

**IMPLEMENTASI ALGORITMA *BOYER-MOORE* DALAM
APLIKASI LF-PRO (*LOST AND FOUND PROPERTY*) DI
UNIVERSITAS BAKRIE**

TUGAS AKHIR



Ristanti Septa Ayu Anggraini

1122001015

**PROGRAM STUDI INFORMATIKA
FAKULTAS TEKNIK DAN ILMU KOMPUTER
UNIVERSITAS BAKRIE**

JAKARTA

2016

**IMPLEMENTASI ALGORITMA *BOYER-MOORE* DALAM
APLIKASI LF-PRO (*LOST AND FOUND PROPERTY*) DI
UNIVERSITAS BAKRIE**

TUGAS AKHIR

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



Ristanti Septa Ayu Anggraini

1122001015

**PROGRAM STUDI INFORMATIKA
FAKULTAS TEKNIK DAN ILMU KOMPUTER
UNIVERSITAS BAKRIE**

JAKARTA

2016

HALAMAN PERNYATAAN ORISINALITAS

Tugas akhir ini adalah hasil karya saya sendiri, dan semua sumber baik yang dikutip maupun di rujuk telah saya nyatakan dengan benar.

Nama : Ristanti Septa Ayu Anggraini

NIM : 1122001015

Tanda Tangan :

Tanggal : 25 Agustus 2016

HALAMAN PENGESAHAN

Tugas Akhir ini diajukan oleh:

Nama : Ristanti Septa Ayu Anggraini
NIM : 1122001015
Program Studi : Informatika
Fakultas : Teknik dan Ilmu Komputer
Judul Skripsi : Implementasi Algoritma Boyer-Moore
dalam Aplikasi LF-Pro “*Lost and Found Property*” di Universitas Bakrie

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

DEWAN PENGUJI

Pembimbing : Yusuf Lestanto, S.T, M.Sc. (.....)
Penguji : Dr. Siti Rohajawati, S.Kom, M.Kom (.....)
Penguji : Prof. Dr. Hoga Saragih, S.T, M.T. (.....)
Ditetapkan di : Jakarta
Tanggal :

UNGKAPAN TERIMA KASIH

Alhamdulillahillobbabil'alam, puji syukur kehadiran Allah SWT yang selalu memberikan ilmu serta melimpahkan nikmat, rahmat, dan karunia-Nya sehingga Tugas Akhir yang berjudul "Implementasi Algoritma Boyer-Moore dalam Aplikasi LF-Pro (*Lost and Found Property*) di Universitas Bakrie" dapat terselesaikan. Shalawat dan salam senantiasa Penulis haturkan kepada Rasulullah SAW, keluarga dan para sahabatnya yang telah membimbing umatnya ke masa yang terang benderang penuh dengan cahaya iman.

Penyusunan Tugas Akhir ini tidak terlepas dari berbagai hambatan dan kesulitan dari awal hingga akhir penyusunan. Begitu banyak pihak yang telah memberikan doa, masukan, bantuan, semangat dan nasihat selama penyusunan Tugas Akhir ini. Oleh karena itu, Penulis sampaikan juga terima kasih kepada

1. Prof. Dr. Hoga Saragih, S.T, M.T. selaku Kepala Program Studi Informatika yang senantiasa memberikan masukan dan motivasi kepada penulis.
2. Bapak Yusuf Lestanto, S.T., M.Sc. selaku dosen pembimbing berkat bimbingan, pengetahuan, arahan dan masukan akhirnya hambatan dan kesulitan dapat diatasi. Penulis menyampaikan terima kasih yang sebesar-besarnya kepada beliau atas waktu, tenaga dan pikiran yang telah diberikan untuk membantu proses penyusunan Tugas Akhir ini.
3. Dr. Siti Rohajawati, S.Kom, M.Kom. selaku dosen penguji yang memberikan saran dan perbaikan dalam penelitian ini.
4. Keluarga tercinta, M. Hairul Imam S.T dan Sinarwati selaku orang tua penulis. Firda Dwi Ayu Ningtyas, Ghazy Finza Adisyahputra, M. Khairul Zafran Raditya selaku saudara penulis yang senantiasa mendampingi dan mendoakan penulis dan selalu menjadi motivasi penulis untuk tidak berputus asa dan tetap semangat dalam penyusunan tugas akhir ini.
5. Fachrurrizal Miftahul Arief yang setia memberikan motivasi, support dan keyakinan untuk menyelesaikan tugas akhir ini tepat waktu serta senantiasa mendengarkan curahan hati penulis.
6. Informatika 2012: Damar Reja, Dewi Fatmawati, Dewi Fatmarani, Fima Hayati, Yonita Rahmasari, Lainatussifa, Airlangga Adie, Hanada Firmandri

dan Eidhil Gifto yang telah melewati 4 tahun suka dan duka selama masa studi di Universitas Bakrie.

7. Seluruh pihak Program Studi Informatika Universitas Bakrie yang telah memberikan pembelajaran yang begitu bermanfaat selama perkuliahan.

Semoga Allah SWT membalas kebaikan dan memberikan keberkahan kepada kita semua. Serta semoga Tugas Akhir ini memberi informasi yang berguna dan dapat bermanfaat bagi semua kalangan bidang pendidikan, khususnya bidang Informatika.

Jakarta, 25 Agustus 2016

Ristanti Septa Ayu Anggraini

HALAMAN PERNYATAAN PERSETUJUAN PUBLIKASI

Sebagai sivitas akademik Universitas Bakrie, saya yang bertanda tangan di bawah ini:

Nama : Ristanti Septa Ayu Anggraini
NIM : 1122001015
Program Studi : Informatika
Fakultas : Teknik dan Ilmu Komputer
Jenis Tugas Akhir : Implementasi

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

Implementasi Algoritma Boyer-Moore dalam Aplikasi LF-Pro (*Lost and Found Property*) di Universitas Bakrie

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

Demikian pernyataan ini saya buat dengan sebenarnya.

Dibuat di : Jakarta

Pada tanggal : 25 Agustus 2016

Yang menyatakan

Ristanti Septa Ayu Anggraini

IMPLEMENTASI ALGORITMA *BOYER-MOORE* DALAM APLIKASI LF-PRO (*LOST AND FOUND PROPERTY*) DI UNIVERSITAS BAKRIE

Ristanti Septa Ayu Anggraini

ABSTRAK

Penelitian ini bertujuan untuk mengimplementasikan Algoritma Boyer-Moore untuk properti barang hilang dan temuan di Universitas Bakrie. Saat ini, proses dokumentasi masih dilakukan secara manual. Hal ini menjadi sebuah masalah bagi pemilik barang yang tidak pernah mendapatkan konfirmasi barang dari pihak security. Aplikasi ini dibangun dengan menggunakan metode *Web Development Life Cycle* (WDLC), dan dirancang dengan PHP dan MySQL. Algoritma Boyer-Moore diterapkan untuk pencarian *string* dengan fitur *auto-complete*. Berdasarkan hasil dari Metode Perbandingan Eksponensial (MPE) untuk membandingkan jumlah iterasi pencarian algoritma Boyer-Moore dan Brute-Force, terlihat bahwa Algoritma Boyer-Moore lebih baik daripada Algoritma Brute-Force untuk penerapan pada fitur *auto-complete*.

Kata Kunci:

Algoritma Boyer-Moore, Pencarian *String*, Barang Hilang dan Temuan

**BOYER-MOORE STRING SEARCH ALGORITHM
IMPLEMENTATION OF LF-PRO (LOST AND FOUND PROPERTY)
APPLICATION IN BAKRIE UNIVERSITY**

Ristanti Septa Ayu Anggraini

ABSTRACT

This research is aimed to implement the Boyer-Moore Algorithm for lost and found property in Universitas Bakrie. Currently, the documenting process is still manual. It's becomes problematic due to the owner who lost the property never found and received again. The application was developed by Web Development Life Cycle (WDLC) method and it used PHP and MySQL tools. The Boyer-Moore Algorithm was applied for string searching with auto-complete feature. Based on Metode Perbandingan Eksponensial (MPE) method for comparing between Boyer-Moore and Brute Force algorithm searching, it found that Boyer-Moore better than Brute Force for applied in auto complete feature.

Keywords:

Boyer-Moore Algorithm, String Searching, Lost and Found Property

DAFTAR ISI

HALAMAN JUDUL	i
HALAMAN PERNYATAAN ORISINALITAS	ii
HALAMAN PENGESAHAN	iii
UNGKAPAN TERIMA KASIH	iv
HALAMAN PERNYATAAN PERSETUJUAN PUBLIKASI.....	vi
ABSTRAK	vii
ABSTRACT	viii
DAFTAR ISI.....	ix
DAFTAR GAMBAR.....	xii
DAFTAR TABEL	xiv
DAFTAR RUMUS	xv
DAFTAR LAMPIRAN	xvi
DAFTAR SINGKATAN.....	xvii
BAB I.....	1
1.1 Latar Belakang Masalah.....	1
1.2 Identifikasi Masalah.....	2
1.3 Rumusan Masalah.....	3
1.4 Batasan Masalah.....	3
1.5 Tujuan Penelitian	3
1.6 Manfaat Penelitian	3
1.7 Sistematika Penulisan	3
BAB II	5
2.1 Penelitian Terdahulu.....	5
2.2 Konsep Dasar <i>String Searching</i>.....	11
2.3 Algoritma <i>Boyer-Moore</i>	12
2.4.1 <i>Good-suffix shift rule</i>	14
2.4.2 <i>Bad-character rule</i>	14
2.4 Algoritma <i>Brute-Force</i>	15
2.5 Perbandingan Algoritma <i>Boyer-Moore</i> dan Algoritma <i>Brute-Force</i>	16
2.6 Model Siklus Pengembangan Perangkat Lunak	18

2.7	Bahasa Pemrograman	22
2.8	<i>Unified Modeling Language (UML)</i>	25
2.9	Pengujian.....	26
BAB III		28
3.1	Kerangka Penelitian	28
3.2	Metode Perancangan dan Pengembangan	30
3.1.1	Pengamatan dan Perencanaan	30
3.1.2	Analisa Kebutuhan Aplikasi	30
3.1.3	Perancangan dan Pembangunan	31
3.1.4	<i>Testing</i>	53
3.1.5	Implementasi	54
3.3	Jenis Penelitian	54
3.4	Objek Penelitian	54
3.5	Metode Pengumpulan Data	54
3.6	Implementasi Algoritma <i>Boyer-Moore</i>	56
BAB IV		60
4.1	Implementasi Sistem	60
4.2	Implementasi Perancangan Antarmuka	61
4.3	Implementasi Data	68
4.4	Implementasi Algoritma <i>Boyer-Moore</i> pada Fitur <i>Auto-complete</i> ...	68
4.5	Hasil Pencarian Berdasarkan Kata Kunci.....	71
4.6	Pengujian Algoritma	73
4.7.1	Menentukan <i>Pattern</i> pada Teks	74
4.7.2	Proses Pencarian Algoritma	75
4.7.3	Menentukan Bobot Kriteria.....	79
4.7.4	Pemberian Nilai Pada Setiap Kriteria	80
4.7.5	Menghitung Skor.....	80
4.7.6	Menentukan Prioritas Keputusan	82
BAB V		84
5.1	Simpulan.....	84
5.2	Saran.....	84
DAFTAR PUSTAKA		86

LAMPIRAN.....	89
----------------------	-----------

DAFTAR GAMBAR

Gambar 2. 1 <i>Good-suffix shift</i> , u terjadi lagi didahului karakter c berbeda dari a (Kristanto, Rachmat, & Santosa, 2013)	14
Gambar 2. 2 <i>Good-suffix shift</i> , hanya <i>suffix</i> dari u yang terjadi lagi di <i>pattern</i> x (Kristanto, Rachmat, & Santosa, 2013)	14
Gambar 2. 3 <i>Bad-character shift</i> , b terdapat di <i>pattern</i> x (Kristanto, Rachmat, & Santosa, 2013)	14
Gambar 2. 4 <i>Bad-character shift</i> , b tidak ada di <i>pattern</i> x (Kristanto, Rachmat, & Santosa, 2013)	15
Gambar 2. 5 Contoh cara kerja pencarian algoritma <i>Brute-Force</i> (Hidayani, Sari, & Suharman, 2012)	16
Gambar 2. 6 <i>Software Engineering Layers</i> (Pressman, 2010)	18
Gambar 2. 7 <i>Web Development Life Cycle Model</i> (WDLC) (Kamatchi, Iyer, & Singh, 2013)	20
Gambar 2. 8 <i>Usage Statistics of Web Technologies</i> (Nagila, 2013)	23
Gambar 3. 1 Kerangka Penelitian	29
Gambar 3. 2 <i>Use Case Diagram</i> Aplikasi LF-Pro	31
Gambar 3. 3 <i>Activity Diagram</i> Aplikasi LF-Pro	39
Gambar 3. 4 <i>Sequence Diagram</i> Halaman <i>Login</i>	41
Gambar 3. 5 <i>Sequence Diagram</i> Lihat Barang Temuan	42
Gambar 3. 6 <i>Sequence Diagram</i> Lihat dan Edit Barang Hilang	43
Gambar 3. 7 <i>Sequence Diagram</i> Tambah Barang Temuan.....	44
Gambar 3. 8 <i>Sequence Diagram</i> Tambah Barang Hilang.....	45
Gambar 3. 9 <i>Sequence Diagram</i> Search Barang Hilang.....	46
Gambar 3. 10 <i>Sequence Diagram</i> Search Barang Hilang.....	47
Gambar 3. 11 <i>Sequence Diagram</i> Tambah Konfirmasi Barang	48
Gambar 3. 12 <i>Sequence Diagram</i> Lihat Konfirmasi Barang	49
Gambar 3. 13 <i>Sequence Diagram</i> Logout	50
Gambar 3. 14 <i>Class Diagram</i> Aplikasi LF-Pro	51
Gambar 3. 15 <i>Data Model</i> LF-Pro	52

Gambar 3. 16 <i>Flowchart</i> Algoritma Boyer-Moore (Pratiwi, Syarif, & Wibowo, 2012)	57
Gambar 4. 1 Halaman <i>Login</i>	61
Gambar 4. 2 Halaman Awal Barang Temuan	62
Gambar 4. 3 Tampilan Menu Barang Hilang	63
Gambar 4. 4 Tambahkan Data Barang Temuan atau Hilang	64
Gambar 4. 5 Formulir Konfirmasi Barang	64
Gambar 4. 6 Tampilan Detail Barang Temuan	65
Gambar 4. 7 Tampilan Data Pemilik	66
Gambar 4. 8 Notifikasi Barang Telah Diambil	66
Gambar 4. 9 Pesan Konfirmasi Barang Hilang via <i>E-mail</i>	67
Gambar 4. 10 Konfirmasi Pengambilan Barang via <i>E-mail</i>	67
Gambar 4. 11 Program <i>JavaScript</i> untuk Fitur <i>Auto-complete</i>	69
Gambar 4. 12 Kode <i>autocomplete.php</i>	69
Gambar 4. 13 Fungsi <i>cariTemuan()</i> dalam <i>class.BoyerMoore.php</i>	70
Gambar 4. 14 Bentuk <i>String</i> yang Akan Diolah	70
Gambar 4. 15 Fungsi <i>makechartable()</i>	71
Gambar 4. 16 <i>Search</i> dengan Fitur <i>Auto-Complete</i> Kondisi Pertama	72
Gambar 4. 17 Hasil <i>Search</i> Pada Kondisi Pertama	72
Gambar 4. 18 <i>Search</i> dengan Fitur <i>Auto-Complete</i> Kondisi Kedua	73
Gambar 4. 19 Hasil dari Pencarian <i>String</i> Kondisi Kedua	73
Gambar 4. 20 Grafik Perhitungan Skor	82

DAFTAR TABEL

Tabel 2. 1 Perbandingan Jenis Algoritma <i>Boyer-Moore</i> (Sagita & Prasetyowati, 2013)	7
Tabel 2. 2 Rangkuman Penelitian Terdahulu	9
Tabel 2. 3 Contoh Algoritma <i>Boyer-Moore</i>	13
Tabel 2. 4 Contoh Algoritma <i>Boyer-Moore</i>	13
Tabel 2. 5 Perbandingan Algoritma <i>Boyer-Moore</i> dan <i>Brute Force</i> (Abdeen, 2011)	17
Tabel 2. 6 Perbandingan model pengembangan aplikasi (Mujumdar, Masiwal, & Chawan, 2012)	19
Tabel 2. 7 Tabel Perbandingan antara ASP.NET dan PHP (Chandran & Angepat, 2011)	23
Tabel 3. 1 <i>Use Case Scenario Login</i>	32
Tabel 3. 2 <i>Use Case Scenario</i> Mencari Data Barang Hilang	32
Tabel 3. 3 <i>Use Case Scenario</i> Mencari Data Barang Temuan	33
Tabel 3. 4 <i>Use Case Scenario</i> Melihat Data Barang Hilang	34
Tabel 3. 5 <i>Use Case Scenario</i> Melihat Data Barang Temuan	34
Tabel 3. 6 <i>Use Case Scenario</i> Mengedit Barang Hilang	35
Tabel 3. 7 <i>Use Case Scenario</i> Membuat Laporan Kehilangan	36
Tabel 3. 8 <i>Use Case Scenario</i> Membuat Konfirmasi Barang	36
Tabel 3. 9 <i>Use Case Scenario</i> Membuat Laporan Penemuan	37
Tabel 3. 10 Penentuan Kriteria (Januardi, 2013)	53
Tabel 4. 1 Penentuan <i>Pattern</i> dan Teks Setelah Jumlah Hurufnya Disamakan	74
Tabel 4. 2 Simulasi Cara Kerja Algoritma <i>Brute Force</i>	76
Tabel 4. 3 Simulasi Cara Kerja Algoritma <i>Boyer-Moore</i>	78
Tabel 4. 4 Pembobotan Kriteria	80
Tabel 4. 5 Pemberian Nilai Pada Setiap Kriteria	80
Tabel 4. 6 Simulasi Perhitungan Analisa Menggunakan Perhitungan Perbandingan Eksponensial	81
Tabel 4. 7 Prioritas Keputusan	82

DAFTAR RUMUS

Rumus 2. 1 Rumus Metode Perbandingan Eksponensial (Januardi, 2013)	26
--	----

DAFTAR LAMPIRAN

Lampiran 1 <i>Software Requirement Specification</i>	90
Lampiran 2 Elisitasi LF-Pro.....	103
Lampiran 3. Rencana Kegiatan Penelitian	104
Lampiran 4. Surat Keterangan Penelitian	105
Lampiran 5. Hasil Wawancara.....	106
Lampiran 6. Algoritma <i>Boyer-Moore</i>	107
Lampiran 7. Surat Pengujian Aplikasi	109

DAFTAR SINGKATAN

ASCII	<i>American Standard Code for Information Interchange</i>
CSS	<i>Cascading Style Sheet</i>
HTML	<i>Hyper Text Markup Language</i>
J2EE	<i>Java 2, Enterprise Edition</i>
JSON	<i>JavaScript Object Nation</i>
KMP	Knuth Morris Pratt
MPE	Metode Perbandingan Eksponensial
MVC	<i>Model View Controller</i>
PHP	<i>Hypertext Preprocessor</i>
RAD	<i>Rapid Application Development</i>
SDLC	<i>Software Development Life Cycle</i>
SI	Sistem Informasi
UML	<i>Unified Modeling Language</i>
WDLC	<i>Web Development Life Cycle</i>

BAB I PENDAHULUAN

1.1 Latar Belakang Masalah

Berdasarkan kemajuan teknologi di setiap instansi di Indonesia, penerapan sistem berbasis teknologi berfungsi untuk mempermudah setiap kegiatan yang dilakukan secara manual. Penerapan teknologi ini untuk mendukung penyampaian informasi yang dapat diakses dengan mudah, cepat, dan sesuai dengan kebutuhan. Selain itu, untuk membuat implementasi yang sesuai dengan kegiatan operasional, dan penyediaan informasi untuk mendukung ketersediaan pelayanan publik (Suprawoto, 2008).

