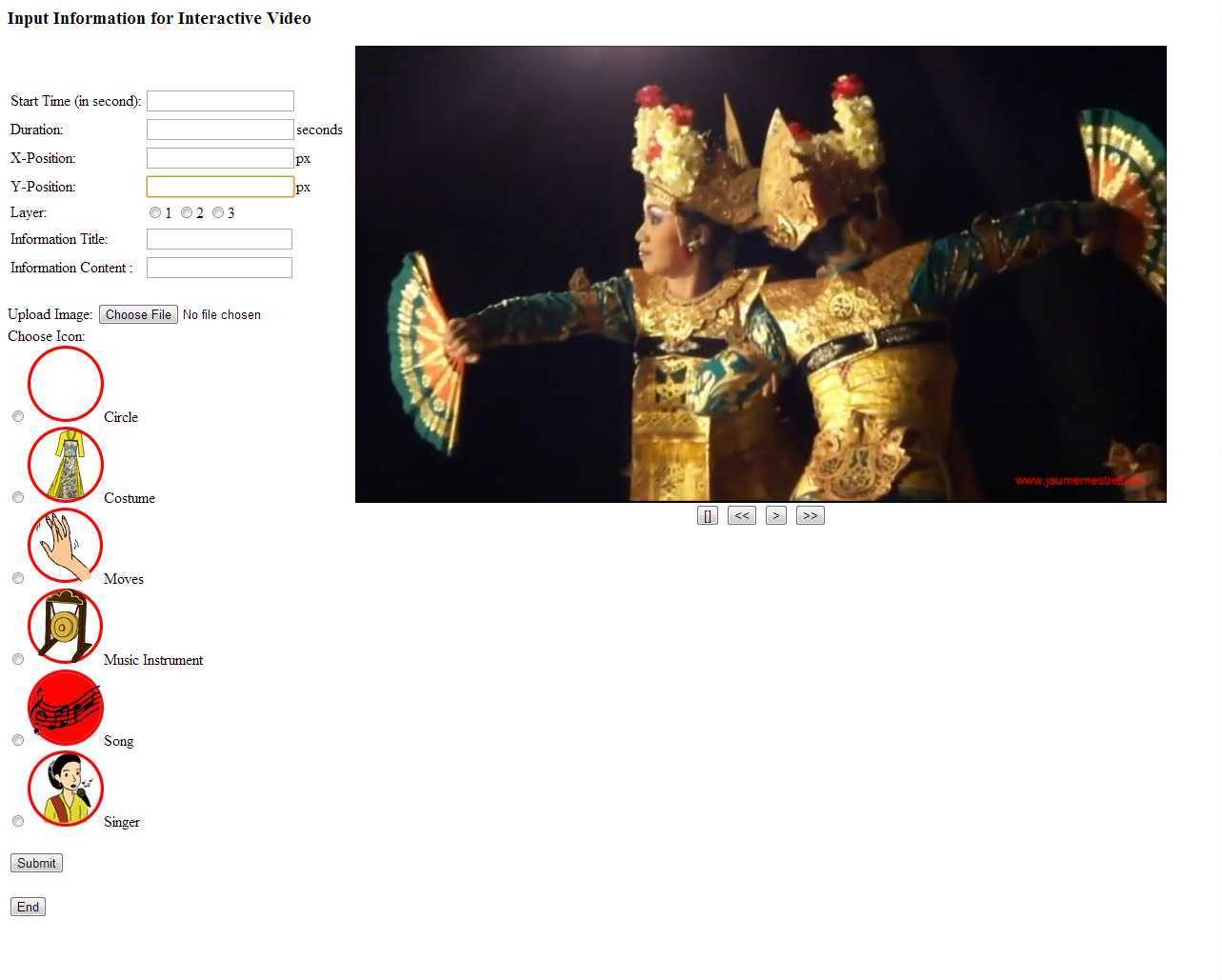


Gambar 4.8 Form Unggah Video

Ketika berkas video sudah selesai diunggah, *admin* akan memasuki halaman *form input* informasi, form\_data.php. Tampilan form input dapat dilihat pada Gambar 4.9. Dengan *form* ini, *admin* dapat memasukkan informasi untuk video interaktif serta memberikan *clickable icon* pada video. Untuk membuat *clickable icon*, *form* yang harus diisi adalah “Start Time“, “Duration”, “X-Coordinate”, “Y-Coordinate”, “Layer”, dan “Choose Icon”. Sementara untuk membuat informasi, yang nantinya akan muncul dalam *pop-up window*, yang harus diisi adalah “Information Title”, “Information Content”, dan “Upload Image”.

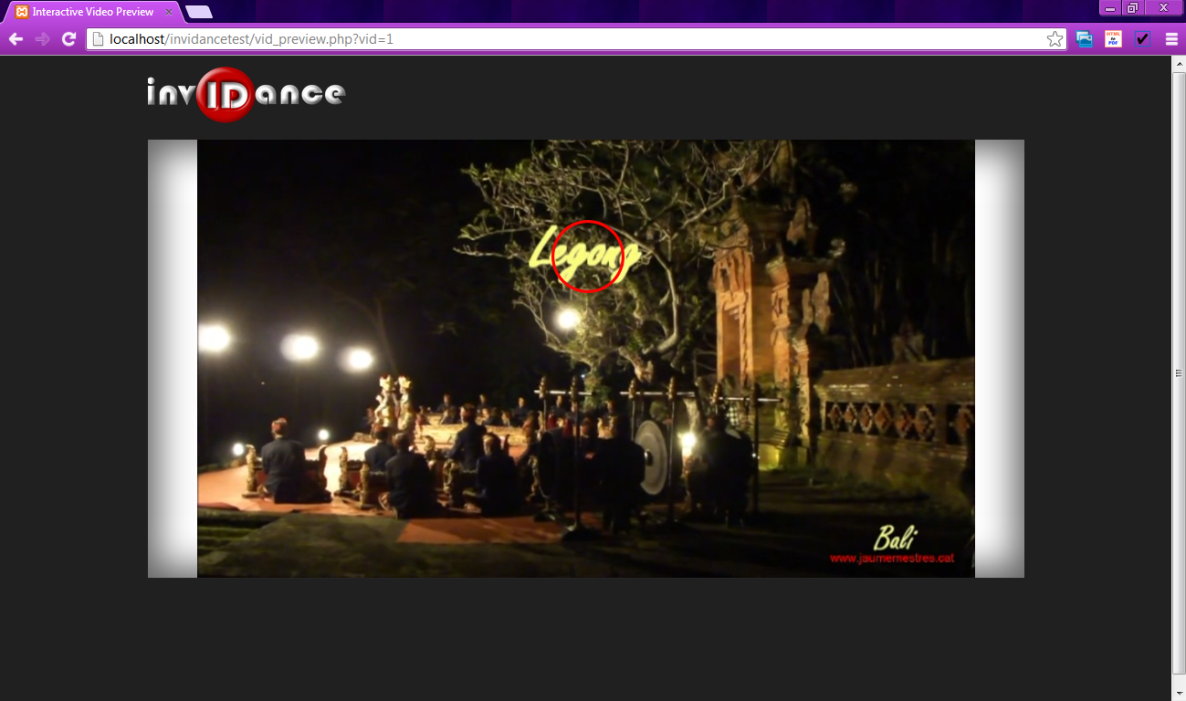
Untuk membuat informasi, *admin* dapat memilih untuk memasukkan gambar atau tidak. Selain itu, jika informasi tersebut tidak ingin diwakilkan oleh *clickable icon* pada video, *admin* dapat meninggalkan form-form untuk membuat *clickable icon* tetap kosong.

Setelah mengisi *form input* informasi, *admin* harus mengklik “Submit” untuk meminta sistem memproses informasi yang dimasukkan. Pada proses ini, sistem akan menjalankan skrip input\_data.php. Skrip ini akan memproses peletakkan properti isi informasi pada tabel text, properti gambar pada tabel image, pembuatan set informasi baru, dan juga pembuatan *clickable icon* baru. Setelah proses selesai, *admin* akan dikembalikan ke halaman form\_data.php untuk mengisi informasi lainnya.



Gambar 4.9 Tampilan Form Input Informasi

Jika *admin* telah selesai memasukkan semua informasi, maka dapat mengklik “End”. Sistem kemudian akan membawa *admin* ke halaman “Interactive Video Preview”, vid\_preview.php, seperti terlihat pada Gambar 4.10.



Gambar 4.10 Halaman Preview Video Interaktif

### Halaman Preview

Sistem *viewer* Invidance diterapkan untuk halaman *preview*. Halaman *preview* dapat diakses setelah *admin* selesai membuat *clickable video*. Pada *preview* akan ditampilkan video yang sebelumnya sudah diunggah, dan juga *clickable icon* sesuai dengan posisi dan waktu yang dimasukkan oleh *admin* pada form *input* informasi.

Pada penampilan video interaktif ini, proses pertama yang dilakukan adalah pengambilan properti video dari *database* berdasarkan kode identitas video yang didapat ketika mengunggah video. Proses pengambilan properti dilakukan oleh skrip fetchvideo.php, kemudian penampilan video dalam format HTML dilakukan oleh skrip loadvideo.php.

Ketika video mulai dimainkan, sebuah Javascript akan berjalan, yaitu clickarea.js. Skrip clickarea.js bertugas men-*trigger* sistem untuk melakukan pencarian ke dalam *database* selama video berjalan. Yang dicari sistem di dalam *database* ialah *clickable icon* yang harus muncul pada *current time* video. Pengecekkan dilakukan menggunakan pemanggilan skrip clickarea.php. Jika pada *current time* terdapat *clickable icon* yang harus muncul, maka skrip clickarea.php akan mengambil properti *clickable icon* tersebut dari *database*, dan mengembalikannya pada skrip clickarea.js untuk ditampilkan pada video.

## Uji Fungsionalitas Sistem Invidance

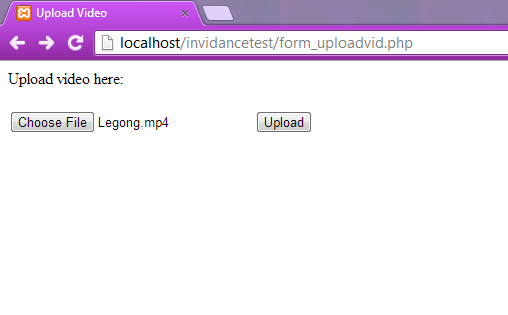
Sistem Invidance terdiri dari beberapa bagian subsistem dengan fungsi berbeda. Subsistem tersebut dapat dibagi ke dalam:

* Pangunggahan Video
* Pembuatan *Clickable Icon*
* Pembuatan Informasi
* Pencarian Kata Kunci
* Pencarian Informasi Terkait

Uji fungsionalitas dimaksudkan untuk mengetahui apakah sistem yang dibuat telah bekerja dengan benar dan hasilnya sesuai dengan tujuan.

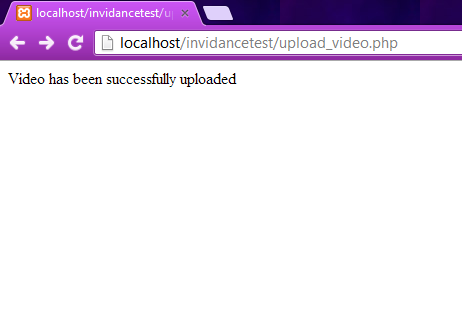
### Pengunggahan Video

Pada uji pengunggahan video, diambil contoh pengunggahan sebuah video berjudul “Legong.mp4”. Gambar 4.11 menunjukkan video yang sudah siap diunggah pada *form* unggah video. Selanjutnya, setelah sistem selesai melakukan proses unggah, muncul notifikasi sekilas yang menyatakan bahwa pengunggahan video berhasil, seperti terlihat pada Gambar 4.12.

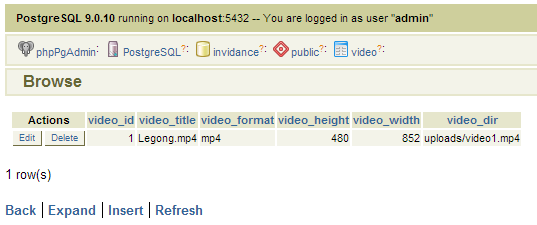


Gambar 4.11 Mengunggah Video

Gambar 4.13 menunjukkan tabel video pada *database*, dimana sudah terdapat data baru yang merupakan hasil dari pengunggahan video “Legong.mp4”. Berkas video juga ditemukan pada direktori yang tertera pada tabel video. Dengan demikian, terbukti bahwa fungsi pengunggahan video berjalan dengan baik.