Kehilangan barang-barang pribadi di Universitas Bakrie merupakan hal yang sudah sering terjadi. Sebagian besar mahasiswa yang merasa kehilangan atau menemukan barang di lingkungan Universitas Bakrie akan segera melapor ke petugas *security* terdekat. Hal ini yang membuat *staff security* merasa kesulitan dalam melakukan penyimpanan dan pencarian barang yang telah ditemukan atau barang yang dicari karena jangka waktu penemuan terlalu lama ataupun catatan yang sudah menumpuk. Dalam satu minggu, pihak *security* bisa menerima laporan kehilangan atau penemuan sebanyak 10 kali laporan, jumlah ini termasuk cukup banyak mengingat jumlah mahasiswa di Universitas Bakrie yang sangat banyak (Lampiran 5).

Dari permasalahan tersebut, perlu adanya sebuah sistem atau sarana khusus yang dapat menampung data temuan barang dan lokasi penyimpanan barang tersebut. Salah satunya dengan membangun aplikasi LF-Pro berbasis *web* yang dapat diakses oleh *staff security* di berbagai pos keamanan di Universitas Bakrie. Dengan adanya sistem informasi temuan barang ini akan memudahkan *staff security* untuk mengelola temuan barang dan laporan kehilangan secara efektif.

Dalam pembangunan aplikasi LF-Pro dibutuhkan suatu metode pencarian yang dapat memudahkan sistem untuk melakukan pencarian. Hadirnya mesin pencarian (*Search Engine*) di dalam sistem informasi memudahkan pengguna komputer dalam mencari berbagai informasi. Untuk memudahkan penggunaanya,

Search Engine menambahkan fitur pencari sugesti hasil terdekat pencarian yaitu menggunakan fitur *auto-complete* (Januardi, 2013). Dalam pencarian fitur *auto-complete*, diperlukan sebuah algoritma dalam pencarian *string*. Algoritma yang digunakan untuk pencarian *string* pada saat ini semakin berkembang. Tujuan dari pengembangan algoritma pencarian *string* adalah agar mendapatkan hasil yang akurat dalam pencariannya. Sampai saat ini algoritma pencarian *string* dibagi menjadi 3 kategori berdasarkan arah pencocokan *string* yaitu dari kiri ke kanan, kanan ke kiri, dan pencocokan yang dimulai dengan menentukan arah pencarian secara spesifik. Metode yang dianggap paling natural dalam prosesnya adalah metode pencarian dengan memulai pencocokan string dari arah kiri ke kanan, pencarian paling efisien dalam praktiknya adalah pencarian yang dilakukan dengan mencocokkan string mulai dari kanan ke kiri, dan pencocokan *string* dari arah yang telah ditentukan memiliki hasil yang paling baik secara teoritis (Kumara, 2009).

Dalam penerapannya, algoritma yang dianggap memiliki hasil yang paling baik dalam praktiknya adalah algoritma yang melakukan pergerakan pencocokan *string* dari arah kanan ke kiri. Salah satu contoh algoritma yang menerapkan pencarian *string* dari kanan ke kiri adalah Algoritma *Boyer-Moore*. Algoritma ini telah banyak dikenal dan dianggap paling efisien untuk pencarian *string*. Pencocokan *string* dari kanan ke kiri membuat informasi pencarian semakin unik, sehingga mempercepat proses pencocokan *string* tersebut (Sagita & Prasetyowati, 2013).

1.2 Identifikasi Masalah

Ditinjau dari latar belakang dan masalah-masalah di atas dapat diidentifikasi bahwa sistem barang hilang dan temuan di Universitas Bakrie masih dilakukan secara manual dan dibutuhkan sebuah aplikasi pencarian barang dengan penerapan fitur *auto-complete searching* menggunakan sebuah algoritma pencarian untuk mempermudah sistem informasi. Salah satu algoritma *searching* yang efektif dalam praktiknya adalah algoritma *Boyer-Moore*.

1.3 Rumusan Masalah

Berdasarkan latar belakang di atas, dirumuskan masalah dalam penelitian ini adalah bagaimana penerapan algoritma *Boyer-Moore* dalam pencarian *string* fitur *auto-complete* pada aplikasi LF-Pro berbasis *web*?

1.4 Batasan Masalah

Untuk membatasi pembahasan agar tidak keluar dari konteks topik penelitian, maka batasan dalam pembahasan masalah sebagai berikut:

1. Aplikasi LF-Pro dibangun sebagai sarana mempermudah bagian keamanan Universitas Bakrie
2. Aplikasi LF-Pro hanya mengelola pencatatan barang yang telah ditemukan dan mendata laporan kehilangan dari civitas akademika Universitas Bakrie.

1.5 Tujuan Penelitian

Tujuan dilakukannya penelitian ini adalah dapat menerapkan algoritma *Boyer-Moore* pada fitur *auto-complete* aplikasi LF-Pro sehingga mempercepat proses pencarian data.

1.6 Manfaat Penelitian

Manfaat yang akan diperoleh dengan adanya penelitian ini adalah

1. Memudahkan bagian *security* dalam melakukan pencatatan dan pencarian data barang yang telah ditemukan serta mengurangi kehilangan data inventaris
2. Menambah referensi bagi peneliti lain dalam penerapan algoritma *Boyer-Moore* dalam proses pencarian.

1.7 Sistematika Penulisan

Sistematika penulisan menguraikan secara singkat mengenai penelitian yang dilakukan. Adapun sistematika penulisan adalah sebagai berikut:

BAB I PENDAHULUAN

Bab ini menjelaskan mengenai latar belakang penelitian, identifikasi masalah, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, dan sistematika penulisan.

BAB II TINJAUAN PUSTAKA

Bab ini menjelaskan *review* dari penelitian sebelumnya. Teori tersebut antara lain *Web Development Life Cycle*, Algoritma *Boyer-Moore*, dan *Unified Modeling Language*.

BAB III METODE PENELITIAN

Bab ini menjelaskan metode penelitian dan alokasi waktu yang dibutuhkan dalam menyelesaikan penelitian ini.

BAB IV IMPLEMENTASI DAN PENGUJIAN

Bab ini menjelaskan mengenai hasil implementasi algoritma *Boyer-Moore* dan pengujian menggunakan Metode Perbandingan Eksponensial.

BAB V SIMPULAN DAN SARAN

Bab ini menjelaskan simpulan dan saran yang mencakup seluruh hasil dari penelitian yang telah dilakukan

BAB II

TINJAUAN PUSTAKA

2.1 Penelitian Terdahulu

Dasar dan acuan yang berupa teori atau temuan melalui hasil dari berbagai penelitian sebelumnya merupakan sebuah hal yang perlu dijadikan data pendukung. Salah satu data pendukung yang menurut peneliti perlu dijadikan bagian dari penelitian adalah melakukan berbagai perbandingan dari penelitian terdahulu yang dianggap relevan dengan permasalahan yang sedang dibahas dalam penelitian ini. Acuan dan perbandingan terkait dengan penelitian terdahulu adalah penerapan algoritma *Boyer-Moore* dan aplikasi LF-Pro. Oleh karena itu, peneliti melakukan langkah kajian terhadap beberapa hasil penelitian berupa tesis dan jurnal melalui internet.

Berikut ini adalah beberapa pemaparan singkat tentang perbandingan dari analisis yang akan dilakukan dengan menganalisis jurnal

1. Analisa Perbandingan Algoritma *Brute Force* dan *Boyer-Moore* dalam pencarian *Word Suggestion* Menggunakan Metode Perbandingan Eksponensial (2013).

Dalam penelitian ini dirancang sebuah aplikasi *Word Suggestion*, yang merupakan aplikasi pencarian sugesti hasil terdekat dengan menggunakan *string matching* dalam sebuah pencarian, dalam perancangannya aplikasi tersebut membandingkan dua metode algoritma pencarian. Membandingkan metode algoritma *Boyer-Moore* dan *Brute Force* dengan beberapa percobaan dan menghitung keefektifan kedua algoritma menggunakan metode Perbandingan Eksponensial. Hasil dari penelitian tersebut adalah dari beberapa pencarian kata yang diperoleh berdasarkan jumlah iterasi yang ada pada kedua algoritma menunjukkan bahwa Algoritma *Boyer-Moore* memiliki jumlah iterasi paling sedikit sehingga menunjukkan Algoritma *Boyer-Moore* merupakan algoritma tercepat dibandingkan dengan Algoritma *Brute Force* (Januardi, 2013).

2. Rancang Bangun Sistem Informasi Pencarian Benda Hilang '*Lost and Found*' Berbasis *Website* di Universitas Brawijaya (2012).

Penelitian tersebut dilakukan dikarenakan tingkat pencarian dan penemuan barang hilang yang cukup banyak dan belum adanya sarana yang menampung barang yang ditemukan. Pencarian barang dan penemuan barang sebelumnya dilakukan dengan menempelkan brosur di tembok yang merupakan cara tradisional dan merusak keindahan tembok UB. Sehingga, peneliti membuat sarana informasi pencarian benda hilang berbasis *web* yang terintegrasi dengan layanan BAIS (*Brawijaya Authentication and Identification System*) yang memungkinkan pengguna *single sign on* untuk aplikasi dalam domain UB. Hanya pengguna otentik saja yang dapat menggunakan layanan aplikasi *Lost and Found*. Peneliti tersebut juga menerapkan rancang bangun menggunakan *Framework Codeigniter* dengan penerapan pola desain *Model View Controller* (MVC). Dalam perancangan aplikasi tersebut peneliti menggunakan 4 diagram yaitu *use case diagram*, *activity diagram*, *sequence diagram*, dan *class diagram*. Hasil dari penelitian tersebut adalah aplikasi *lost and found* berhasil dibangun dan ditujukan untuk mahasiswa Universitas Brawijaya setelah dilakukan dua tahap pengujian yaitu pengujian validasi dan pengujian *feedback* dari *user* (Wibisono, Priharsari, & Muttaqin, 2012).

3. Studi Perbandingan Implementasi Algoritma *Boyer-Moore*, *Turbo Boyer-Moore*, dan *Tuned Boyer-Moore* dalam Pencarian *String* (2013).

Dalam penelitian tersebut, peneliti menganalisis dan membuat pencarian *string* dengan menggunakan tiga jenis Algoritma *Boyer-Moore* yaitu Algoritma *Boyer-Moore*, *Turbo Boyer-Moore*, dan *Tuned Boyer-Moore* dengan tujuan adalah untuk mengetahui bagaimana performa algoritma-algoritma tersebut, terutama di bidang waktu yang diperlukan untuk mencari suatu *pattern* dalam *text*. Peneliti membangun sebuah aplikasi menggunakan metode *prototyping* dan menggunakan Microsoft Visual Studio dengan bahasa C#. Aplikasi tersebut mendukung pencarian dengan menggunakan tiga algoritma, pengubah kata (*replace*), *highlight* kata yang dicari, dan pemberian informasi waktu yang dibutuhkan masing-masing algoritma untuk pencarian serta

algoritma mana yang membutuhkan waktu paling sedikit untuk pencarian. Dari penelitian yang dilakukan, dapat disimpulkan bahwa algoritma yang tercepat dalam pencarian *string* adalah algoritma *Boyer-Moore*. Berdasarkan Tabel 2.1 Algoritma Turbo *Boyer-Moore* merupakan algoritma tercepat kedua dan yang paling lambat adalah *Tuned Boyer-Moore* (Sagita & Prasetyowati, 2013).

Tabel 2. 1 Perbandingan Jenis Algoritma *Boyer-Moore* (Sagita & Prasetyowati, 2013)

Algoritma	Karakteristik
<i>Boyer-Moore</i>	Pencocokan karakter dari kanan ke kiri dan bukan dari kiri ke kanan sehingga akan lebih banyak informasi yang didapat
<i>Turbo Boyer-Moore</i>	Membutuhkan ruang lebih tetapi membutuhkan pemrosesan ekstra. Ruang ekstra yang diperlukan berguna untuk mengingat faktor dari teks yang cocok dengan akhiran dari <i>string</i> yang dicari selama <i>attempt</i> terakhir dan hanya jika <i>good-suffix</i> dilakukan
<i>Tuned Boyer-Moore</i>	Fitur utama dari algoritma ini adalah simplifikasi dari algoritma <i>Boyer-Moore</i> , mudah untuk diimplementasikan, hanya menggunakan <i>bad-character shift</i> , dan sangat cepat dalam praktiknya

4. Perancangan dan Pembuatan Sistem Informasi Kehilangan Berbasis Web (2014).

Penelitian tersebut bertujuan untuk membuat aplikasi berbasis *web* untuk sistem informasi kehilangan di Universitas Muhammadiyah Surakarta. Penelitian ini dilakukan mengingat mahasiswa merasa kesulitan dalam

menemukan barang yang telah hilang dan mahasiswa tidak dapat mengandalkan pihak satpam saja, sehingga dibutuhkan suatu sarana yang dapat diakses oleh semua pihak untuk menemukan barang dan dapat dijadikan arsip oleh pihak satpam setiap bulannya. Perancangan dilakukan menggunakan *tool software ApacheFriends XAMPP* (Barispaket) *version 1.6.7* (*MySQL 5.0.51 (Community Server)*, *PHP 5.2.6* dan *PHP 5.2.6* dan *phpMyAdmin 2.11.7*), dengan *web desainer Macromedia Dreamweaver 8*. Sistem kehilangan berbasis *web* tersebut sudah dibenahi dan dibuat dengan menggunakan bahasa pemrograman *PHP* dan *database MySQL*. Dari hasil pengujian dengan berbagai macam internet *browser* secara *localhost* maupun *online* dapat dilihat bahwa sistem dapat berjalan lancar (Supriyanto, 2014).

Tabel 2. 2 Rangkuman Penelitian Terdahulu

No	Judul	Pengarang	Tahun	Hasil	Perbedaan Dengan LF-Pro
1	Analisa Perbandingan Algoritma <i>Brute Force</i> dan <i>Boyer-Moore</i> dalam pencarian <i>Word Suggestion</i> Menggunakan Metode Perbandingan Eksponensial	Andri Januardi	2013	Dibuat sebuah aplikasi <i>word suggestion</i> pada mesin pencarian dengan menggunakan dua perbandingan algoritma yaitu <i>Boyer-Moore</i> dan <i>Brute Force</i> , dan menunjukkan hasil dari pencarian masing-masing algoritma.	LF-Pro hanya menerapkan algoritma Boyer-Moore pada fitur <i>auto-complete</i> dan hasil pengujian Metode Perbandingan Eksponensial dibatasi hanya pada maksimal karakter yang telah ditentukan di fitur <i>auto-complete</i> tersebut.
2	Rancang Bangun Sistem Informasi Pencarian Benda Hilang ' <i>Lost and Found</i> ' Berbasis <i>Website</i> di Universitas Brawijaya.	Dedi Arief Wibisono, Diah Priharsari, ST., MT, Adharul Muttaqin, ST., MT	2012	Dibuat sebuah aplikasi berbasis <i>web</i> yang diintegrasikan dengan menggunakan NIM & <i>password</i> yang dimiliki mahasiswa, maka mahasiswa dapat <i>login</i> di aplikasi dan dapat memberi informasi mengenai kehilangan/penemuan benda.	LF-Pro merupakan aplikasi pencarian barang hilang dan temuan yang hanya berfokus pada proses pencarian dan inventaris data, dan <i>user</i> pada LF-Pro hanya bagian <i>security</i> sebagai pengelola barang temuan dan barang hilang.

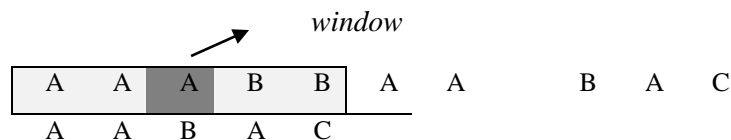
No	Judul	Pengarang	Tahun	Hasil	Perbedaan Dengan LF-Pro
3	Studi Perbandingan Implementasi Algoritma <i>Boyer-Moore</i> , <i>Turbo Boyer-Moore</i> , dan <i>Tuned Boyer-Moore</i> dalam Pencarian <i>String</i> .	Vina Sagita, Maria Irmina Prasetyowati	2013	Peneliti membandingkan dan membangun sebuah aplikasi yang dapat menghitung kecepatan sebuah pencarian data berdasarkan algoritma yang dipakai, hasilnya adalah algoritma <i>Boyer-Moore</i> merupakan algoritma tercepat jika dibandingkan dengan algoritma <i>Turbo Boyer-Moore</i> dan <i>Tuned Boyer-Moore</i> .	LF-Pro mengambil algoritma <i>boyer-moore</i> dengan menggunakan <i>bad-character rule shift</i> dan <i>good suffix shift rule</i> .
4	Perancangan dan Pembuatan Sistem Informasi Kehilangan Berbasis <i>Web</i>	Supriyanto	2014	Perancangan <i>website</i> kehilangan dengan menggunakan <i>software ApacheFriends XAMPP</i> (Barispaket) <i>version 1.6.7</i> (<i>MySQL 5.0.51 (Community Server)</i>), <i>PHP 5.2.6</i> dan <i>phpMyAdmin 2.11.7</i>), dengan <i>web desainer Macromedia Dreamweaver 8</i> .	Pembangunan aplikasi LF-Pro menggunakan OOP PHP dengan algoritma <i>boyer-moore</i> pada fitur pencarian.