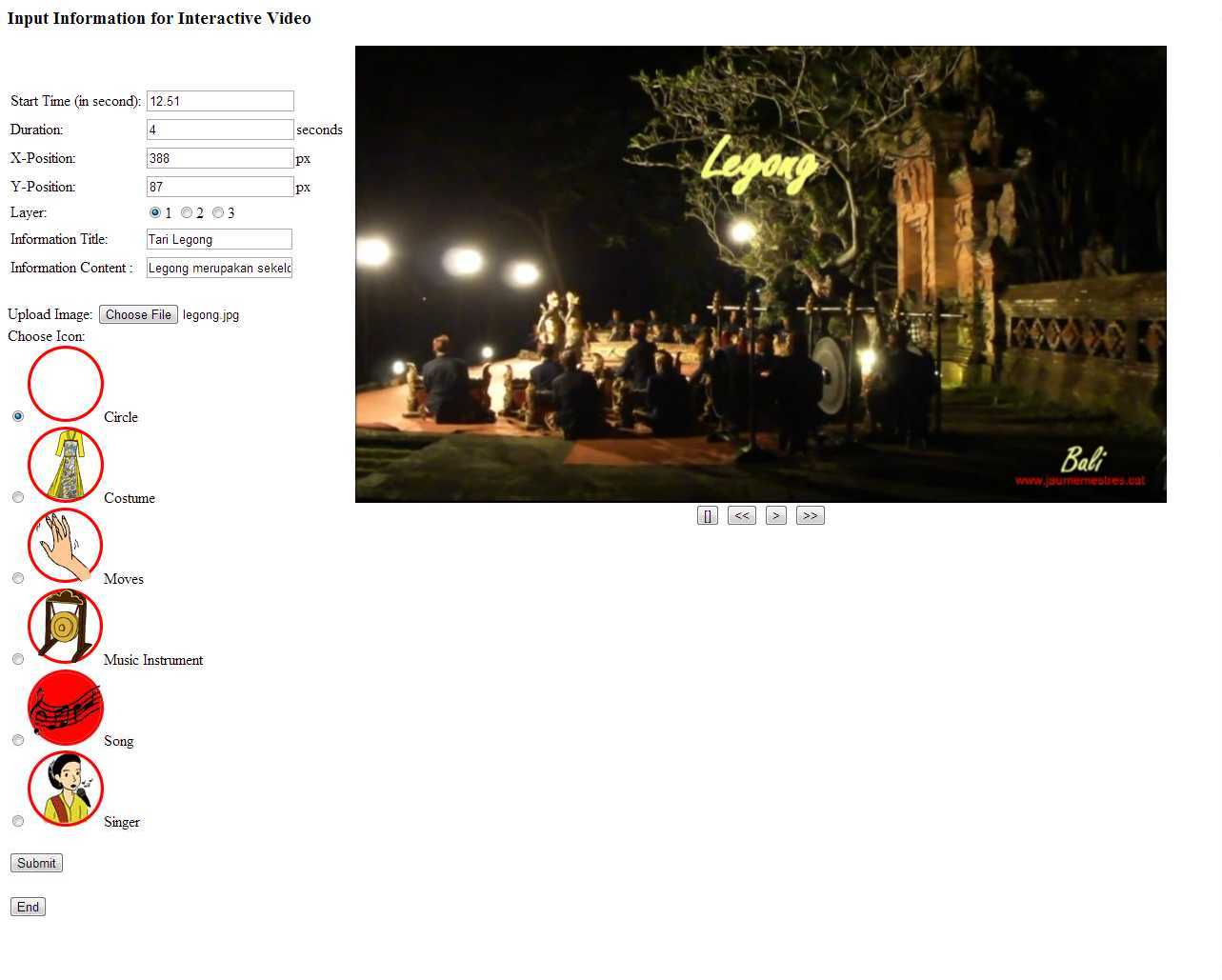
Gambar 4.12 Notifikasi Video Berhasil Diunggah



Gambar 4.13 Hasil Unggah Video pada Tabel video

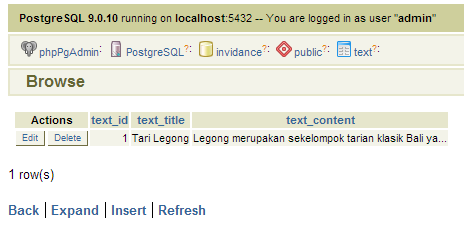
### Pembuatan Clickable Icon dan Informasi

Uji pembuatan *clickable icon* dan informasi dilakukan dengan mengisi *form* *input* informasi pada halaman form\_data.php. Gambar 4.14 memperlihatkan pengisian *form* untuk membuat *clickable icon*, membuat informasi, sekaligus mengunggah gambar untuk informasi.

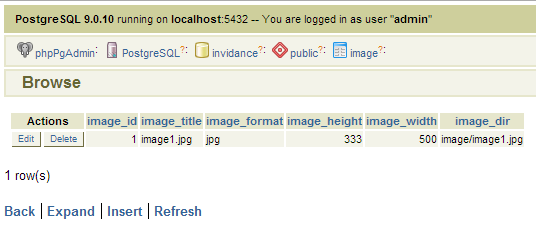


Gambar 4.14 Membuat *Clickable icon* dan Informasi

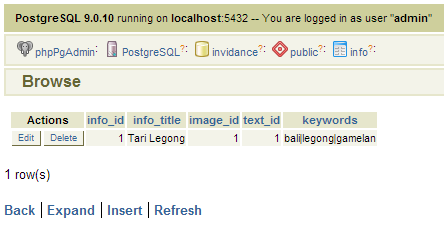
Fungsi pembuatan *clickable icon* dan informasi ini berjalan dengan baik, terbukti dengan masuknya data *input* dengan tepat ke tabel-tabel dalam *database*. Informasi teks masuk ke dalam tabel text seperti terlihat pada Gambar 4.15, informasi gambar masuk ke dalam tabel image seperti terlihat pada Gambar 4.16, set informasi pada tabel info terbentuk dengan benar seperti terlihat pada Gambar 4.17, dan *clickable icon* terbentuk dengan informasi yang benar pada tabel click\_area seperti terlihat pada Gambar 4.18.



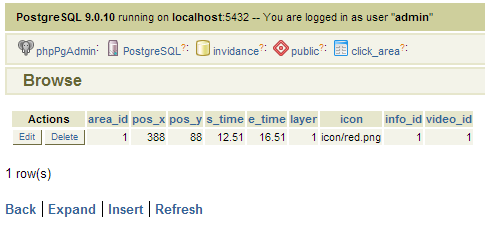
Gambar 4.15 Hasil Masukkan Informasi pada Tabel text



Gambar 4.16 Hasil Masukkan Informasi pada Tabel image



Gambar 4.17 Hasil Masukkan Informasi pada Tabel info



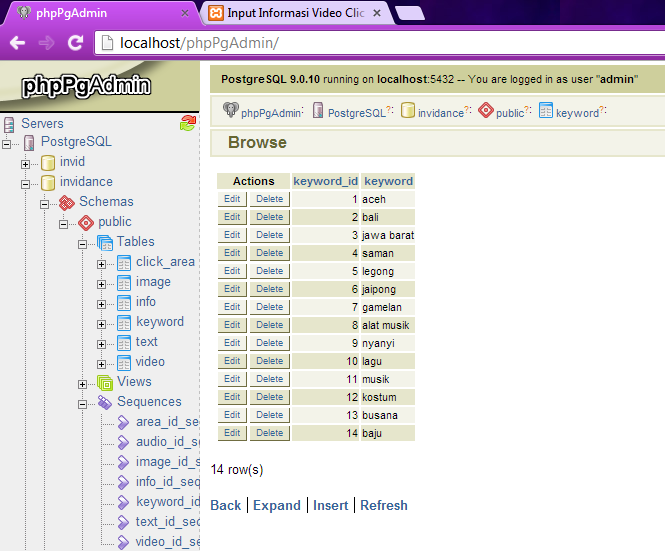
Gambar 4.18 Hasil Masukkan Informasi pada Tabel click\_area

### Pencarian Kata Kunci dan Pencarian Informasi Terkait

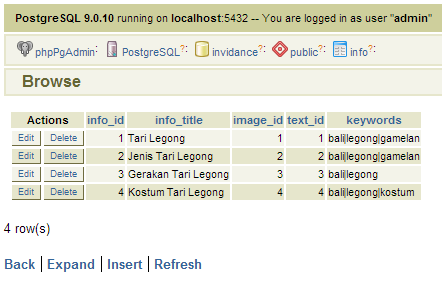
Pencarian kata kunci dilakukan dengan mencari kata kunci yang ada dalam daftar kata kunci pada isi informasi teks. Daftar kata kunci terdapat dalam *database* pada tabel keyword, seperti ditunjukkan oleh Gambar 4.19.

Pengujian dilakukan dengan empat informasi mengenai tari Legong, dan hasilnya ditunjukkan oleh Gambar 4.20. Kata kunci yang didapatkan sesuai dengan fungsi sistem pencari kata kunci. Kata kunci yang tertera tersebut merupakan kata-kata yang memang terdapat pada isi informasi teks.

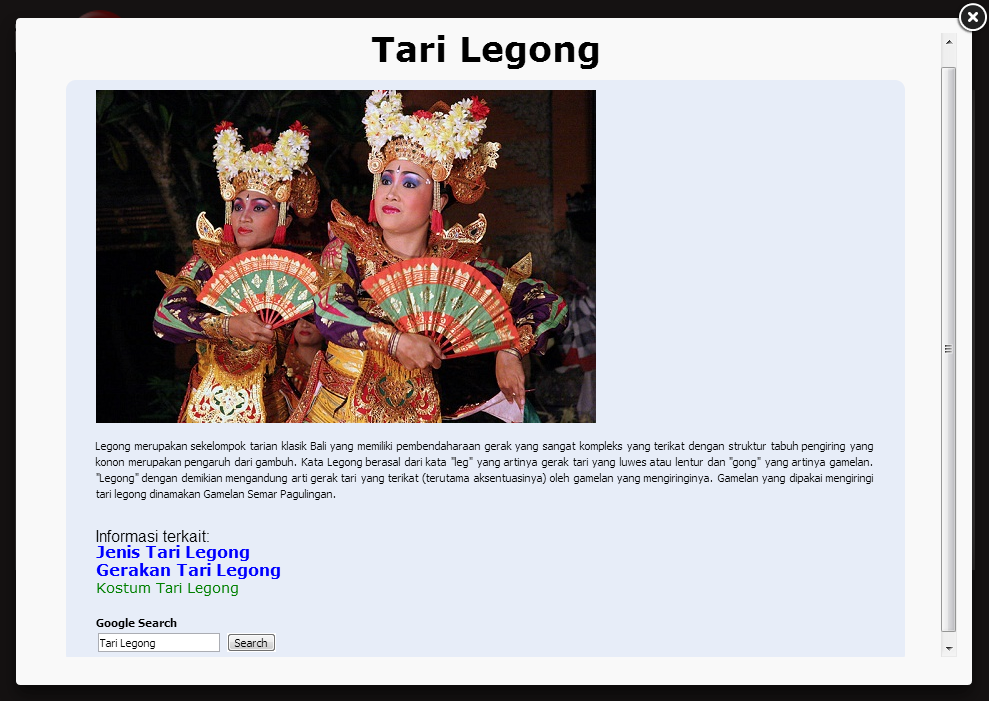
Selanjutnya, untuk uji coba pencarian informasi terkait, dilakukan dengan membuka salah satu informasi pada jendela *pop-up* dan melihat bagian “Informasi Terkait” yanga da di bagian bawah jendela *pop-up*, seperti pada Gambar 4.21. Ketika membuka informasi “Tari Legong”, terlihat bahwa informasi tersebut memiliki keterkaitan dengan informasi “Jenis Tari Legong”, “Gerakan Tari Legong”, dan “Kostum Tari Legong”.



Gambar 4.19 Tabel keyword



Gambar 4.20 Hasil Pencarian Kata Kunci pada Tabel info



Gambar 4.21 Penampilan Informasi terkait pada *Pop-up Window*

Format penulisan informasi terkait pada “Kostum Tari Legong” terlihat berbeda dari kedua informasi lainnya. Ini dikarenakan berdasarkan perhitungan presentase kemiripan kata kunci, informasi tersebut berada pada tingkat kemiripan sedang. Sementara dua informasi lainnya memiliki tingkat kemiripan yang tinggi.

Meskipun fungsi pencarian kata kunci serta informasi terkait ini sudah berjalan dengan baik, namun dirasa masih ada kekurangan pada fungsi. Kekurangan itu ialah pada kesesuaian kata kunci serta informasi terkait yang didapat dari suatu informasi. Fungsi pencarian kata kunci ini hanya berdasarkan pada kata apa saja yang terdapat dalam isi informasi, dan sangat sederhana untuk mempelajari apa sebenarnya isi informasi tersebut. Bila dalam informasi terdapat kata kunci yang sebenarnya sangat jauh keterhubungannya dengan informasi, sistem akan tetap mengambil kata tersebut sebagai kata kunci. Sistem belum dapat mengerti apakah di dalam teks informasi, kata kunci tersebut bernilai positif, dalam artian memang terkait langsung, ataukah kata kunci tersebut bernilai negatif, yaitu disebutkan dalam informasi namun sebenarnya tidak terkait dengan informasi.