2.2 Konsep Dasar *String Searching*

String merupakan urutan dari karakter, dimana karakter ini dapat terdiri dari beberapa alfabet. Misalnya adalah *string* biner yang terdiri dari dua alfabet, yaitu 0 dan 1, jadi *string* biner merupakan suatu urutan karakter 0 maupun 1. Contoh lain adalah *string American Standard Code for Information Interchange (ASCII)* yang terdiri dari 256 alfabet.

Pencarian *string* pada dasarnya adalah mencari *pattern* P yang memiliki panjang m dalam suatu teks T yang memiliki panjang n. Pola dan teks dalam pencarian ini dimasukkan ke dalam sebuah *array*, pola dinyatakan dengan $P[0 \dots m-1]$ dan teks dinyatakan dengan $T[0 \dots n-1]$.

Suatu teks dikatakan cocok adalah apabila *pattern* yang dimasukkan pada teks yang dicari adalah tepat sama, begitu pula sebaliknya.



- Abu-abu muda menunjukkan kecocokan
- Abu Abu tua menunjukkan ketidakcocokan

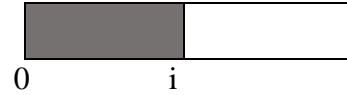
Dalam pencarian *string*, *window* merupakan sebuah kotak teks yang memiliki ukuran sama dengan panjang pola teks, fungsi *window* adalah membantu pencarian pola, *window* ditempatkan pada posisi paling kiri dari teks yang akan dicari. Setelah itu dilakukan sebuah percobaan dengan membandingkan karakter-karakter dalam *window* dengan karakter yang ada di pola. Dua sebab terjadinya pergeseran *window* adalah jika terjadi kecocokan dari seluruh karakter dalam pola atau pola tersebut ditemukan di dalam teks. Sebab pertama pada pergeseran ini adalah untuk mencari pola selanjutnya. Karena sebab kedua adalah jika terjadi ketidakcocokan. Mekanisme tersebut diulang sampai batas kanan *window* melebihi batas kanan dari teks.

Misalnya S adalah sebuah *string* dengan panjang m maka ada beberapa bagian dari *string* (Utomo, Harjo, & Handoko, 2011):

- *Substring* $S[i \dots j]$ adalah bagian dari *string* antara i dan j



- *Prefix* dari *S* adalah sebuah *substring* yaitu $S[0 \dots i]$



- *Suffix* dari *S* adalah sebuah *substring* yaitu $S[i \dots m-1]$



Dimana i dan j adalah suatu indeks *array* antara 0 dan $m-1$

Contoh:

Sebuah *string* *S*

A	n	D	r	A	L
0				5	

Panjang *string* = 6

Substring $S[1 \dots 3] = \text{"ndr"}$

Semua kemungkinan *prefix* dari *S*:

“andral”, “andra”, “andr”, “and”, “an”, “a”

Semua kemungkinan *suffix* dari *S*:

“andral”, “ndral”, “dral”, “ral”, “al”, “l”

2.3 Algoritma *Boyer-Moore*

Menurut Wibisono (Kumara, 2009), algoritma *Boyer-Moore* adalah salah satu algoritma untuk mencari *string* di dalam teks, algoritma ini ditemukan oleh R.M Boyer dan J.S Moore. Algoritma *Boyer-Moore* pencocokan *string* dari kanan ke kiri, akan tetapi pergeseran *window* tetap dimulai dari kiri ke kanan. Jika *pattern* dan teks cocok, maka dilakukan perbandingan karakter teks dan pola yang sebelumnya, yaitu dengan mengurangi indeks pola dan teks masing-masing sebanyak satu. (Argakusumah & Hansun, 2014).

Menurut Chiquita (Chiquita, 2012), dengan menggunakan algoritma *Boyer-Moore* ini, secara rata-rata proses pencarian akan menjadi lebih cepat jika dibandingkan dengan algoritma lainnya. Alasan dilakukannya pencocokan *string* dari kanan ke kiri dapat dilihat dari contoh berikut:

Tabel 2. 3 Contoh Algoritma *Boyer-Moore*

M	I	N	U	M		S	I	R	U	P
S	I	R	U	P						

Pada Tabel 2.3, dengan melakukan perbandingan dari posisi paling kanan *string* dapat dilihat bahwa karakter “m” pada *string* “minum” tidak cocok dengan karakter “p” pada *string* “sirup” yang dicari, dan karakter “m” tidak pernah ada dalam *string* “sirup” yang dicari sehingga *string* “sirup” dapat digeser melewati *string* “sirup”, sehingga posisinya seperti berikut.

Tabel 2. 4 Contoh Algoritma *Boyer-Moore*

M	I	N	U	M		S	I	R	U	P
					S	I	R	U	P	

Pada Tabel 2.4, terlihat bahwa loncatan algoritma cukup besar, hal ini ditandai dengan lompatan pencocokan *pattern* “sirup” yang melompat sebanyak lima karakter sekaligus dikarenakan *pattern* “p” pada “sirup” tidak sama dengan “m” (Argakusumah & Hansun, 2014).

Secara sistematis, algoritma *Boyer-Moore* melakukan langkah-langkah pencocokan *string* sebagai berikut:

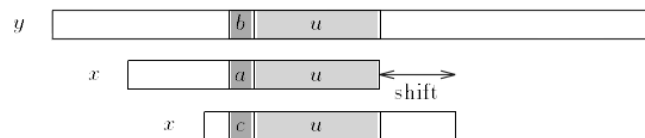
1. Algoritma *Boyer-Moore* mencocokkan *pattern* dari awal teks
2. Pencocokan dimulai dari kanan ke kiri, proses pencocokan dilakukan setiap karakter *pattern* dengan teks yang bersesuaian hingga salah satu kondisi terpenuhi, kondisi yang harus dipenuhi adalah:
 - a. *Pattern* yang dicari dan teks yang dibandingkan mengalami ketidakcocokan
 - b. *Pattern* tepat sama dan cocok. Selanjutnya algoritma akan memberitahukan penemuan posisi.
3. Selanjutnya, algoritma menggeser *pattern* dengan cara memaksimalkan nilai dari pergeseran *good-suffix* dan *bad-character*, kemudian mengulangi langkah dua sampai *pattern* berada di ujung kanan teks.

Algoritma ini juga memiliki aturan untuk pergeseran *pattern* yaitu *good-suffix rule* dan *bad character rule*.

2.4.1 Good-suffix shift rule

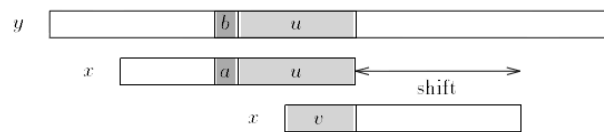
Good-suffix shift rule merupakan perbandingan karakter yang cocok ke karakter *pattern*, aturan *good-suffix shift rule* adalah:

1. Gambar 2.1 menjelaskan pergeseran dari $x[i]=a$ ke karakter lain yang letaknya lebih kiri dari $x[i]$ dan terletak di sebelah kiri segmen u .



Gambar 2. 1 Good-suffix shift, u terjadi lagi didahului karakter c berbeda dari a (Kristanto, Rachmat, & Santosa, 2013)

2. Jika tidak ada segmen yang sama dengan u , maka dicari u yang merupakan *suffix* terpanjang u seperti yang terlihat pada Gambar 2.2.

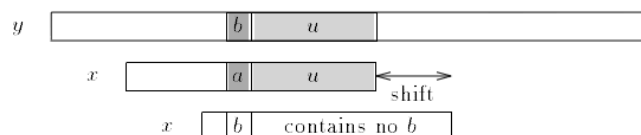


Gambar 2. 2 Good-suffix shift, hanya suffix dari u yang terjadi lagi di pattern x (Kristanto, Rachmat, & Santosa, 2013)

2.4.2 Bad-character rule

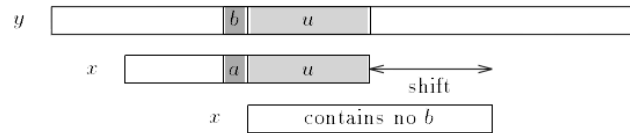
Aturan pada *bad character rule* adalah membandingkan karakter *pattern* yang tidak cocok, aturan ini adalah sebagai berikut:

1. Gambar 2.3 menjelaskan jika *bad-character* $y[i+j]$ terdapat pada *pattern* di posisi tekanan k yang lebih kiri dari $x[i]$ maka *pattern* digeser ke kanan sejauh $i-k$



Gambar 2. 3 Bad-character shift, b terdapat di pattern x (Kristanto, Rachmat, & Santosa, 2013)

2. Jika *bad-character* $y[i+j]$ tidak ada *pattern* sama sekali, maka *pattern* digeser ke kanan sejauh i seperti yang terlihat pada Gambar 2.4.



Gambar 2. 4 *Bad-character shift*, b tidak ada di *pattern* x (Kristanto, Rachmat, & Santosa, 2013)

3. Jika *bad-character* $y[i+j]$ terdapat pada *pattern* di posisi tekanan k yang lebih kanan dari $x[i]$, maka *pattern* seharusnya digeser sejauh $i-k$ yang hasilnya negatif (*pattern* digeser kembali ke kiri). Maka bila kasus ini terjadi akan diabaikan.

Apabila terdapat kasus ketidakcocokan, algoritma akan membandingkan langkah yang diambil oleh *good-suffix shift* dan *bad-character shift* dimana yang akan digunakan adalah langkah yang paling besar (Kristanto, Rachmat, & Santosa, 2013).

2.4 Algoritma *Brute-Force*

Algoritma *Brute-Force* merupakan algoritma pencarian dengan cara membandingkan satu persatu masing-masing karakter *string* dari kiri ke kanan (Utomo, Harjo, & Handoko, 2011). Dalam penerapannya, algoritma *Brute-Force* merupakan algoritma *string* termudah. Diasumsikan bahwa teks berada dalam sebuah *array* $T[1...n]$ dan *pattern* berada di dalam *array* $P[1...m]$, maka pencocokan algoritma *Brute Force* adalah sebagai berikut (Munir, 2004-2007) :

1. *Pattern* P dicocokkan pada awal teks T
2. Bandingkan setiap karakter dalam *pattern* P dengan karakter yang sesuai dalam teks T dengan pergerakan dari kiri ke kanan sampai:
 - Semua karakter dan *pattern* tepat sama atau cocok (pencarian berhasil), atau
 - Diitemukan ketidakcocokan pada karakter (pencarian tidak berhasil)

3. Apabila *pattern* P tidak menemukan kecocokan dan teks T belum habis, maka geser *pattern* P tepat satu karakter ke kanan dan ulangi langkah kedua.

Dalam pencocokan *string* dengan algoritma *Brute-Force* terdapat beberapa persoalan sebagai berikut:

1. Teks (*text*), yaitu *string* yang panjangnya sebanyak n karakter
2. *Pattern*, yaitu *string* dengan panjang m karakter ($m < n$) yang akan dicari dalam teks

Contoh cara kerja algoritma *Brute-Force* adalah sebagai berikut:

Pattern : FORMASI

Teks : INFO INFORM DIINFORMASIKAN

	I	N	F	O		I	N	F	O	R	M		D	I	I	N	F	O	R	M	A	S	I	K	A	N
1	F	O	R	M	A	S	I																			
2		F	O	R	M	A	S	I																		
3			F	O	R	M	A	S	I																	
4				F	O	R	M	A	S	I																
5					F	O	R	M	A	S	I															
6						F	O	R	M	A	S	I														
7							F	O	R	M	A	S	I													
8								F	O	R	M	A	S	I												
9									F	O	R	M	A	S	I											
10										F	O	R	M	A	S	I										
11											F	O	R	M	A	S	I									
12												F	O	R	M	A	S	I								
13													F	O	R	M	A	S	I							
14														F	O	R	M	A	S	I						
15															F	O	R	M	A	S	I					
16																F	O	R	M	A	S	I				
17																	F	O	R	M	A	S	I			

Gambar 2. 5 Contoh cara kerja pencarian algoritma *Brute-Force* (Hidayani, Sari, & Suharman, 2012)

Pada Gambar 2.5 menjelaskan contoh cara kerja pencarian algoritma *Brute-Force*, dimana algoritma ini mencari satu per satu karakter dari kiri ke kanan, algoritma ini tidak akan berhenti mencari apabila tidak menemukan karakter yang cocok sampai data yang ada benar-benar habis.

2.5 Perbandingan Algoritma *Boyer-Moore* dan Algoritma *Brute-Force*

Algoritma *Boyer-Moore* memiliki analisis kompleksitas yang menyangkut tiga hal antara lain kompleksitas waktu, kompleksitas ruang, dan

waktu pemrosesan. Kasus terburuk pada algoritma *Boyer-Moore* terjadi apabila pertama kali masing-masing percobaan membandingkan simbol teks tidak cocok dengan *pattern*. Sehingga kompleksitas waktu terbaik yang dicapai dengan notasi big O yaitu $O(N/M)$. Kompleksitas waktu yang dibutuhkan untuk kasus rata-rata adalah $O(N/M)$.

Algoritma *Boyer-Moore* membutuhkan waktu dalam *preprocessing* fungsi *good suffix-shift* sebesar $O(M)$. Sedangkan untuk kompleksitas pada preprocessing aturan *bad character* adalah $O(M + |\text{alphabet}|)$ untuk kompleksitas ruang dari algoritma *Boyer-Moore* adalah sebesar $O(m + |\text{alphabet}|)$ (Aulia, 2008).

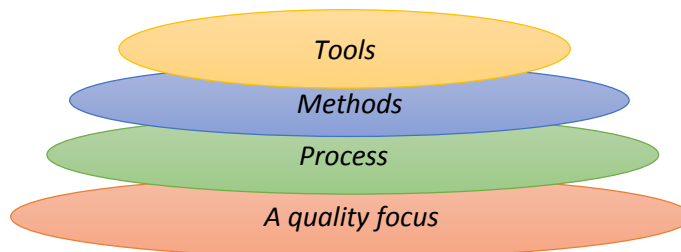
Tabel 2. 5 Perbandingan Algoritma *Boyer-Moore* dan *Brute Force* (Abdeen, 2011)

	<i>Boyer-Moore</i>	<i>Brute-Force</i>
<i>Best Case</i>	n/m	$m+n$
<i>Worse Case</i>	$m+n$	$m.n$
<i>Processing the Pattern</i>	<i>Preprocesses the pattern</i>	<i>No preprocessing</i>
<i>Time Complexity</i>	$O(n.m)$	$O((n-m+1)*m)$
Cara kerja algoritma	Membaca <i>string</i> dari kanan ke kiri	Membaca <i>string</i> dari kiri ke kanan
Kelebihan	Kecepatan <i>pattern</i> yang panjang	Metode sederhana dan mudah dimengerti
Kekurangan	Tidak bagus untuk <i>binary string</i> dan lebih lambat untuk <i>pattern</i> yang pendek	Pergeseran <i>pattern</i> dilakukan tiap satu karakter

Tabel 2.5 menjelaskan perbandingan Algoritma *Boyer-Moore* dan *Brute-Force*, dari tabel tersebut terlihat kelebihan dan kekurangan masing-masing algoritma.

2.6 Model Siklus Pengembangan Perangkat Lunak

Proses pembuatan sebuah perangkat lunak baru yang berfungsi untuk menggantikan perangkat lunak lama secara keseluruhan ataupun hanya memperbaiki perangkat lunak yang ada disebut pengembangan perangkat lunak. Pengembangan perangkat lunak memerlukan suatu metode khusus agar lebih cepat dan tepat dalam proses deskripsi solusi dan pengemangan perangkat lunak. Hasil dari pengembangan perangkat lunak juga lebih mudah untuk dikembangkan dan dipelihara. Metodologi pengembangan perangkat lunak adalah suatu pengorganisasian kumpulan metode dan konvensi notasi yang telah didefinisikan untuk mengembangkan perangkat lunak. Gambar 2.6 menggambarkan batu landasan yang menopang rekayasa perangkat lunak (Pressman, 2010, hal. 14).



Gambar 2. 6 Software Engineering Layers (Pressman, 2010)

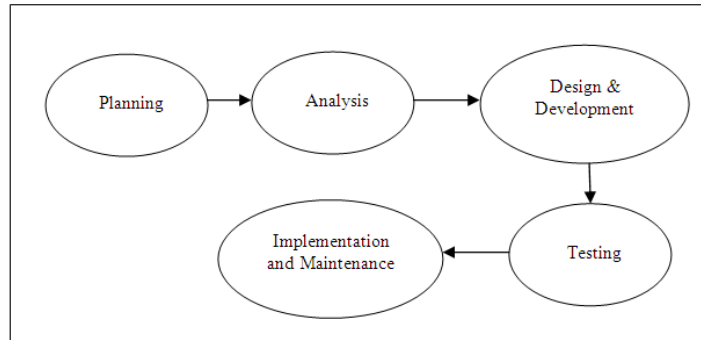
Setiap pengembangan perangkat lunak tidak pernah lepas dari sebuah SDLC (*Software Development Life Cycle*). Banyak model pengembangan yang dipakai dalam SDLC untuk mengembangkan sebuah perangkat lunak aplikasi seperti: *Spiral*, *Waterfall*, *RAD* (*Rapid, Application Development*), dll (Mountaines, 2013). Penggunaan model yang sesuai dengan kebutuhan pada proyek pengembangan aplikasi yang dilakukan akan berdampak pada kualitas aplikasi. Maka dari itu, pengembang aplikasi harus dapat mengukut aplikasi yang akan dibuat sebelum memilih SDLC yang sesuai dalam praktik pengembangan aplikasi. Untuk meningkatkan kualitas pengembangan aplikasi, maka perlu untuk mengetahui karakteristik dari masing-masing model pengembangan dan menyesuaikan kebutuhan dari beberapa aspek dalam proyek yang dilakukan. Tabel 2.6 menjelaskan perbandingan model pengembangan aplikasi dari masing-masing model.