Melihat kondisi tersebut, isi informasi yang dimasukkan untuk video interaktif sangat perlu diperhatikan dan diolah dengan baik, agar sistem tidak memberikan kesalahan kualitatif yang terlalu besar. Selain itu, daftar kata kunci juga harus lengkap dan dapat mewakili seluruh aspek dalam tema informasi.

## Uji Performa Sistem Invidance

Pengujian performa dilakukan untuk mengetahui rata-rata *response time* dari sistem saat menjalankan fungsi-fungsi subsistem. Response time merupakan salah satu parameter penting dalam performa aplikasi Web. *Response time* merupakan waktu yang dihabiskan sistem untuk merespon *request*, atau waktu yang dihabiskan untuk me-*load* skrip. Pada aplikasi Web, *response time* merupakan hal yang paling dirasakan oleh pengguna. Sistem yang optimal memiliki *response time* yang cepat, sehingga pengguna tidak perlu menunggu lama untuk melihat hasil dari kerja sistem.

Uji *response time* dilakukan pada skrip-skrip sistem Invidance yang terkait dengan *database*. Pengujian ini dimaksudkan untuk mengetahui waktu yang dibutuhkan untuk sebuah subsistem dalam melakukan transaksi *database* serta memproses data (*loading time*). Pengujian dilakukan dengan *localhost* menggunakan XAMPP.

Pada pengujian performa ini digunakan skrip PHP untuk menghitung *response time* dari masing-masing skrip subsistem Invidance. Skrip pengujian terdiri dari dua bagian, yaitu bagian memulai pengukuran waktu yang ditunjukkan pada Gambar 4.22, dan bagian akhir pengukuran waktu pada Gambar 4.23. Pengujian dilakukan sebanyak sepuluh kali untuk setiap subsistem dengan parameter-parameter tertentu.

<?php

$time = microtime();

$time = explode(' ', $time);

$time = $time[1] + $time[0];

$start = $time;

?>

Gambar 4.22 Skrip PHP untuk Memulai Pengukuran *Response Time* [14]

<?php

$time = microtime();

$time = explode(' ', $time);

$time = $time[1] + $time[0];

$finish = $time;

$total\_time = round(($finish - $start), 4);

echo 'Page generated in '.$total\_time.' seconds.';

?>

Gambar 4.23 Skrip PHP untuk Mengakhiri Pengukuran *Response Time* [14]

### Pengunggahan Properti Video

Pengujian performa pengunggahan properti video dilakukan pada skrip upload\_video.php, dimana pada skrip ini terdapat pemgambilan properti dari video yang diunggah, serta pemasukkan properti video ke dalam *database*. Pengujian dilakukan pada tiga video dengan properti berbeda seperti yang dispesifikasikan oleh Tabel 4.1.

Tabel 4.1 Spesifikasi Properti Video Pengujian

|  |  |  |  |
| --- | --- | --- | --- |
| Judul Video | Durasi | Bitrate | Ukuran |
| Video 1 | 1 menit 14 detik | 2449 kbps | 21.6 MB |
| Video 2 | 2 menit 25 detik | 2463 kbps | 45.5 MB |
| Video 3 | 3 menit 34 detik | 2288 kbps | 58.5 MB |

Gambar 4.24 menunjukkan grafik rata-rata *response time* untuk pengunggahan properti video. Data hasil pengujian dapat dilihat pada Lampiran 1. Rata-rata *response time* paling rendah dimiliki oleh Video 3 dengan nilai 3.89 ms, disusul oleh Video 2 dengan 4.28 ms, dan terakhir Video 1 dengan 4.29 ms. Selain mengunggah properti video ke *database*, pada skrip pengunggahan video juga terdapat penyalinan *temporary file* video ke dalam sistem *file* Invidance. Proses penyalinan ini berpengaruh pada waktu proses pengunggahan properti video. Jika dilihat pada grafik, berkas video dengan ukuran terbesar, yaitu Video 3 dengan ukuran 58.5 MB, justru memiliki waktu unggah yang paling kecil.

Waktu yang dihabiskan untuk menyalin *temporary file* video ini di luar kemampuan sistem Invidance. Hal yang mungkin mempengaruhi ialah terkait dengan sistem manajemen memori pada komputer, tepatnya pada saat mencari ruang memori untuk video. Selain itu, *bitrate* video juga mungkin berpengaruh. Video 3 memiliki *bitrate* paling kecil, sementara Video 1 dan Video 2 *bitrate*-nya tidak berbeda jauh.

Gambar 4.24 Grafik *Response Time* Pengunggahan Properti Video

Dalam konteks video, *bitrate* merupakan jumlah informasi video per detik. Semakin besar *bitrate*, maka kualitas video semakin bagus, dan juga semakin besar tempat penyimpanan yang dibutuhkan. Dengan demikian, untuk memperoleh waktu pengunggahan yang cepat dapat dilakukan dengan kompresi *bitrate* video, namun tetap harus memperhatikan aspek kualitas video.

Gambar 4.25 Grafik *Response Time* Pengunggahan Properti Video Tanpa Penyalinan *Temporary File*

Gambar 4.25 Menunjukkan grafik *response time* untuk mengunggahan properti video, namun tanpa pengaruh dari penyalinan *temporary file*. Pengujian terhadap ketiga video ini menghasilkan rata-rata *response time* yang relatif mirip, perbedaannya di bawah 1 ms.

### Pembuatan Clickable Icon dan Informasi

Pengujian performa pembuatan *clickable icon* dan informasi menghitung *response time* dari skrip input\_data.php saat mengambil data serta mengunggahnya ke dalam *database*. Pengujian dilakukan dengan tiga informasi berbeda dengan spesifikasi yang dijabarkan pada Tabel 4.2. Gambar 4.26 menunjukkan grafik hasil pengujian *response time* dari pengunggahan informasi. Data hasil pengujian dapat dilihat pada Lampiran 2.