Tabel 2. 6 Perbandingan model pengembangan aplikasi (Mujumdar, Masiwal, & Chawan, 2012)

<i>Model/Features</i>	<i>Waterfall</i>	<i>Incremental</i>	<i>Spiral</i>	<i>Agile</i>	<i>RUP</i>
<i>Requirement Specifications</i>	<i>Beginning</i>	<i>Beginning</i>	<i>Beginning</i>	<i>Frequently changed</i>	<i>Beginning</i>
<i>Cost</i>	<i>Low</i>	<i>Low</i>	<i>Expensive</i>	<i>Very High</i>	<i>Expensive</i>
<i>Resource Control</i>	<i>Yes</i>	<i>Yes</i>	<i>Yes</i>	<i>No</i>	<i>Yes</i>
<i>Simplicity</i>	<i>Simple</i>	<i>Intermediate</i>	<i>Intermediate</i>	<i>Complex</i>	<i>Simple and clear</i>
<i>Risk Analysis</i>	<i>Only at beginning</i>	<i>Intermediate</i>	<i>High</i>	<i>High</i>	<i>Only at beginning of last phase</i>
<i>Flexibility</i>	<i>Rigid</i>	<i>Less Flexible</i>	<i>Flexible</i>	<i>Highly Flexible</i>	<i>Considerable</i>
<i>Reusability</i>	<i>Limited</i>	<i>Yes</i>	<i>Yes</i>	<i>Use Case reuse</i>	<i>Support reusability of existing classes</i>

Web Development Life Cycle (WDLC) adalah suatu metodologi baru yang diusulkan khusus untuk pengembangan aplikasi *web*. Metodologi ini didasarkan pada metodologi sebelumnya yang ditemukan dalam literatur untuk menciptakan suatu proses terstruktur untuk masalah yang sangat terstruktur dari pengembangan aplikasi *web* itu sendiri. WDLC adalah hibrida dari dua metodologi sebelumnya yang dikenal sebagai *Systems Development Life Cycle and Prototyping*. WDLC menggunakan komponen dari masing-masing metodologi, menggabungkan ke dalam sebuah pendekatan baru yang akan mengurangi waktu pengembangan, menambahkan struktur untuk masalah yang tidak terstruktur dan menjaga pengguna yang terlibat dalam seluruh siklus hidup pengembangan (French, 2011).

Pada Gambar 2.7 menggambarkan lima tahapan WDLC yang memungkinkan proses perancangan selesai. Masing-masing tahapan mencakup seperangkat tugas, yang mengandalkan teknik yang menghasilkan *file* dokumen tertentu untuk memahami proyek (Kamatchi, Iyer, & Singh, 2013).



Gambar 2. 7 Web Development Life Cycle Model (WDLC) (Kamatchi, Iyer, & Singh, 2013)

1. Website Planning

Fase pertama dalam WDLC adalah *planning*. Beberapa tahapan yang harus dilakukan antara lain:

- a. Mengidentifikasi tujuan dari *website* yang akan dibangun, sehingga dapat menentukan perencanaan secara tepat
- b. Memahami siapakah yang akan menggunakan *website*, hal ini merupakan tahapan untuk mengidentifikasi target pengguna, menentukan halaman yang akan diakses oleh pengguna, dan mengidentifikasi teknologi yang dibutuhkan dalam mengakses *website* tersebut
- c. Memahami teknologi *website* apa yang akan digunakan, seperti *web browser*, akses internet, dan resolusi layar monitor
- d. Mengidentifikasi isi konten dari *website* yang dibedakan berdasarkan penggunaanya
- e. Menentukan informasi apa saja yang perlu diletakkan di dalam *website* tersebut.

2. *Website Analysis*

Tahapan ini merupakan rangkaian aktivitas dimana seorang analis menggabungkan informasi yang dibutuhkan oleh pengguna, menganalisis kebutuhan fungsional dari sistem, kebutuhan masukan data dan sumber daya, serta kebutuhan presentasi dan keluaran data. Berikut ini adalah tahapan-tahapan yang perlu dilakukan.

- a. Mengidentifikasi tugas atau pekerjaan yang harus diselesaikan oleh masing-masing pengguna sistem
- b. Mengidentifikasi *site map*, menentukan struktur dari *website*, dan finalisasi konten yang akan diletakkan pada halaman *web*
- c. Menganalisa kualitas kebutuhan data yang memang benar-benar dibutuhkan oleh pengguna agar dapat dihasilkan keluaran yang benar dan tepat.

3. *Web Page Design and Development*

Tahapan ini meliputi *blue print* dari *website* dengan mempresentasikannya ke dalam desain *logical* dan *physical* yang akan dibangun pada tahapan *development*. Desain yang dibuat meliputi *data models*, *process models*, dan *presentation models*. Desain tersebut dibuat dalam bentuk dokumen sebagai panduan dalam pengembangan dan pengujian sistem.

Seorang *developer* memiliki tanggung jawab dalam pembangunan kode program, dan membuat *data sets* untuk masukan serta memverifikasi bahwa program dapat menghasilkan keluaran sesuai yang diharapkan oleh pengguna. Hanya pada tahapan ini, konseptual *website* diterjemahkan ke dalam sebuah *website* yang bermanfaat dan atraktif.

4. *Website Testing*

Tim pengembangan mendemonstrasikan *website* kepada pengguna. Mereka memastikan bahwa *website* berjalan sesuai dengan yang diharapkan oleh pengguna. Hal ini meliputi perencanaan

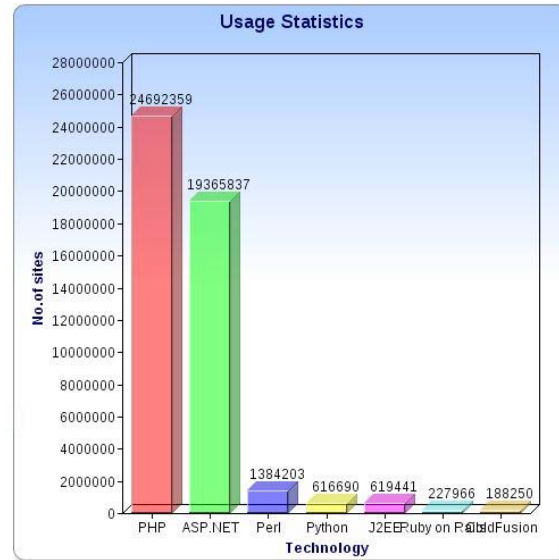
pengujian, membuat *text data*, mengeksekusi *text runs*, mencocokkan hasil teks sesuai dengan yang diharapkan, menganalisa dan memperbaiki *bugs* yang terjadi hingga tidak terjadi kesalahan. *Website* harus diuji pada tahapan yang berbeda, yang meliputi *content*, *functionality*, *usability*, dan *correctness*.

5. *Website Implementation and Maintenance*

Tahapan ini meliputi instalasi *website* pada sistem komputer yang dipakai oleh pengguna. Selanjutnya adalah tahapan pemeliharaan *website* yang ditujukan untuk memastikan bahwa kebutuhan informasi masih sesuai dengan yang diharapkan. Hal ini dapat menjaga agar *website* tetap *up to date*.

2.7 Bahasa Pemrograman

Teknologi layanan *web* didasarkan pada konsep komputasi berorientasi layanan-layanan *web* standar yang mengintegrasikan aplikasi berbasis *web* melalui menghubungkan dan berbagi proses bisnis di seluruh jaringan di mana aplikasi dari vendor yang berbeda, bahasa dan *platform* komunikasi satu sama lain dengan klien (Al-Fedaghi, 2011). Beberapa teknologi terkemuka yang sebagian besar digunakan dalam mengembangkan dan menerapkan aplikasi berbasis *web* adalah ASP.NET, *Hypertext Preprocessor* (PHP) , *ColdFusion*, *Perl*, *Python*, *Java 2 Enterprise Edition* (J2EE), *Ruby on Rails* dll. Dari statistik pada Gambar 2.8 menunjukkan bahwa PHP berada pada posisi teratas yang paling sering digunakan sekitar 24692359 *websites*, dan posisi kedua adalah ASP.NET yang digunakan lebih dari 19365837 *websites* (Nagila, 2013).



Gambar 2. 8 Usage Statistics of Web Technologies (Nagila, 2013)

Salah satu aspek penting yang dicatat selama pengembangan adalah penggunaan memori. ASP.NET terlihat cukup mahal dengan penggunaan memori yang dapat menjadi masalah serius ketika mengembangkan aplikasi *web* yang lebih besar. Sedangkan penggunaan memori pada PHP lebih efisien daripada ASP.NET. Alasannya adalah PHP memiliki *code path* kecil yang berarti kode sisi *server* lebih sedikit jika dibandingkan dengan ASP.NET. Tabel 2.7 menunjukkan tabel perbandingan ASP.NET dan PHP. (Chandran & Angepat, 2011).

Tabel 2. 7 Tabel Perbandingan antara ASP.NET dan PHP (Chandran & Angepat, 2011)

Pengukuran	ASP.NET	PHP
Biaya	<p>Program ASP perlu IIS untuk dipasang di <i>platform Windows server</i>.</p> <p>Konektivitas <i>database</i> mahal; MS-SQL adalah produk Microsoft</p>	<p>Program PHP berjalan di <i>Apache</i> pada <i>server Linux</i> dan <i>Unix</i> secara gratis.</p> <p>Menggunakan MySQL sebagai <i>database</i> (gratis)</p>

Pengukuran	ASP.NET	PHP
Kecepatan	ASP.NET adalah bahasa yang dioptimalkan, dikompilasi dan lebih cepat dalam eksekusi	PHP adalah bahasa yang ditafsirkan dan kurang cepat dalam eksekusi
Penggunaan Memori	Panjangnya <i>code path</i> membuat penggunaan memori lebih mahal	Kecilnya <i>code path</i> membuat penggunaan memori lebih efisien
<i>Support and Resources</i>	Perbaikan dan pembaruan dibuat oleh jumlah yang tersedia dari <i>developer</i> Microsoft sendiri. Kurang dukungan yang tersedia untuk memecahkan tantangan baru.	Lebih banyak pengembang <i>open source</i> dan sumber daya yang tersedia dari forum PHP. Dukungan lebih tersedia dari forum PHP
Editor dan perangkat	Paling banyak <i>Microsoft Visual Studio</i> yang digunakan untuk membangun aplikasi .NET	Editor independen. Memiliki akses ke editor dalam jumlah yang luas.
Pengembangan dan <i>coding</i>	Rata-rata waktu pengembangan lebih lama untuk situs yang lebih kecil.	Rata-rata waktu pengembangan lebih singkat untuk situs yang lebih kecil.

2.8 Unified Modeling Language (UML)

UML adalah suatu kumpulan pemodelan yang digunakan untuk menggambarkan sebuah sistem *software* yang terkait dengan objek, yang dijelaskan sebagai berikut (Whitten & Bentley, 2007).

- b. Objek, merupakan sesuatu yang dapat dilihat, disentuh, atau dirasakan. Sehingga *user* dapat menyimpan serta melakukan pencatatan perilaku mengenai objek tersebut. Dan memiliki dua karakteristik yaitu:
 1. Atribut, merupakan data yang mewakili karakteristik *interest* mengenai sebuah objek.
 2. *Behavior*, adalah kumpulan dari aktivitas yang dapat dilakukan oleh objek dan terkait dengan fungsi-fungsi yang bertindak pada suatu data objek (atribut). Dan pada siklus berorientasi objek, perilaku objek merujuk kepada metode, operasi, atau fungsi.
- c. Kelas, merupakan suatu set objek yang memiliki atribut dan *behavior* yang sama, biasanya disebut dengan *object class*.
- d. Generalisasi/Specialisasi, merupakan sebuah teknik dimana atribut dan *behavior* yang umum pada beberapa tipe kelas objek, akan dikelompokkan (atau diabstraksi) ke dalam kelasnya sendiri, disebut sebagai *supertype*. Atribut dan metode kelas objek *supertype* kemudian akan diwariskan oleh kelas objek tersebut (*subtype*).
- e. *Inheritance*, merupakan konsep dimana metode atau atribut yang ditentukan di dalam sebuah *object class* lainnya.

Menurut penelitian yang dilakukan oleh Nabil Mohammed Ali Munassar yang berjudul *Comparison Between Traditional Approach and Object Oriented Approach in Software Engineering Development* (2011), di dalam UML terdapat beberapa diagram yang digunakan untuk menjelaskan sistem berorientasi objek (Munassar & Govardan, 2012).

- a. *Use Case*

Use case merupakan deskripsi secara *static* yang menggambarkan bagaimana sistem itu digunakan oleh konsumen atau *user* dan sistem lainnya. Selain itu juga, *use case* diagram menjelaskan hubungan satu

sama lainnya di dalam sistem. Lingkaran pada *use case* mempresentasikan aktivitas sedangkan *person* menggambarkan *user*.

b. *Class Diagram*

Class diagram menggambarkan kelas-kelas yang terdapat pada sistem. Di dalam *class diagram* terdapat kotak yang menggambarkan kelas itu sendiri serta hubungan antar kelas. Di dalam kotak tersebut terdapat *function* yang bisa digunakan dari kelas tersebut.

c. *Sequence Diagram*

Sequence diagram menggambarkan tentang interaksi antara objek pada sistem. *Sequence* digunakan selama desain *subsystem* dan merupakan pemodelan dinamis selama analisa, desain sistem bahkan penangkapan kebutuhan pada sistem.

2.9 Pengujian

Salah satu metode yang digunakan untuk menentukan urutan prioritas dan alternatif keputusan dengan kriteria jamak adalah Metode Perbandingan Eksponensial (MPE). Dalam pengerjaannya, metode perbandingan eksponensial memiliki beberapa prosedur antara lain (Januardi, 2013):

1. Menyusun alternatif-alternatif
2. Menentukan kriteria atau perbandingan
3. Menentukan tingkat kepentingan pada setiap kriteria keputusan
4. Melakukan penilaian terhadap semua alternatif pada setiap kriteria
5. Menghitung skor atau sebuah nilai total setiap alternatif
6. Menentukan urutan prioritas keputusan.

Adapun rumus matematika yang dipakai dalam Metode Perbandingan Eksponensial adalah:

$$Total\ Nilai\ (TNi) = \sum_{i=1}^m (RK_{ij})^{TKK_j} \dots\dots\dots (Rumus\ 2.1)$$

Rumus 2. 1 Rumus Metode Perbandingan Eksponensial (Januardi, 2013)

Keterangan:

TNi : Total nilai alternatif ke-i

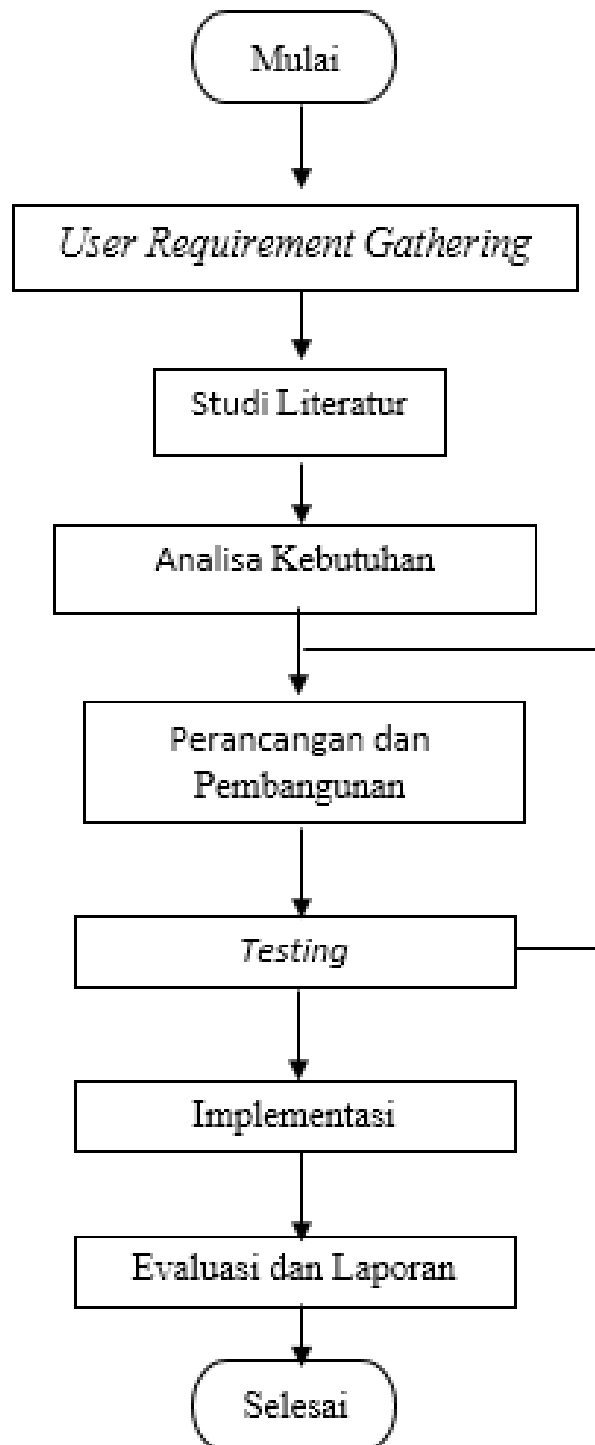
RK_{ij} : Derajat kepentingan relatif kriteria ke-j pada pilihan keputusan i
 TKK_j : Derajat kepentingan kriteria keputusan ke-j; $TKK_j > 0$
 m : Jumlah kriteria keputusan
 n : Jumlah pilihan keputusan
 j : 1,2,3,...m; m : Jumlah kriteria
 i : 1,2,3,...n; n : Jumlah pilihan kriteria (Marimin, 2015)

BAB III

METODOLOGI PENELITIAN

3.1 Kerangka Penelitian

Pada penelitian ini memiliki tahapan atau aktivitas yang dilakukan seperti pada Gambar 3.1.



Gambar 3. 1 Kerangka Penelitian

3.2 Metode Perancangan dan Pengembangan

Penelitian ini menggunakan metode WDLC (Kamatchi, Iyer, & Singh, 2013) yang merupakan gabungan dari metode SDLC dengan metode *prototype*. Metode ini akan diterapkan pada pembangunan aplikasi LF-Pro Universitas Bakrie. Penelitian ini dilakukan selama tujuh bulan (Lampiran 3). Dalam penelitian ini, penggunaan metode pengembangan diterapkan dalam fase-fase sebagai berikut.

3.1.1 Pengamatan dan Perencanaan

Pada fase ini dilakukan pengamatan selama dua minggu mengenai masalah yang ada, melakukan wawancara dengan pihak *security* dan beberapa mahasiswa terkait dengan masalah yang ada untuk menganalisa dan melakukan perumusan masalah. Selain melakukan wawancara, pengumpulan informasi juga dilakukan dengan studi kepustakaan, yaitu dengan *review* buku dan jurnal.

3.1.2 Analisa Kebutuhan Aplikasi

Pada tahapan ini akan menggabungkan seluruh informasi yang diperoleh pada tahap sebelumnya, kemudian menganalisis kebutuhan fungsional dan non-fungsional dari sistem tersebut, menganalisis kebutuhan data masukan dan data keluaran. Hasil dari analisis tersebut berupa elisitasi.

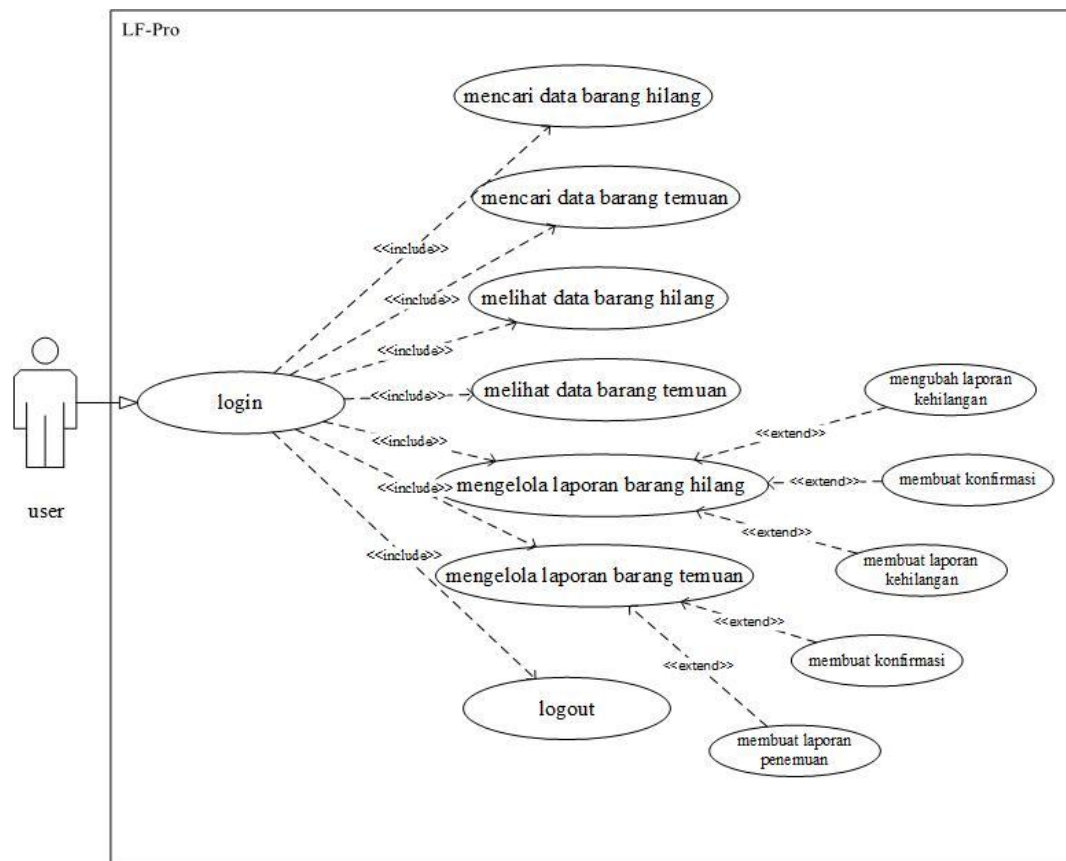
Selain itu, pada tahap ini dilakukan analisis *user requirement*. Terdapat dua *user requirement* yaitu *functional requirement* dan *non-functional requirement*. Keduanya akan menampilkan detail kebutuhan yang diinginkan oleh *user*. Pada tahap ini juga akan menjelaskan secara detail sistem yang akan dibuat dan menentukan informasi yang akan ditampilkan dalam sistem. Hasil kebutuhan aplikasi. Hasil analisa kebutuhan aplikasi yang terlampir pada Lampiran 2.

3.1.3 Perancangan dan Pembangunan

Pada tahap ini akan dilakukan proses perancangan sistem berdasarkan hasil dari tahapan sebelumnya. Hasil dari perancangan sistem ini akan digunakan pada tahap pembangunan.

1. Use Case Diagram

Use case diagram menggambarkan apa saja fitur yang ada dalam perancangan aplikasi ini. Berikut use case diagram dari perancangan aplikasi LF-Pro.



Gambar 3. 2 Use Case Diagram Aplikasi LF-Pro

Dalam aplikasi LF-Pro, *actor* yang menjalankan aplikasi hanya dilakukan oleh *user* karena *user* akan secara otomatis melakukan aktivitas, dan tidak ada pengelolaan data lebih lanjut, LF-Pro hanya berfungsi sebagai media penyimpanan data dan pencarian barang hilang, sehingga LF-Pro tidak membutuhkan *admin* untuk mengelola data lebih lanjut.

Deskripsi Gambar 3.2 dari *use case diagram* di atas akan dijelaskan secara lebih detail dalam Tabel *use case scenario* berikut:

Tabel 3. 1 Use Case Scenario Login

Use case name	Login	
Use case ID	1	
Actor	<i>User</i>	
Description	<i>Use case</i> ini menggambarkan kegiatan pada saat <i>user</i> melakukan <i>login</i>	
Pre-Condition	<i>User</i> membuka aplikasi pada <i>browser</i>	
Trigger	<i>Use case</i> ini dilakukan agar <i>actor</i> dapat masuk ke dalam aplikasi.	
Typical of Event	Actor Action	System Response
	1. Membuka aplikasi	2. Menampilkan halaman untuk <i>login</i> ke dalam sistem
	3. Memasukkan <i>username</i> dan <i>password</i>	4. Cek validasi <i>username</i> dan <i>password</i>
		5. Menampilkan halaman sesuai hak akses yang dimiliki <i>actor</i>
Alternate Course	Jika <i>username</i> dan <i>password</i> salah, maka akan muncul notifikasi lalu Aktor harus melakukan <i>login</i> kembali.	
Post-Condition	Aplikasi menampilkan halaman home.	

Tabel 3. 2 Use Case Scenario Mencari Data Barang Hilang

Use case name	Mencari data barang hilang	
Use case ID	2	
Actor	<i>User</i>	
Description	<i>Use case</i> ini menggambarkan kegiatan pada saat <i>user</i> melakukan pencarian data barang hilang pada <i>database</i> dengan fitur <i>auto-complete</i>	
Pre-Condition	<i>User</i> membuka aplikasi pada <i>browser</i>	
Trigger	<i>Use case</i> ini dilakukan agar <i>actor</i> dapat mencari barang hilang yang terdapat pada sistem	
Typical of Event	Actor Action	System Response
	1. Membuka aplikasi	2. Menampilkan halaman untuk <i>login</i> ke dalam sistem

	3. Memasukkan <i>username</i> dan <i>password</i>	4. Cek validasi <i>username</i> dan <i>password</i>
		5. Menampilkan halaman sesuai hak akses yang dimiliki <i>actor</i>
	6. Mengisi <i>field search</i> barang hilang	7. Menampilkan <i>auto-complete</i> sesuai dengan <i>string</i> yang dicari
	8. Klik tombol <i>search</i>	9. Menampilkan data sesuai dengan yang dicari
Alternate Course	Jika barang yang dicari tidak ada di <i>database</i> maka akan muncul <i>alert</i> dan <i>actor</i> diharapkan untuk menambah data barang hilang.	
Post-Condition	Aplikasi menampilkan halaman data barang hilang yang dicari	

Tabel 3. 3 Use Case Scenario Mencari Data Barang Temuan

Use case name	Mencari data barang temuan	
Use case ID	3	
Actor	<i>User</i>	
Description	<i>Use case</i> ini menggambarkan kegiatan pada saat <i>user</i> melakukan pencarian data barang temuan pada <i>database</i> dengan fitur <i>auto-complete</i>	
Pre-Condition	<i>User</i> membuka aplikasi pada <i>browser</i>	
Trigger	<i>Use case</i> ini dilakukan agar <i>actor</i> dapat mencari barang temuan yang terdapat pada sistem	
Typical of Event	Actor Action	System Response
	1. Membuka aplikasi	2. Menampilkan halaman untuk <i>login</i> ke dalam sistem
	3. Memasukkan <i>username</i> dan <i>password</i>	4. Cek validasi <i>username</i> dan <i>password</i>
		5. Menampilkan halaman sesuai hak akses yang dimiliki <i>actor</i>
	6. Mengisi <i>field search</i> barang hilang	7. Menampilkan <i>auto-complete</i> sesuai dengan <i>string</i> yang dicari
	8. Klik tombol <i>search</i>	9. Menampilkan data sesuai dengan yang dicari
Alternate Course	Jika barang yang dicari tidak ada di <i>database</i> maka akan muncul <i>alert</i> dan <i>actor</i> diharapkan untuk menambah data barang hilang.	
Post-Condition	Aplikasi menampilkan halaman data barang temuan yang dicari.	

Tabel 3. 4 *Use Case Scenario* Melihat Data Barang Hilang

Use case name	Melihat data barang hilang	
Use case ID	4	
Actor	User	
Description	Use case ini menggambarkan kegiatan untuk melihat data Barang Hilang yang ada dalam bentuk Tabel	
Pre-Condition	User membuka aplikasi pada browser	
Trigger	Use case ini dilakukan agar actor dapat mencari barang hilang yang terdapat pada sistem	
Typical of Event	Actor Action	System Response
	1. Membuka aplikasi	2. Menampilkan halaman untuk login ke dalam sistem
	3. Memasukkan username dan password	4. Cek validasi username dan password
		5. Menampilkan halaman sesuai hak akses yang dimiliki actor
	6. Klik menu barang hilang	7. Menampilkan data barang hilang
Alternate Course	Tidak terdapat <i>scenario error input</i> pada tahap ini.	
Post-Condition	Aplikasi menampilkan halaman data barang hilang	

Tabel 3. 5 *Use Case Scenario* Melihat Data Barang Temuan

Use case name	Melihat data barang temuan	
Use case ID	5	
Actor	User	
Description	Use case ini menggambarkan kegiatan untuk melihat data Barang Temuan yang ada dalam bentuk Tabel	
Pre-Condition	User membuka aplikasi pada browser	
Trigger	Use case ini dilakukan agar actor dapat mencari barang temuan yang terdapat pada sistem	
Typical of Event	Actor Action	System Response
	1. Membuka aplikasi	2. Menampilkan halaman untuk login ke dalam sistem
	3. Memasukkan username dan password	4. Cek validasi username dan password

		5. Menampilkan halaman sesuai hak akses yang dimiliki <i>actor</i>
	6. Klik menu barang hilang	7. Menampilkan data barang hilang
Alternate Course	Tidak terdapat <i>scenario error input</i> pada tahap ini.	
Post-Condition	Aplikasi menampilkan halaman data barang temuan.	

Tabel 3. 6 Use Case Scenario Mengedit Barang Hilang

Use case name	Mengelola Barang Hilang	
Use case ID	6	
Actor	<i>User</i>	
Description	<i>Use case</i> ini menggambarkan kegiatan untuk mengelola barang hilang yang telah disimpan dalam <i>database</i>	
Pre-Condition	<i>User</i> membuka aplikasi pada <i>browser</i>	
Trigger	<i>Use case</i> ini dilakukan agar <i>actor</i> dapat mengelola barang hilang yang terdapat pada sistem	
Typical of Event	Actor Action	System Response
	1. Membuka aplikasi	2. Menampilkan halaman untuk <i>login</i> ke dalam sistem
	3. Memasukkan <i>username</i> dan <i>password</i>	4. Cek validasi <i>username</i> dan <i>password</i>
		5. Menampilkan halaman sesuai hak akses yang dimiliki <i>actor</i>
	6. Klik menu barang hilang	7. Menampilkan data barang hilang
	8. Klik ikon edit	9. Menampilkan data barang yang akan diedit
	10. Insert data yang akan di edit kemudian tekan ikon <i>update</i>	11. Meng- <i>update</i> data barang hilang dan menampilkan notifikasi bahwa data telah di <i>update</i> , kemudian sistem akan langsung menuju data barang hilang
Alternate Course	Tidak terdapat <i>scenario error input</i> pada tahap ini.	
Post-Condition	Aplikasi menampilkan halaman data barang hilang.	

Tabel 3. 7 *Use Case Scenario* Membuat Laporan Kehilangan

Use case name	Membuat Laporan Kehilangan	
Use case ID	7	
Actor	User	
Description	Use case ini menggambarkan kegiatan untuk membuat laporan kehilangan dan akan disimpan dalam <i>database</i>	
Pre-Condition	User membuka aplikasi pada <i>browser</i>	
Trigger	Use case ini dilakukan agar <i>actor</i> dapat menambahkan data barang hilang	
Typical of Event	Actor Action	System Response
	1. Membuka aplikasi	2. Menampilkan halaman untuk <i>login</i> ke dalam sistem
	3. Memasukkan <i>username</i> dan <i>password</i>	4. Cek validasi <i>username</i> dan <i>password</i>
		5. Menampilkan halaman sesuai hak akses yang dimiliki <i>actor</i>
	6. Klik menu barang hilang	7. Menampilkan data barang hilang
	8. Klik <i>button</i> tambahkan data	9. Menampilkan formulir data barang dan data pelapor
	10. Input data barang hilang dan data pelapor dan klik <i>button</i> tambah	11. System akan menginputkan data ke dalam <i>database</i> dan akan menampilkan notifikasi bahwa data telah di input
Alternate Course	Apabila ada data barang atau data pelapor yang tidak diisi maka akan muncul notifikasi, dan <i>user</i> harus melengkapi data tersebut	
Post-Condition	Aplikasi menampilkan halaman data barang hilang.	

Tabel 3. 8 *Use Case Scenario* Membuat Konfirmasi Barang

Use case name	Membuat Konfirmasi Barang
Use case ID	8
Actor	User
Description	Use case ini menggambarkan kegiatan untuk membuat laporan kehilangan dan akan disimpan dalam <i>database</i>
Pre-Condition	User membuka aplikasi pada <i>browser</i>

Trigger	<i>Use case</i> ini dilakukan agar <i>actor</i> dapat membuat konfirmasi barang apabila barang telah ditemukan	
Typical of Event	Actor Action	System Response
	1. Membuka aplikasi	2. Menampilkan halaman untuk <i>login</i> ke dalam sistem
	3. Memasukkan <i>username</i> dan <i>password</i>	4. Cek validasi <i>username</i> dan <i>password</i>
		5. Menampilkan halaman sesuai hak akses yang dimiliki <i>actor</i>
	6. Klik menu barang hilang	7. Menampilkan data barang hilang
	8. Klik <i>detail</i> barang	9. Menampilkan <i>detail</i> barang dan pelapor
	10. Klik ikon konfirmasi	11. Menampilkan formulir konfirmasi barang
Alternate Course	Apabila ada data pelapor yang tidak diisi maka akan muncul notifikasi, dan <i>user</i> harus melengkapi data tersebut	
Post-Condition	Aplikasi menampilkan notifikasi barang dan mengirim konfirmasi via <i>e-mail</i> .	

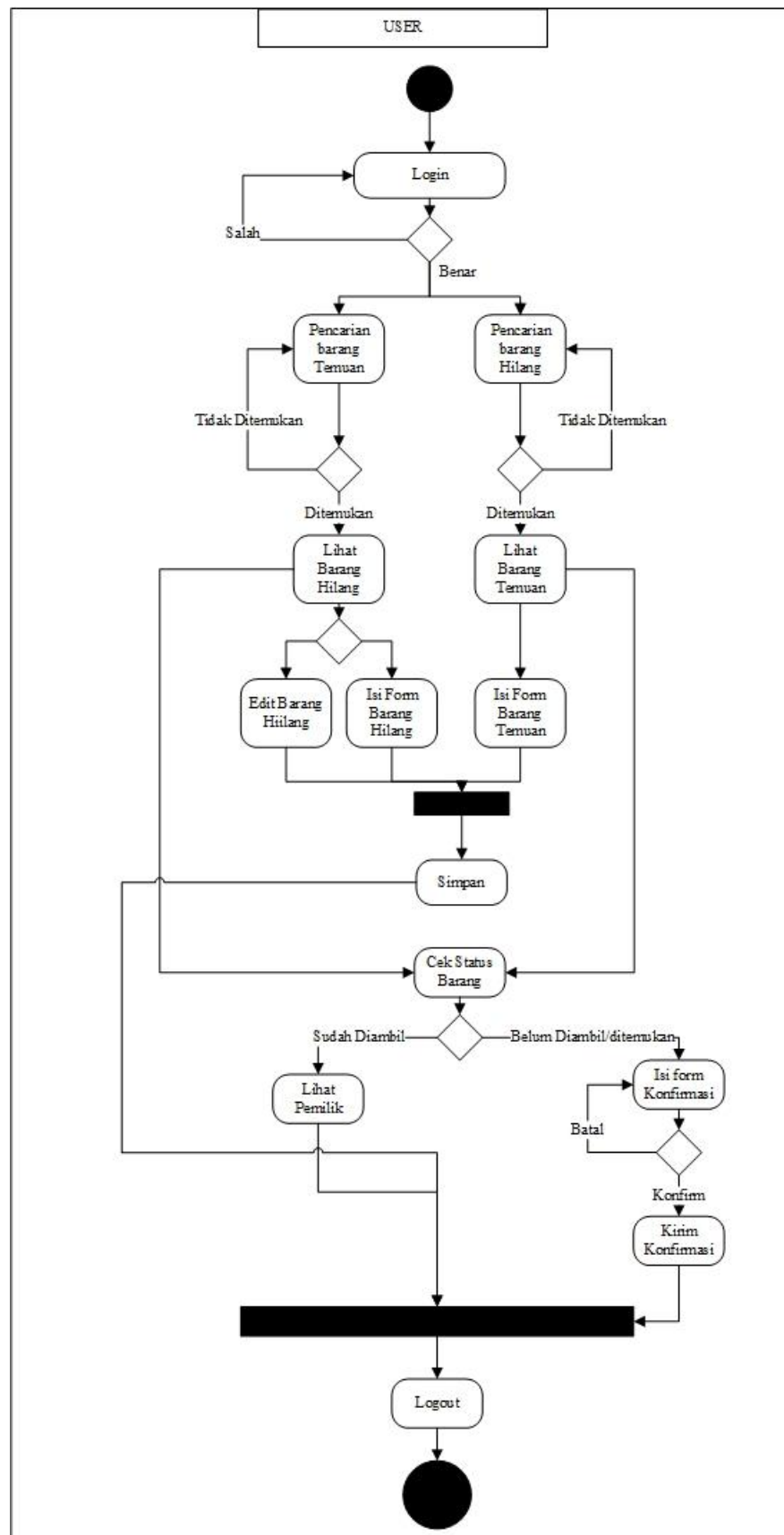
Tabel 3. 9 *Use Case Scenario* Membuat Laporan Penemuan

Use case name	Membuat Laporan Penemuan	
Use case ID	9	
Actor	<i>User</i>	
Description	<i>Use case</i> ini menggambarkan kegiatan untuk membuat laporan penemuan dan akan disimpan dalam <i>database</i>	
Pre-Condition	<i>User</i> membuka aplikasi pada <i>browser</i>	
Trigger	<i>Use case</i> ini dilakukan agar <i>actor</i> dapat menambahkan data barang temuan	
Typical of Event	Actor Action	System Response
	1. Membuka aplikasi	2. Menampilkan halaman untuk <i>login</i> ke dalam sistem
	3. Memasukkan <i>username</i> dan <i>password</i>	4. Cek validasi <i>username</i> dan <i>password</i>
		5. Menampilkan halaman sesuai hak akses yang dimiliki <i>actor</i>

	6. Klik menu barang temuan	7. Menampilkan data barang temuan
	8. Klik <i>button</i> tambahkan data	9. Menampilkan formulir data barang dan data pelapor
	10. Input data barang temuan dan data pelapor dan klik <i>button</i> tambah	11. System akan menginputkan data ke dalam <i>database</i> dan akan menampilkan notifikasi bahwa data telah di input
Alternate Course	Apabila ada data barang atau data pelapor yang tidak diisi maka akan muncul notifikasi, dan <i>user</i> harus melengkapi data tersebut	
Post-Condition	Aplikasi menampilkan halaman data barang temuan.	