Tabel 4.2 Spesifikasi Informasi Pengujian

|  |  |
| --- | --- |
| Judul Informasi | Jumlah Kata |
| Info 1 | 112 |
| Info 2 | 245 |
| Info 3 | 317 |

Gambar 4.26 Grafik *Response Time* Input Informasi ke *Database*

Pada pengujian *response time* untuk pengunggahan informasi ke *database* didapati bahwa rata-rata *response time* untuk Info 3 bernilai paling tinggi, yaitu 16.97 ms, dibandingkan Info 1 dengan 7.27 ms dan Info 2 dengan 14.36 ms. Hal ini sesuai karena jumlah kata pada Info 3 lebih banyak dibandingkan dua informasi lainnya, sehingga memang seharusnya membutuhkan waktu lebih lama untuk mengunggah informasi ke *database*.

Pada masing-masing pengujian *response time* pengunggahan properti informasi, dapat dilihat fluktuasi *response time* yang signifikan. Namun, nilai paling tinggi masih berkisar di bawah 0.1 *second*, sehingga fluktuasi ini tidak terlalu terasa oleh pengguna. fluktuasi *response time* ini dipengaruhi oleh kerja prosesor pada komputer *server*, yang sekaligus komputer client karena menggunakan *localhost*. Pada komputer terdapat banyak *background process* yang bekerja secara dinamis, sehingga mempengaruhi fluktuasi *response time*.

### Pencarian Kata Kunci

Pengujian performa pencarian kata kunci menghitung *response time* dari skrip input\_data.php khususnya pada bagian pencarian kata kunci dari isi informasi teks. Pada pengujian ini digunakan tiga informasi dengan panjang berbeda, yaitu Info 1 memiliki 112 kata, Info 2 memiliki 245 kata, dan Info 3 memiliki 317 kata. Gambar 4.27 menunjukkan grafik rata-rata *response time*. Data hasil pengujian dapat dilihat pada Lampiran 3.

Gambar 4.27 Grafik *Response Time* Pencarian Kata Kunci

Hasil pengujian menunjukkan rata-rata *response time* untuk Info 1 sebesar 2.83 ms, Info 2 sebesar 3.54 ms, dan Info 3 sebesar 3.73 ms. Pada pencarian kata kunci, jumlah kata atau karakter pada informasi sangat berpengaruh pada seberapa cepat proses pencarian kata kunci dapat dilakukan. Setiap kata kunci yang disimpan dalam sistem dicocokkan satu per satu dengan kata-kata di dalam informasi. Tepatnya, sistem mencari urutan karakter dalam informasi yang sesuai dengan urutan karakter pada setiap kata kunci dalam *database*. Secara logika, Info 3 akan memberikan *response time* yang lebih lama. Hal ini sesuai dengan yang didapatkan dari pengujian terhadap tiga jenis informasi.

### Penarikan Properti Media dari Database pada Halaman Preview

Pengujian performa dilakukan pada tiga skrip, yaitu fetchvideo2.php, clickarea.php dan popup\_info2.php untuk mengetahui *response time* dari masing-masing skrip. Skrip fetchvideo2.php berfungsi untuk menarik properti video dari *database*, yaitu properti video\_format, video\_height, video\_width, dan video\_dir. Setelah properti video didapatkan, kemudian disisipkan dalam skrip HTML untuk menampilkan video. Skrip clickarea2.php berfungsi untuk menarik properti *clickable area* dari *database*, yaitu pos\_x, pos\_y, icon, dan info\_id. Properti *clickable area* ini kemudian akan digunakan untuk menampilkan *clickable icon* pada video.

Skrip popup\_info2.php berfungsi untuk menarik properti informasi yang akan ditampilkan dalam *pop-up window* berdasarkan info\_id. Properti yang pertama ditarik adalah properti dari tabel info, yaitu info\_title, image\_id, dan text\_id. Properti image\_id kemudian digunakan untuk mengambil properti gambar dari tabel image, yaitu image\_title dan image\_dir, dan lalu digunakan untuk menampilkan gambar informasi pada *pop-up window*. Properti text\_id digunakan untuk mengambil properti teks dari tabel text, yaitu text\_content, kemudian ditampilkan pada *pop-up window*. Terakhir, skrip popup\_info2.php menarik informasi kata kunci untuk set informasi, kemudian mencocokkan kata kunci dengan kata kunci dari set informasi lain dalam tabel info. Set informasi yang memiliki kemiripan diberikan tautan dalam *pop-up window*.

Tabel 4.3 Response Time Penarikan Properti Media

|  |  |
| --- | --- |
| Jenis Media | Response Time (ms) |
| Video | 2.72 |
| *Clickable Area* | 2.68 |
| Informasi *Pop-up* | 3.42 |

Tabel 4.3 menunjukkan rata-rata *response time* untuk penarikan properti video, *clickable area,* dan informasi *pop-up* dari *database*. Data hasil pengujian dapat dilihat pada Lampiran 4. Rata-rata *response time* untuk video yaitu 2.72 ms, untuk *clickable area* 2.68 ms, dan untuk informasi yaitu 3.42 ms. Penarikan properti video dan *clickable area* hampir sama dan lebih cepat daripada informasi dikarenakan properti video dan *clickable area* kontennya lebih singkat daripada informasi. Pada informasi terdapat properti teks informasi yang isinya relatif panjang. Selain itu, skrip popup\_info2.php juga lebih rumit karena di dalamnya terdapat pencocokan kata kunci untuk informasi terkait.

Pada artikel “Response Times: The 3 Important Limits” yang ditulis oleh Jakob Nielsen dalam Website Nielsen Norman [15], dikatakan bahwa terdapat tiga batasan waktu berdasarkan kemampuan persepsi manusia yang harus diperhatikan untuk membuat aplikasi Web. Batasan pertama yaitu 0.1 detik dimana sistem bekerja dengan cepat dan pengguna merasakan proses yang instan. Batasan kedua yaitu 1 detik, dimana pengguna merasakan adanya delay, namun cenderung mengabaikan. Batasan ketiga yaitu 10 detik, dimana pengguna bisa saja mengalihkan fokusnya.

Ketiga batasan waktu ini dapat digunakan sebagai parameter kualitas performa penarikan properti media. Dari rata-rata *response time* untuk penarikan properti video, *clickable area* dan informasi yang diperoleh dapat diketahui bahwa proses yang dilakukan sistem termasuk instan. Namun, dalam Web Invidance ada banyak faktor lain yang mempengaruhi penyajian video dan informasi kepada pengguna, misalnya skrip untuk antarmuka, juga kualitas jaringan bila aplikasi Web Invidance diterapkan menggunakan jaringan. Dalam pembuatan Web, faktor antarmuka juga penting untuk diperhatikan, bukan hanya dari sisi estetika, tetapi juga jangan sampai memperlambat sistem yang sudah bekerja dengan cepat.