2. Activity Diagram

Activity Diagram menggambarkan alur aktivitas yang memungkinkan dapat dilakukan pada aplikasi mulai dari awal proses hingga proses berakhir. Gambar 3.3 menggambarkan *activity diagram* pada aplikasi LF-Pro.



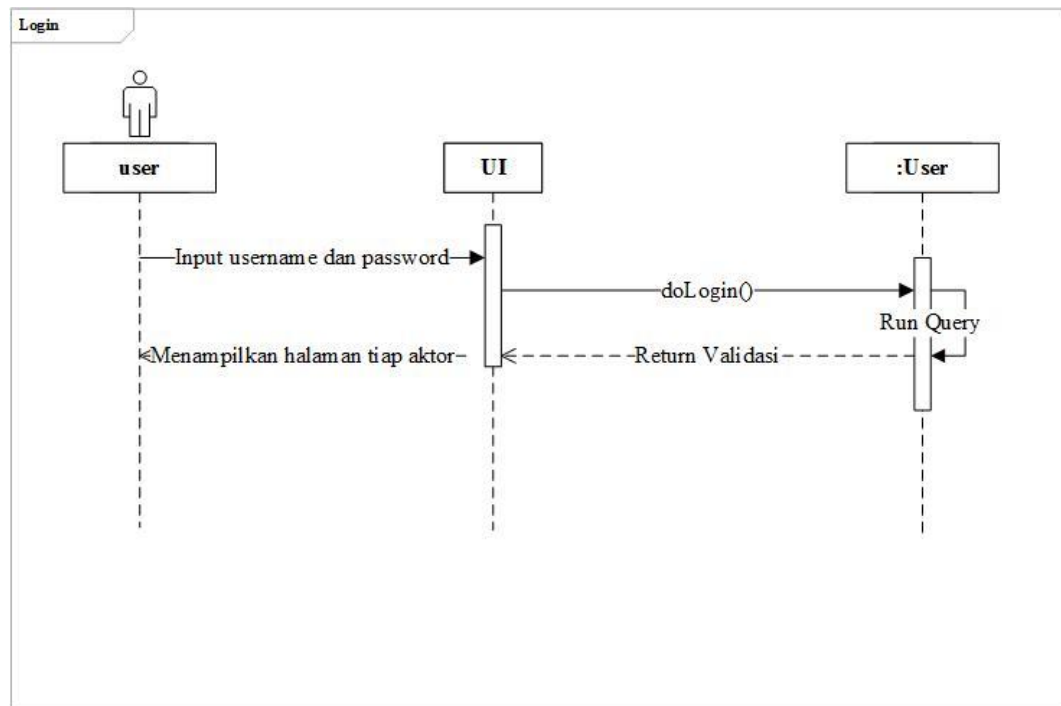
Gambar 3.3 Activity Diagram Aplikasi LF-Pro

Gambar 3.3 di atas merupakan *activity diagram* aplikasi LF-Pro. Langkah awal yaitu *user* melakukan *login* ke sistem dengan memasukkan nomor induk dan *password*. Jika nomor induk dan *password* yang dimasukkan salah, maka harus mengulang proses *login*. Namun jika proses *login* telah benar maka *user* akan masuk ke halaman home. *User* dapat melakukan pencarian barang sesuai dengan nama barang. Apabila barang ditemukan maka *user* dapat melihat detail barang sedangkan apabila barang tidak ditemukan di *database* maka *user* harus mengulang pencarian. Untuk barang hilang, *user* dapat melakukan aktivitas edit barang, cek status dan mengisi form barang sedangkan untuk barang temuan *user* dapat mengisi form dan cek status barang saja.

Untuk melakukan pengecekan status barang *user* diharapkan melihat status barang yang ada, apabila barang sudah diambil/sudah ditemukan maka *user* hanya dapat melihat pemilik barang/orang yang melakukan konfirmasi, sedangkan apabila status barang belum diambil/ditemukan maka *user* diharapkan mengisi form konfirmasi, form konfirmasi yang telah valid akan mengirimkan hasil konfirmasi via email kepada pemilik barang. Setelah semua aktivitas selesai, *user* dapat keluar dari sistem dengan menekan tombol untuk *logout*.

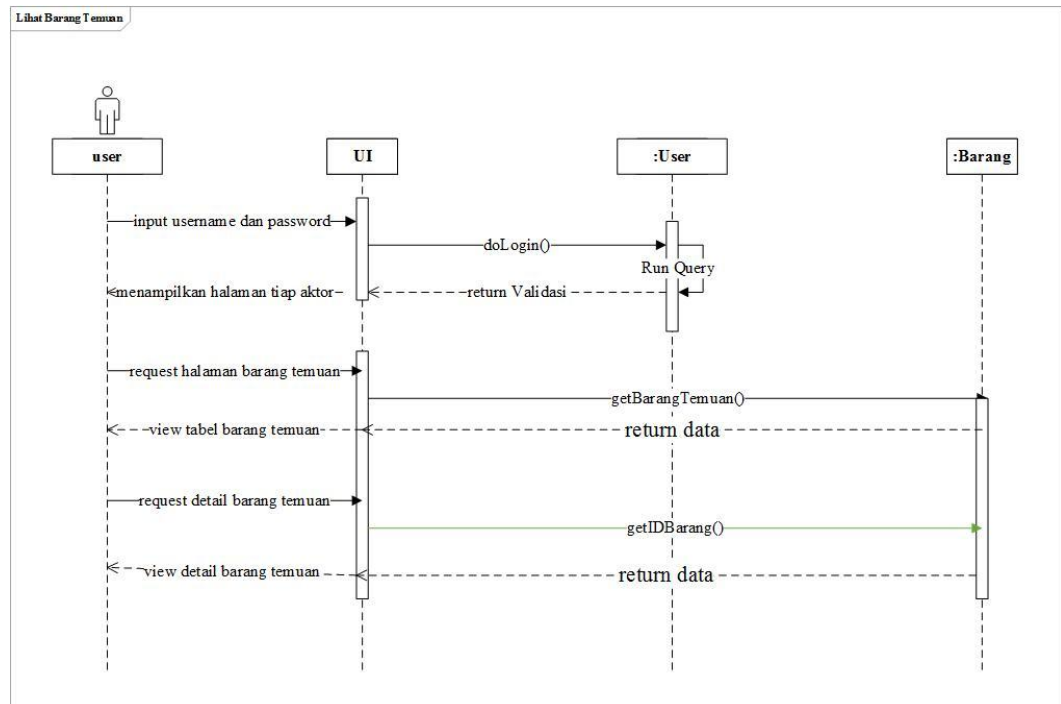
3. *Sequence Diagram*

Sequence diagram menggambarkan interaksi antar objek serta mengilustrasikan urutan pesan yang terjadi selama aplikasi LF-Pro Universitas Bakrie dijalankan yang terdapat pada *sequence diagram* berikut.



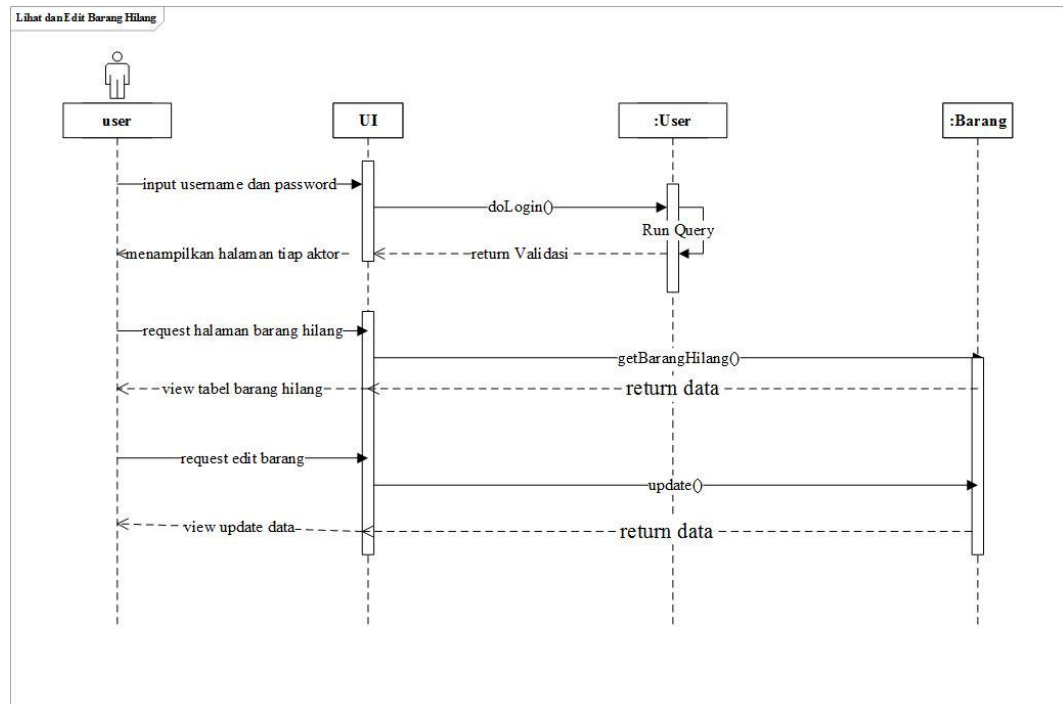
Gambar 3. 4 Sequence Diagram Halaman Login

Gambar 3.4 menjelaskan apa saja yang terlibat serta apa saja yang terjadi ketika *user* melihat halaman awal aplikasi. Sistem terlebih dahulu menjalankan UI untuk menampilkan perintah pada *user* dimana *user* dapat memasukkan nomor induk dan *password*. Kemudian sistem akan mengecek ke dalam *database* dengan memanggil fungsi `doLogin()` pada kelas *User* dan melakukan pengecekan validasi `Run Query`, kemudian mengirimkan hasil validasi ke tampilan untuk diberitahukan kepada pengguna aplikasi.



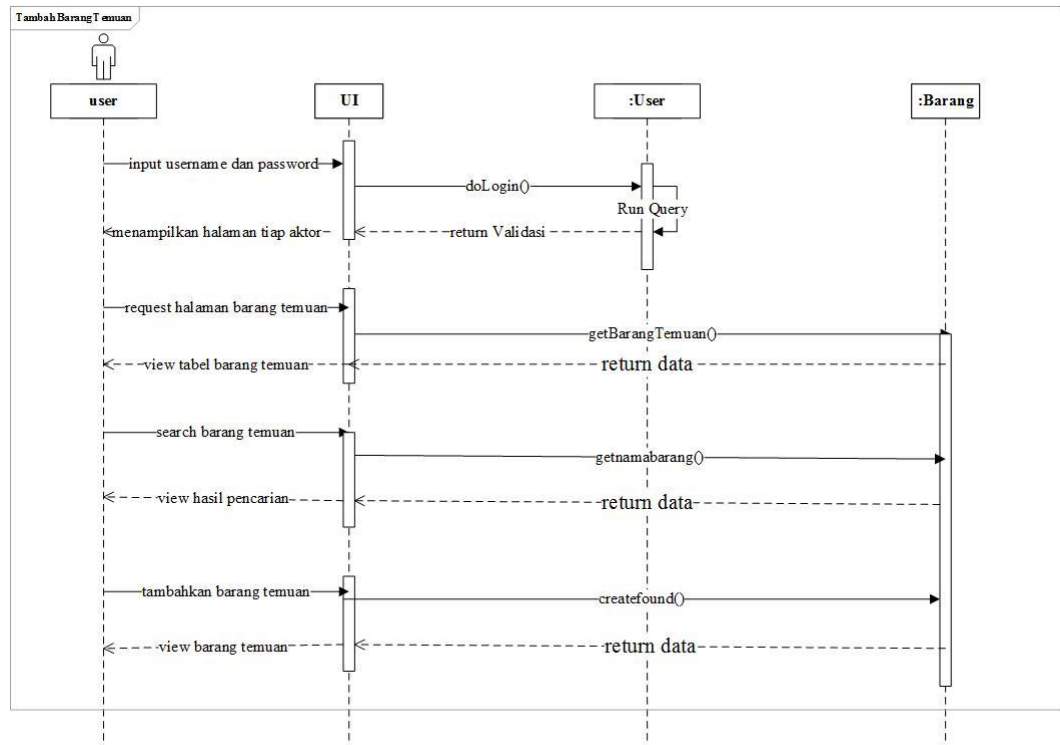
Gambar 3.5 Sequence Diagram Lihat Barang Temuan

Gambar 3.5 menjelaskan kelas apa saja yang terlibat serta pesan apa saja yang terjadi ketika *user* ingin melihat data barang temuan yang disediakan. Ketika *user* menekan menu yang diinginkan maka UI aplikasi akan memberikan menu yang tersedia. Seperti Gambar 3.4 menerangkan proses *request* halaman barang temuan untuk melihat data barang temuan yang ada, setelah itu *user* menekan menu detail barang untuk melihat detail barang yang temuan yang diinginkan.



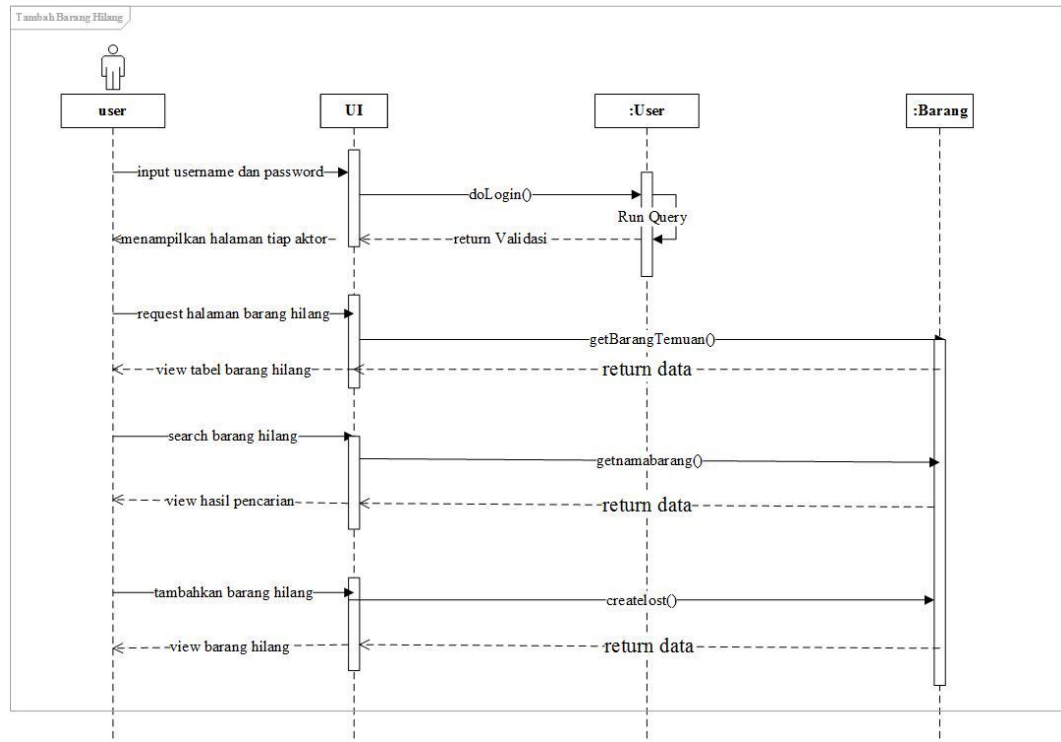
Gambar 3. 6 Sequence Diagram Lihat dan Edit Barang Hilang

Gambar 3.6 menjelaskan proses lihat dan edit data barang hilang yang telah diinputkan oleh *user*. Setelah *user* melakukan *login*, *user* dapat melihat data barang hilang yang telah diinputkan oleh *user* dengan menggunakan fungsi `getBarangHilang()` pada kelas barang, setelah itu *user* dapat melakukan edit masing-masing barang hilang apabila pemilik barang meminta untuk mengubah data barang atau data pemilik yang telah diinput. Dalam proses edit data sistem menggunakan fungsi `update()` pada kelas barang untuk mengedit dan melakukan *update* data.



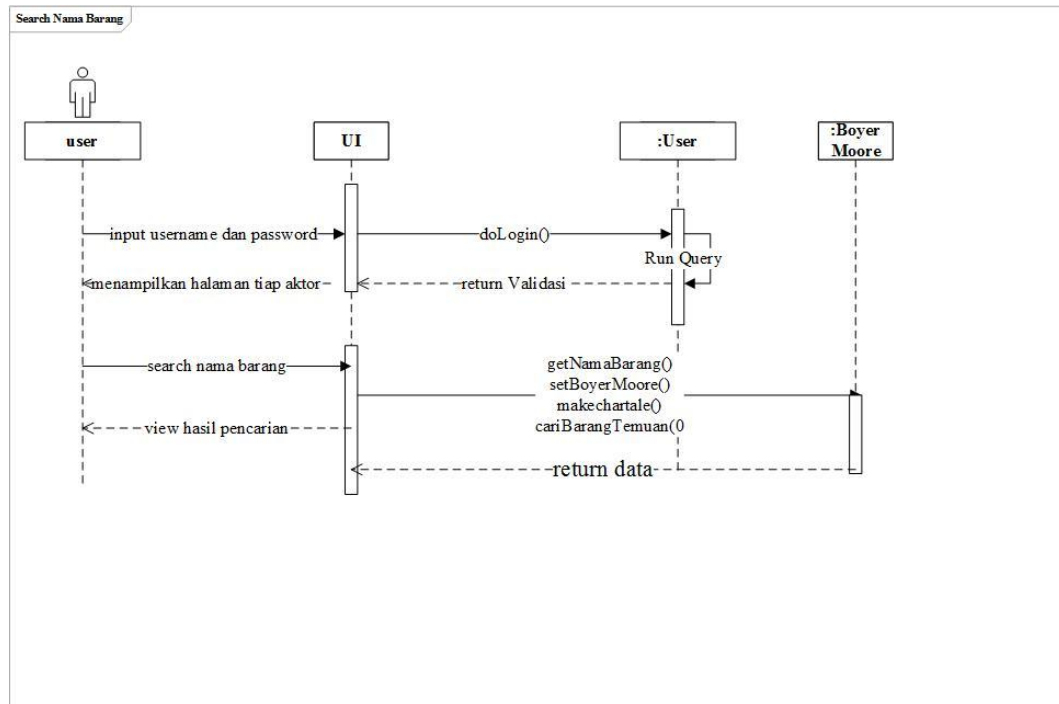
Gambar 3. 7 Sequence Diagram Tambah Barang Temuan

Gambar 3.7 menjelaskan kelas apa saja yang terlibat serta pesan apa saja yang terjadi ketika *user* melakukan penambahan data pengguna aplikasi. Ketika *user* membuka menu tambah barang temuan, maka sistem akan menampilkan formulir untuk mengisi data lengkap barang temuan dan data pelapor. Setelah *user* melakukan input data secara lengkap, maka sistem akan memanggil fungsi `createfound()` pada kelas barang dan menyimpan data yang akan dimasukkan ke dalam *database*. Kemudian memberikan pemberitahuan dengan menampilkan notifikasi pada tampilan yang akan langsung me-*refresh* halaman barang temuan.



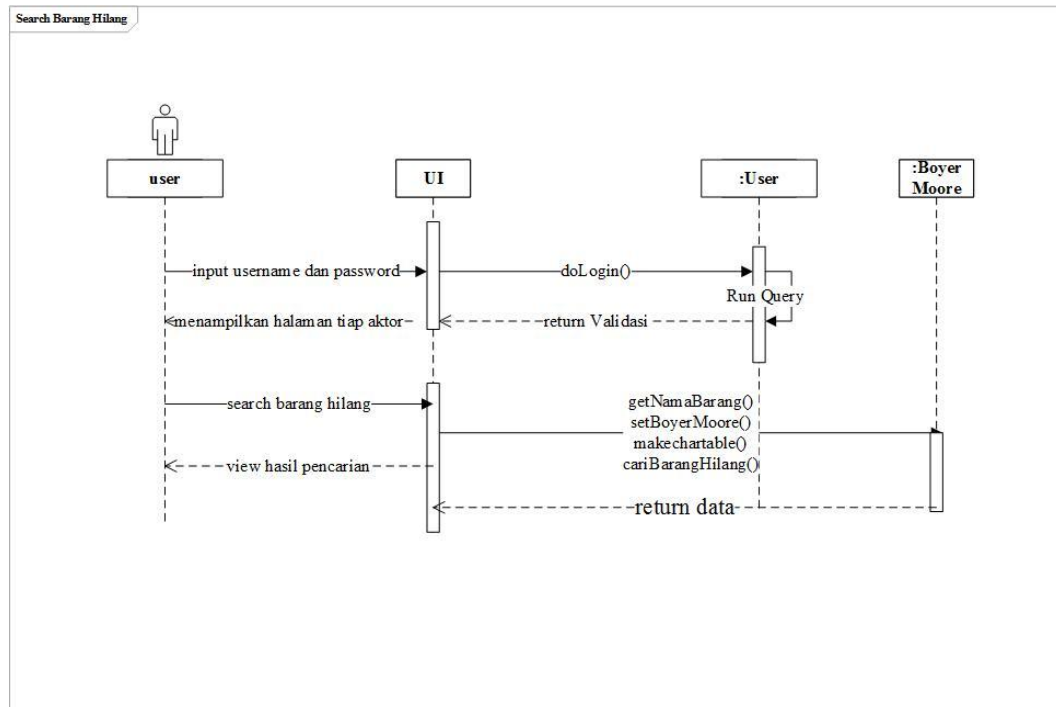
Gambar 3. 8 Sequence Diagram Tambah Barang Hilang

Gambar 3.8 menjelaskan kelas apa saja yang terlibat serta pesan apa saja yang terjadi ketika *user* melakukan penambahan data pengguna aplikasi. Ketika *user* membuka menu tambah barang hilang, maka sistem akan menampilkan formulir untuk mengisi data lengkap barang hilang dan data pelapor. Setelah *user* melakukan input data secara lengkap, maka sistem akan memanggil fungsi `createlost()` pada kelas barang dan menyimpan data yang akan dimasukkan ke dalam *database*. Kemudian memberikan pemberitahuan dengan menampilkan notifikasi pada tampilan yang akan langsung me-*refresh* halaman barang hilang.



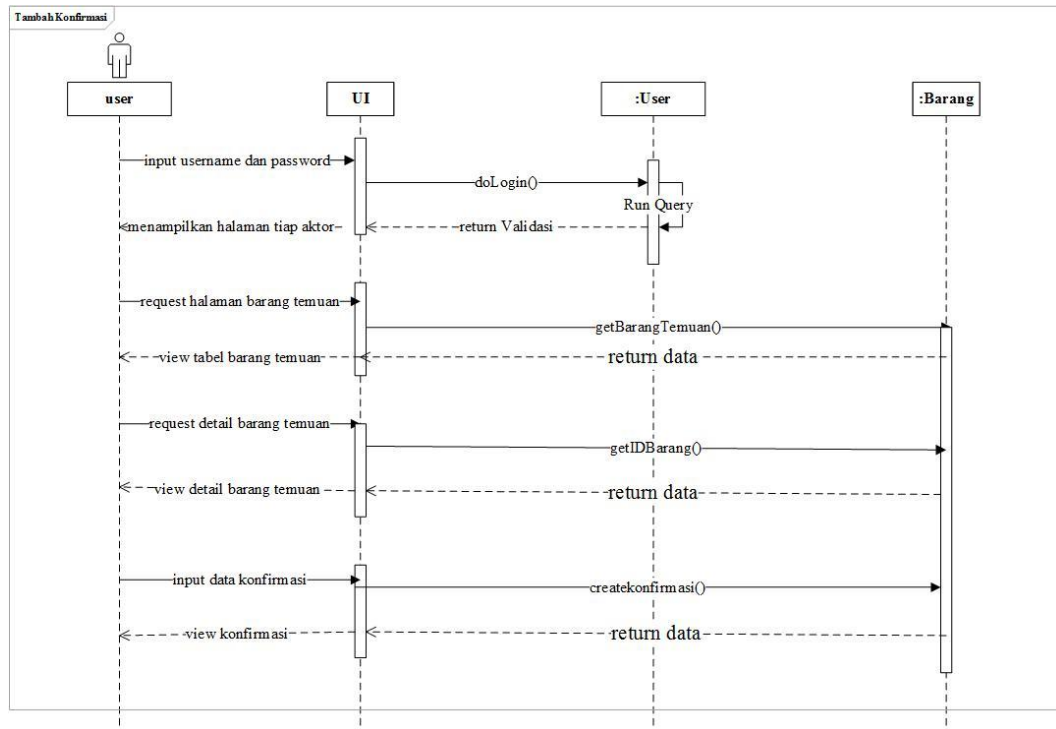
Gambar 3. 9 Sequence Diagram Search Barang Hilang

Gambar 3.9 menjelaskan kelas apa saja yang digunakan dalam proses pencarian barang temuan, ketika *user* memasukkan kata berbentuk *string* ke dalam kolom *search* maka sistem akan memanggil fungsi *getNamaBarang()* kemudian memanggil fungsi *setBoyerMoore()* untuk melakukan pencarian dengan *pattern* yang sesuai, dari hasil fungsi *setBoyerMoore()* maka akan dipanggil fungsi *makechartale()* yang akan mengeksekusikan hasil dari fungsi *setBoyerMoore()* dan dicocokkan kembali dengan data yang ada di *database* dengan fungsi *cariBarangTemuan()*.



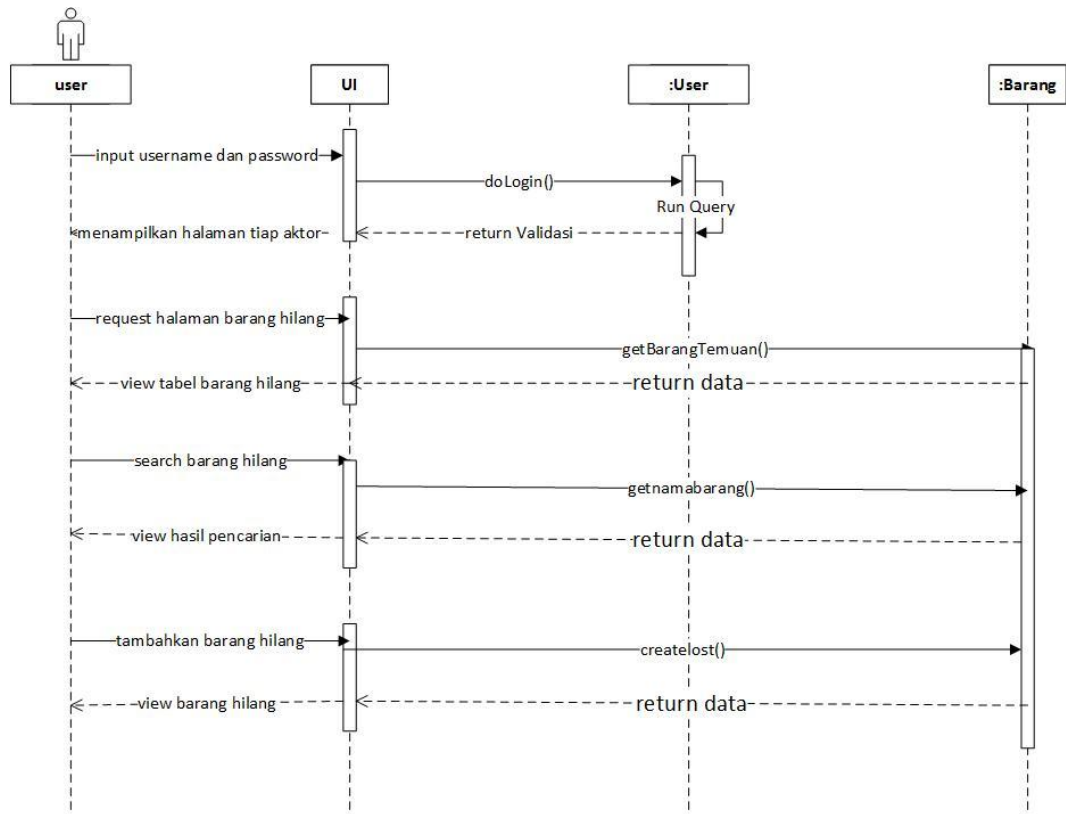
Gambar 3. 10 Sequence Diagram Search Barang Hilang

Gambar 3.10 menjelaskan kelas apa saja yang digunakan dalam proses pencarian barang hilang, ketika *user* memasukkan kata berbentuk *string* ke dalam kolom *search* maka sistem akan memanggil fungsi `getNamaBarang()` kemudian memanggil fungsi `setBoyerMoore()` untuk melakukan pencarian dengan *pattern* yang sesuai, dari hasil fungsi `setBoyerMoore()` maka akan dipanggil fungsi `makechartable()` yang akan mengeksekusikan hasil dari fungsi `setBoyerMoore()` dan dicocokkan kembali dengan data yang ada di *database* dengan fungsi `cariBarangHilang()`.



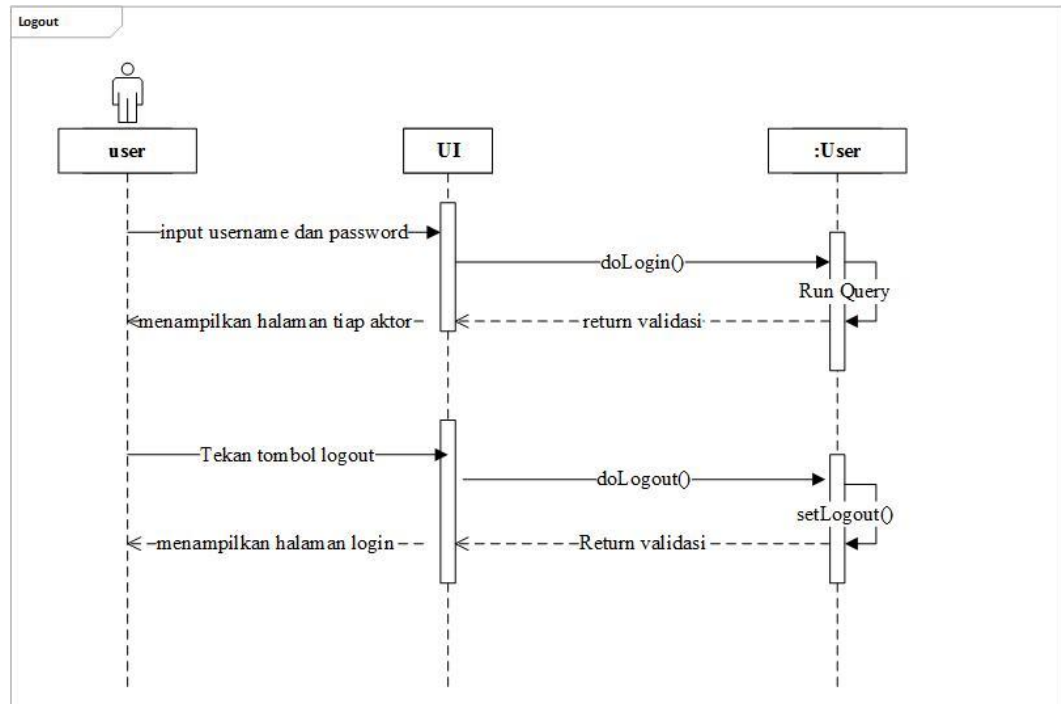
Gambar 3. 11 Sequence Diagram Tambah Konfirmasi Barang

Gambar 3.11 menjelaskan kelas apa saja yang terlibat serta pesan apa saja yang terjadi ketika *user* melakukan penambahan data konfirmasi barang . Ketika *user* membuka menu konfirmasi barang, maka sistem akan menampilkan formulir untuk mengisi data lengkap konfirmasi barang. Setelah *user* melakukan input data secara lengkap, maka sistem akan memanggil fungsi `createkonfirmasi()` pada kelas barang dan menyimpan data yang akan dimasukkan ke dalam *database*. Kemudian memberikan pemberitahuan dengan menampilkan notifikasi pada tampilan yang akan langsung me-*refresh* halaman utama. Pada fitur konfirmasi ini, *user* akan meminta pemilik atau pelapor untuk mengecek *e-mail* karena pemberitahuan telah mengambil barang akan dikirimkan via *e-mail*.



Gambar 3. 12 Sequence Diagram Lihat Konfirmasi Barang

Gambar 3.12 menjelaskan kelas apa saja yang terlibat serta pesan apa saja yang terjadi ketika *user* melihat data konfirmasi barang. Ketika *user* membuka menu detail konfirmasi, maka sistem akan menampilkan data konfirmasi barang yang telah dipilih. Data yang telah dipilih ini memanggil fungsi `getIDKonfirmasi()` yang berisi data konfirmasi yang telah terhubung.

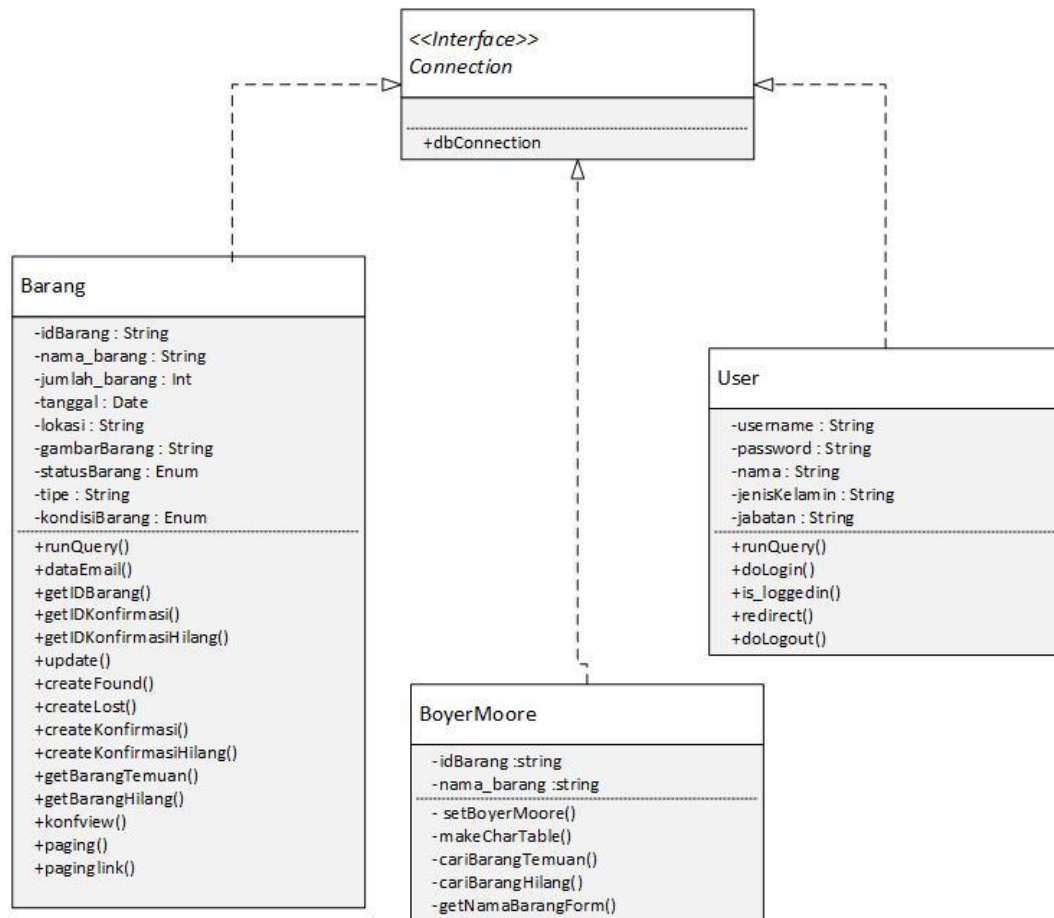


Gambar 3. 13 Sequence Diagram Logout

Gambar 3.13 menjelaskan aktivitas *logout user*. Ketika perintah *logout* dilakukan, maka sistem akan memanggil fungsi `doLogout()` pada kelas *User* dan melakukan perintah *destroy session* pada fungsi `setLogout()` pada kelas *User*, kemudian mengirimkan hasil validasi ke tampilan untuk diberitahukan kepada pengguna aplikasi dan kembali ke halaman *login* aplikasi.