# KESIMPULAN

Kesimpulan yang dapat diambil dari implementasi, pengujian, serta analisis sistem Invidance, yaitu:

1. Sistem *database* Invidance dibagi menjadi dua, yaitu sistem *admin* dan sistem *viewer*. Uji fungsionalitas pada sistem *admin* dan *preview* menunjukkan bahwa tiap fungsi subsistem Invidance sudah dapat bekerja.
2. Pada pengunggahan properti video ke sistem Invidance, video berukuran paling besar namun dengan *bitrate* paling kecil lebih cepat diproses.
3. Uji performa pembuatan dan pengunggahan properti *clickable icon* dan informasi menunjukkan bahwa semakin panjang karakter teks informasi maka semakin lama waktu yang dihabiskan.
4. Uji performa pencarian kata kunci menunjukkan semakin panjang karakter teks informasi, maka semakin lama waktu yang dihabiskan. Fungsi pencarian belum terlalu akurat dalam menentukan kata kunci dikarenakan algoritma sistem yang sederhana dan belum dapat memahami makna dari tiap frasa dalam informasi.
5. Uji performa penarikan properti media pada halaman *preview* memberikan rata-rata *response time* untuk penarikan properti video sebesar 2.73 ms, penarikan properti *clickable area* sebesar 2.68 ms, dan penarikan properti informasi sebesar 3.42 ms. Hasil *response time* yang didapat terbilang cepat dan *delay* dapat diabaikan oleh pengguna.

DAFTAR ACUAN

1. Zhang, Z., & Zhang, R. (2008). *Multimedia data mining: a systematic introduction to concepts and theory*. Chapman & Hall/CRC Data Mining and Knowledge Discovery Series.
2. Kalipsiz, Oya. (2000). Multimedia *database*s. *IEEE International Conference on Information Visualization*, halaman 111-115.
3. Elmasri, Ramez, & Navathe Shamkant B. (2011) *Fundamental of database system* (6th ed.). Pearson.
4. Korth, Henry F., & Silberschatz, Abraham. (1991) *Database system concepts* (2nd ed.). New York: McGraw-Hill.
5. Centre for Object Technology. (1998). *Database management systems: relational, object-relational, and object-oriented data models* (V1.1 Final Version). Center of IT-Research (CIT), Danish Ministry of Industry.
6. Yuliano, Triswansyah. (2007). Pengenalan PHP. *Komunitas e-Learning IlmuKomputer.com*. IlmuKomputer.com.
7. Obitko, Marek. (2007). Semantic web. *Ontologies and Semantic Web*. Diakses pada 8 September 2012. http://www.obitko.com/tutorials/ontologies-semantic-web/semantic-web.html
8. World Wide Web Consortium. (2004). RDF primer. *W3C Recommendation*. Diakses pada 8 September 2012. http://www.w3.org/TR/rdf-primer/
9. TechTerms.com. (2011). Interactive video. *Software Terms Definition*. Diakses pada 16 Desember 2012. http://www.techterms.com/definition/ interactive\_video
10. MySQL.com. *About MySQL*. Diakses pada 2 Januari 2012. http://www.mysql.com/about/
11. Conrad, Tim. (2004). *PostgreSQL vs. MySQL vs. commercial databases: it's all about what you need*. Diakses pada 2 Januari 2012 http://www.devx.com/dbzone/Article/20743
12. PostgreSQL.org. *About PostgreSQL*. Diakses pada 2 Januari 2012. <http://www.postgresql.org/about/>
13. Informit*.* *The Database Normalization Process*. Diakses pada 1 Juli 2013. http://www.informit.com/articles/article.aspx?p=30646
14. PHP Jabbers. *Measuring PHP page load time*. Diakses pada 27 Mei 2013. http://www.phpjabbers.com/measuring-php-page-load-time-php17.html
15. Nielson, Jakob. (1993). *Response Times: The 3 Important Limits*. Diakses pada 10 Januari 2013. http://www.nngroup.com/articles/response-times-3-important-limits/

DAFTAR PUSTAKA

Dix, Alan; Finlay, Janet; Abowd, Gregory D.; & Beale, Russel. (2004). *Human-computer interaction*. England: Pearson.

Elmasri, Ramez, & Navathe, Shamkant B. (2011) *Fundamental of database system* (6th ed.). Pearson.

Hassanzadeh, Oktie; Kemensietsidis, Anastasios, & Velegrakis, Yannis. (2012). Data management issues on the semantic web. IEEE 28th International Conference on Data Engineering, halaman 1024-1026.

Ines, Fayech, & Habib, Ounalli. (2012). An ontological approach for SQL query expansion. IEEE International Conference on Information Technology and e-Service 2012.

Kalipsiz, Oya. (2000). Multimedia databases. *IEEE International Conference on Information Visualization*, halaman 111-115.

Korth, Henry F., & Silberschatz, Abraham. (1991). *Database system concepts* (2nd ed.). New York: McGraw-Hill.

Mukherjee, Partha Kr.; Nasipuri, Mita; Basu, D.K.; & Kundu, M. (2004). Indexing and searching in multimedia *database* management system. Proceedings of the IEEE INDICON 2004. First India Annual Conference, halaman 143-146.

Zhang, Z., & Zhang, R. (2008). *Multimedia data mining: a systematic introduction to concepts and theory*. Chapman & Hall/CRC Data Mining and Knowledge Discovery Series.

Centre for Object Technology. (1998). *Database management systems: relational, object-relational, and object-oriented data models* (V1.1 Final Version). Center of IT-Research (CIT), Danish Ministry of Industry.

Conrad, Tim. (2004). *PostgreSQL vs. MySQL vs. commercial databases: it's all about what you need*. Diakses pada 2 Januari 2012 http://www.devx.com/dbzone/Article/20743

Nielson, Jakob. (1993). *Response Times: The 3 Important Limits*. Diakses pada 10 Januari 2013. http://www.nngroup.com/articles/response-times-3-important-limits/

Obitko, Marek. (2007). Semantic web. *Ontologies and Semantic Web*. Diakses pada 8 September 2012. http://www.obitko.com/tutorials/ ontologies-semantic-web/semantic-web.html

Travers, Chris. (2012). *O/R modelling interlude: PostgreSQL vs MySQL*. Diakses pada 2 Januari 2013. http://ledgersmbdev.blogspot.co.uk/ 2012/09/ or-modelling-interlude-postgresql-vs.html

Yuliano, Triswansyah. (2007). Pengenalan PHP. *Komunitas e-Learning IlmuKomputer.com*. IlmuKomputer.com.

MySQL.com. *About MySQL*. Diakses pada 2 Januari 2012. http://www.mysql.com/about/

PostgreSQL.org. *About PostgreSQL*. Diakses pada 2 Januari 2012. <http://www.postgresql.org/about/>

TechTerms.com. (2011). Interactive video. *Software Terms Definition*. Diakses pada 16 Desember 2012. http://www.techterms.com/definition/ interactive\_video

WikiVS.com. (2013). *MySQL vs PostgreSQL*. Diakses pada 2 Januari 2013. http://www.wikivs.com/wiki/MySQL\_vs\_PostgreSQL

World Wide Web Consortium. (2004). RDF primer. *W3C Recommendation*. Diakses pada 8 September 2012. http://www.w3.org/TR/rdf-primer/

# LAMPIRAN 1: Data Hasil Uji Pengunggahan Properti Video

Tabel 1 Response Time Pengunggahan Properti Video ke Database

|  |  |  |  |
| --- | --- | --- | --- |
| Pengujian Ke- | Video 1 | Video 2 | Video 3 |
| 1 | 4.1 ms | 4.4 ms | 3.9 ms |
| 2 | 4.5 ms | 4.2 ms | 4.3 ms |
| 3 | 4.2 ms | 4.3 ms | 3.7 ms |
| 4 | 4.3 ms | 4.3 ms | 3.6 ms |
| 5 | 4.4 ms | 3.8 ms | 4.2 ms |
| 6 | 4.8 ms | 4.4 ms | 3.7 ms |
| 7 | 4.4 ms | 4.7 ms | 4.2 ms |
| 8 | 4.3 ms | 3.9 ms | 4.4 ms |
| 9 | 3.8 ms | 4.2 ms | 3.7 ms |
| 10 | 4.1 ms | 4.6 ms | 3.2 ms |
| Rata-rata | 4.29 ms | 4.28 ms | 3.89 ms |
| Rata-rata Keseluruhan | 4.15 ms | | |

Tabel 2 Response Time Pengunggahan Properti Video ke Database Tanpa Penyalinan Temporary File

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Pengujian Ke- | Video 1 | | Video 2 | Video 3 |
| 1 | 2.6 ms | | 2.6 ms | 2.7 ms |
| 2 | 3 ms | | 2.5 ms | 2.7 ms |
| 3 | 2.6 ms | | 3 ms | 2.6 ms |
| 4 | 2.6 ms | | 2.9 ms | 2.5 ms |
| 5 | 3.1 ms | | 2.1 ms | 2.7 ms |
| 6 | 3.1 ms | | 3.2 ms | 2.3 ms |
| 7 | 2.8 ms | | 3 ms | 2.9 ms |
| 8 | 2.7 ms | | 2.3 ms | 2.9 ms |
| 9 | 2.1 ms | | 3 ms | 2.3 ms |
| 10 | 2.4 ms | | 2.9 ms | 1.7 ms |
| Rata-rata | 2.7 ms | | 2.75 ms | * 1. ms |
| Rata-rata Keseluruhan | | 2.66 ms | | | |

LAMPIRAN 2: Data Hasil Uji Pembuatan Clickable Icon dan Informasi

Tabel 3 Response Time Pembuatan Clickable Icon dan Informasi

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Pengujian ke- | Info 1 | | Info 2 | Info 3 |
| 1 | 6.6 ms | | 7.9 ms | 10.3 ms |
| 2 | 8.8 ms | | 8.3 ms | 9.3 ms |
| 3 | 5.9 ms | | 10.3 ms | 8.5 ms |
| 4 | 7.2 ms | | 32.8 ms | 45.2 ms |
| 5 | 8.6 ms | | 36 ms | 9.3 ms |
| 6 | 8 ms | | 9.6 ms | 8.2 ms |
| 7 | 7.2 ms | | 12.5 ms | 49.1 ms |
| 8 | 7.7 ms | | 8.5 ms | 8.7 ms |
| 9 | 6.3 ms | | 8.1 ms | 11.3 ms |
| 10 | 6.4 ms | | 9.6 ms | 9.8 ms |
| Rata-rata | 7.27 ms | | 14.36 ms | * 1. ms |
| Rata-rata Keseluruhan | | 12.87 ms | | | |

LAMPIRAN 3: Data Hasil Uji Pencarian Kata Kunci

Tabel 4 Response Time Pencarian Kata Kunci

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Pengujian ke- | Info 1 | | Info 2 | Info 3 |
| 1 | 2.7 ms | | 3.4 ms | 3.8 ms |
| 2 | 2.9 ms | | 3.1 ms | 3.8 ms |
| 3 | 2.2 ms | | 3.1 ms | 3.9 ms |
| 4 | 3.5 ms | | 4.7 ms | 3.5 ms |
| 5 | 3.2 ms | | 4.2 ms | 3.4 ms |
| 6 | 3.3 ms | | 3.1 ms | 3.6 ms |
| 7 | 2.2 ms | | 3 ms | 3.6 ms |
| 8 | 3.5 ms | | 3.2 ms | 4.7 ms |
| 9 | 2 ms | | 3.2 ms | 3.5 ms |
| 10 | 2.8 ms | | 4.4 ms | 3.5 ms |
| Rata-rata | 2.83 ms | | 3.54 ms | * 1. ms |
| Rata-rata Keseluruhan | | 3.37 ms | | | |

LAMPIRAN 4: Data Hasil Uji Penarikan Properti Media dari Database pada Halaman Preview

Tabel 5 Response Time Penarikan Properti Media

|  |  |  |  |
| --- | --- | --- | --- |
| Pengujian Ke- | Video | Informasi | Clickable Area |
| 1 | 2.5 | 3.1 | 3.2 |
| 2 | 2.2 | 3.2 | 3.6 |
| 3 | 2.3 | 3.1 | 2.4 |
| 4 | 2.3 | 3.5 | 2.4 |
| 5 | 2.6 | 3.1 | 2.9 |
| 6 | 2.6 | 4.4 | 2.3 |
| 7 | 3.8 | 3.1 | 2.2 |
| 8 | 3.4 | 3.1 | 2.9 |
| 9 | 3 | 3.9 | 2.9 |
| 10 | 2.5 | 3.7 | 2 |
| Rata-rata | 2.72 | 3.42 | 2.68 |
| Rata-rata Keseluruhan | | 2.94 ms | | |