4. Class Diagram

Class diagram menggambarkan kelas yang dibuat dengan hubungannya terhadap kelas lainnya. Penentuan kelas dikelompokkan berdasarkan kemiripan *behavior*. Berikut adalah rancangan *class diagram* dari aplikasi LF-Pro Universitas Bakrie yang dibuat.



Gambar 3. 14 Class Diagram Aplikasi LF-Pro

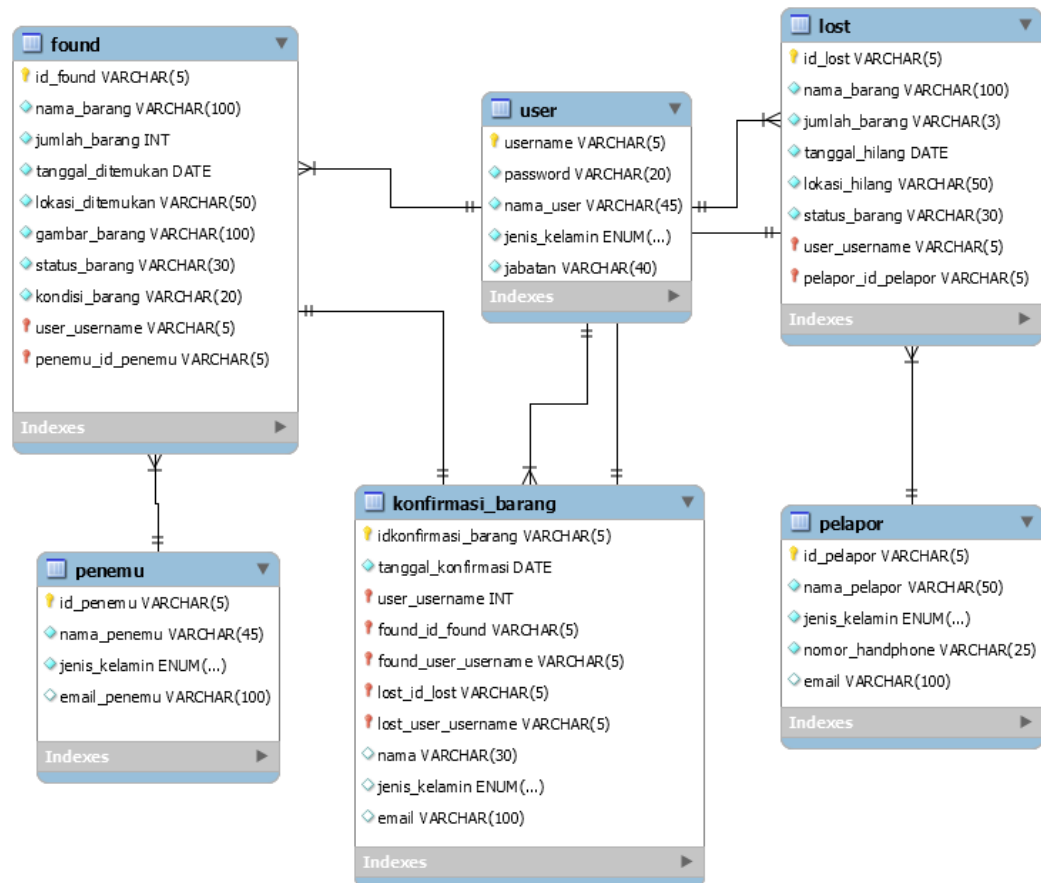
Pada Gambar 3.14 menjelaskan bahwa terdapat 4 kelas yang akan digunakan pada pengembangan aplikasi LF-Pro Universitas Bakrie, antara lain:

- *Connection*, merupakan *class interface* yang akan menghubungkan koneksi aplikasi ke *database*. Kelas ini hanya memiliki metode yang bertugas untuk menghubungkan koneksi.
- *User*, merupakan kelas yang mengelola proses data dari pengguna aplikasi. Kelas ini akan mengelola semua tugas yang berhubungan dengan pengguna aplikasi.
- *Barang*, merupakan kelas yang mengelola proses data yang berhubungan dengan barang temuan dan barang hilang. Kelas ini dikelompokkan berdasarkan *behavior* yang dimiliki yaitu bertugas mengelola data barang hilang dan barang temuan dalam aplikasi.

- Boyer-Moore, merupakan kelas yang menjalankan fungsi-fungsi dari algoritma *boyer-moore*. Kelas ini dikelompokkan berdasarkan *behavior* yang dimiliki yaitu bertugas melakukan pencarian data barang hilang dan barang temuan.

5. Database design

Perancangan *database* adalah untuk menentukan isi dan pengaturan data yang dibutuhkan dalam suatu perancangan sistem. Berikut adalah rancangan *database* untuk aplikasi LF-Pro Universitas Bakrie:



Gambar 3. 15 Data Model LF-Pro

Gambar 3.15 menggambarkan data model aplikasi LF-Pro yang terdiri dari enam tabel yaitu tabel found, user, lost, penemu, konfirmasi_barang dan pelapor, dimana masing-masing tabel memiliki relasi satu sama lain.

3.1.4 Testing

Proses pengujian dilakukan dengan MPE yang akan menguji efektivitas dari Algoritma *Boyer-Moore*. Adapun tahapan analisis yang akan dilakukan adalah:

a. Menentukan alternatif

Analisa ini menggunakan alternatif Algoritma *Brute Force* sebagai pembanding dari Algoritma *Boyer-Moore*.

b. Menentukan kriteria

Dalam menentukan kriteria yang akan dipakai, dapat dijelaskan pada Tabel 3.10

Tabel 3. 10 Penentuan Kriteria (Januardi, 2013)

Kriteria	Keterangan
Jumlah Iterasi Algoritma	Perhitungan jumlah iterasi/perulangan (<i>Looping</i>) yang terjadi pada saat algoritma melakukan usaha pencocokan <i>string</i>
Jumlah Huruf Pada <i>pattern</i>	Jumlah huruf yang dicocokkan oleh algoritma

a. Menentukan bobot kriteria

Penentuan bobot merupakan salah satu komponen paling penting yang berpengaruh pada hasil analisa.

b. Pemberian nilai pada setiap kriteria

Tahap ini adalah sebuah tahap dimana pada setiap kriteria yang telah terbentuk diberi nilai.

c. Menghitung skor

Setelah nilai pada setiap kriteria yang akan dimasukkan, maka tahapan selanjutnya yang akan dilakukan adalah melakukan perhitungan dengan rumus MPE.

d. Menentukan prioritas keputusan

Pada prioritas keputusan akan terlihat total nilai dari alternatif terendah yang memperoleh nilai pertama, karena semakin tinggi nilai total yang telah diperoleh, maka semakin tinggi pula jumlah usaha yang dilakukan oleh algoritma tersebut.

3.1.5 Implementasi

Pada tahap terakhir dilakukan pengimplementasian sistem informasi dan melakukan pengujian akhir sistem

3.3 Jenis Penelitian

Jenis penelitian yang dilakukan adalah implementasi algoritma *Boyer-Moore* yang akan diterapkan pada aplikasi sistem LF-Pro di Universitas Bakrie berbasis *website*. Rancang bangun yang dilakukan diawali dengan identifikasi kebutuhan dan batasan kebutuhan pengguna, dalam hal ini akan dilakukan pengamatan, wawancara dengan narasumber dan studi literatur.

3.4 Objek Penelitian

Objek penelitian yang dilakukan adalah aplikasi LF-Pro yang akan diterapkan di Universitas Bakrie. Aplikasi ini berfungsi untuk memudahkan bagian *security* dalam memproses berita kehilangan dan pencarian hasil barang temuan yang telah disimpan oleh bagian *security*. Aplikasi ini berbasis *web* dengan menerapkan algoritma *Boyer-Moore* pada proses pencarian *string*.

3.5 Metode Pengumpulan Data

Pengumpulan data dan analisa data dalam proses penelitian ini dilakukan untuk melengkapi metode penelitian. Adapun pengumpulan data yang dilakukan adalah sebagai berikut.

1. Pengamatan

Pada proses pengamatan, dilakukan sebagai tahap awal dalam penentuan penelitian yang akan dilakukan. Dalam hal ini melihat proses kerja secara langsung dan membuat hipotesa masalah-masalah yang terjadi pada pengalaman kehilangan barang dan penemuan barang di Universitas

Bakrie. Berikut ini merupakan definisi lokasi penelitian yang berlangsung:

Nama Institusi : Universitas Bakrie Jakarta
Bidang : Lembaga Pendidikan
Alamat : Jl. HR.Rasuna Said Kav C-22
Gedung Pasar Festival Lt GF/22, Setiabudi,
Jakarta Selatan
Telp : 021 5276543
Waktu Wawancara : 12 Februari 2016

2. Wawancara

Wawancara dilakukan kepada koordinator *security* Universitas Bakrie yang menangani kehilangan barang maupun temuan barang. Berikut ini merupakan rincian wawancara:

Narasumber : Universitas Bakrie Jakarta
Profesi : Koordinator *Security* Universitas Bakrie
Alamat : Jl. HR.Rasuna Said Kav C-22
Gedung Pasar Festival Lt GF/22, Setiabudi,
Jakarta Selatan
Telp : 021 5276543
Waktu Wawancara : 12 Februari 2016

Hasil dari wawancara serta daftar pernyataan terlampir pada lampiran.

3. Studi Literatur

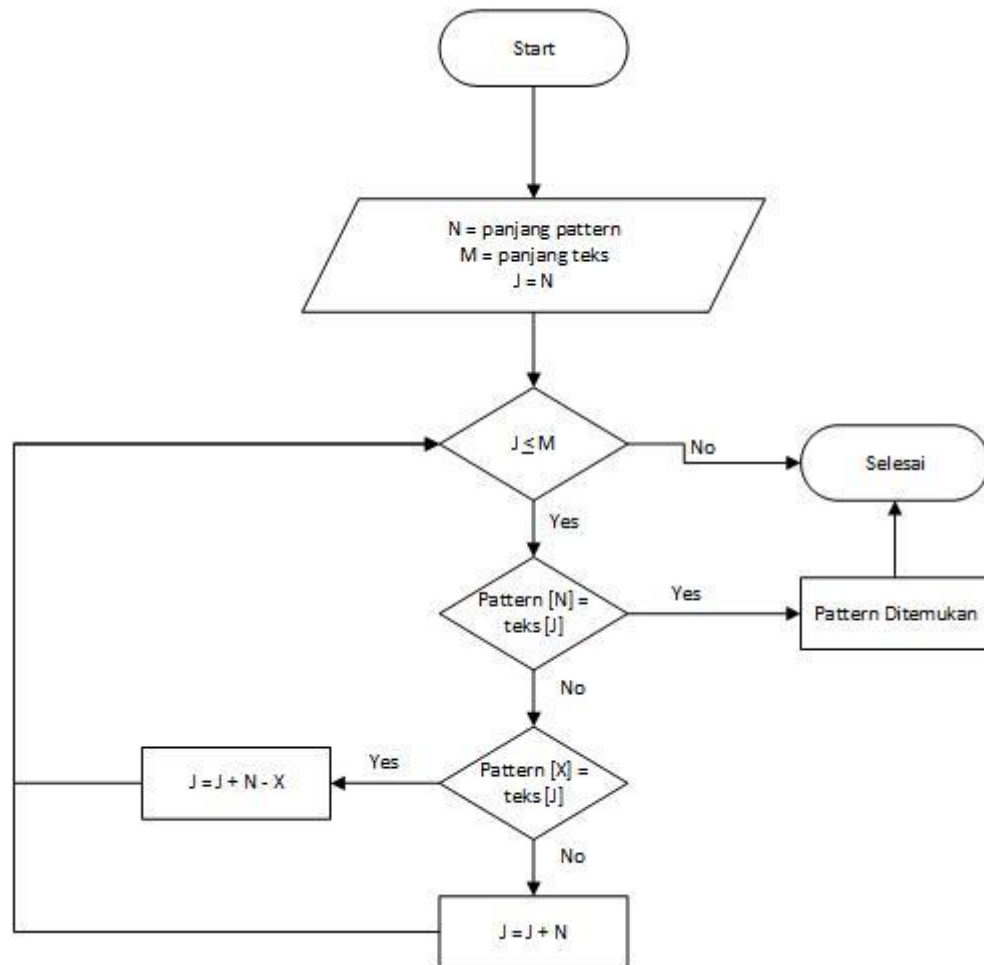
Dalam melakukan penelitian ini, dilakukan kajian pustaka dengan mempelajari beberapa buku teks, jurnal penelitian, *e-book*, tugas akhir serta materi-materi di internet yang mendukung dalam proses penelitian. Hasil dari tahap ini merupakan sebuah *literature review* yang dapat digunakan untuk menentukan landasan teori yang tepat untuk penelitian. *Literature Review* juga digunakan untuk pemilihan metode sistem pencarian *string*, model siklus pengembangan perangkat lunak, bahasa pemrograman, dan pengujian aplikasi dalam penelitian.

3.6 Implementasi Algoritma *Boyer-Moore*

Algoritma *Boyer-Moore* dianggap sebagai sebuah algoritma pencocokan *string* (*string matching*) yang paling efisien pada kebanyakan aplikasi, sebagai contoh pada *text editor* dan *command substitution*. Hal ini dikarenakan algoritma ini bekerja sangat cepat pada kasus dimana alfabet berukuran sedang dan *pattern* yang akan dicari relatif panjang.

Algoritma *Boyer-Moore* menelusuri karakter-karakter pada *pattern* dari kanan ke kiri, dimulai dari karakter yang berada pada posisi paling kanan. Selama proses pencocokan antara *pattern* P dengan *text* T, ketidakcocokan antara karakter *text* $T[i] = c$ dengan karakter *pattern* $P[j]$ yang bersesuaian ditangani dengan skenario sebagai berikut.

Jika c tidak terdapat pada P, maka geser P secara keseluruhan melewati $T[i]$.
Atau jika tidak – c terdapat dalam P – geser P sampai kemunculan karakter c pada P dapat bersesuaian dengan $T[i]$.



Gambar 3. 16 Flowchart Algoritma Boyer-Moore (Pratiwi, Syarif, & Wibowo, 2012)

Teknik ini mampu menghindari perbandingan-perbandingan yang tidak dibutuhkan dengan menggeser *pattern* relatif terhadap *text*.

Setiap karakter *c* pada alfabet memiliki nilai kemunculan (*last(c)*) sebagai berikut (Kent, t.thn.).

$$Last(c) = \begin{cases} \text{Index (posisi) kemunculan} \\ \text{terakhir karakter } c \text{ pada } pattern \\ P & \text{Jika } c \text{ ada dalam } P \\ -1 & \text{Jika } c \text{ tidak ada dalam } P \end{cases}$$

Nilai kemunculan ini menentukan seberapa jauh pergeseran *pattern* *P* dapat dilakukan jika karakter *C* di dalam *text* tidak cocok dengan *pattern*. Berikut ini adalah contoh dari nilai kemunculan suatu karakter.

T:	0	1	2	3	4	5	6	7	8	9	10
	T	G	T	C	A	A	A	G	T	T	C
P:	0	1	2	3	4	5					
	T	G	T	C	A	G					

Maka, didapatkan nilai kemunculan sebagai berikut.

c	A	C	T	G	S
<i>Last(c)</i>	4	3	2	5	-1

Implementasi algoritma *Boyer-Moore* yang diterapkan pada tugas akhir ini, didasarkan pada *pseudocode* berikut.

Input : Text dengan n karakter dan *Pattern* dengan m karakter

Output : Index dari *substring* awal dari T yang cocok dengan P

```

for  $x \in \Sigma$ 
     $last[x] \leftarrow -1$ 
for  $y \leftarrow m$  downto 1
     $last[P[y]] \leftarrow y$ 

 $i \leftarrow m-1$ 
 $j \leftarrow m-1$ 
Repeat
    If  $P[j] = T[i]$  then
        If  $j=0$  then
            return  $i$            // we have a match
        else
             $i \leftarrow i - 1$ 
             $j \leftarrow j - 1$ 
    else
         $i \leftarrow i + m - \text{Min}(j, 1 + last[T[i]])$ 
         $j \leftarrow m - 1$ 

```

```

until i > n - 1
Return "no match"

```

Komputasi dari fungsi *last* itu sendiri membutuhkan waktu $O(m + |\Sigma|)$. Sedangkan pada kasus terbaik dari algoritma tersebut, untuk teks dengan panjang n dan pola yang akan dicari dengan panjang m , dibutuhkan waktu n/m . Hal ini dikarenakan pada kasus terbaik, hanya satu karakter di dalam m yang perlu di cek. Hal ini juga menjelaskan bahwa semakin panjang pola yang akan dicari, maka akan semakin cepat algoritma tersebut menemukannya.

Pada kasus terburuk algoritma ini membutuhkan waktu $m \cdot n$ untuk dapat menemukan hasil yang cocok. Kasus buruk ini terjadi ketika *string* yang dicari terdiri dari pengulangan sebuah karakter, dan *string* yang menjadi target terdiri dari $m-1$ pengulangan dari karakter itu yang didahului dengan suatu karakter yang berbeda. Pada skenario ini, harus dilakukan pengecekan sebanyak $n-m+1$, dimana setiap pengecekan membutuhkan m perbandingan. Oleh karena itu, *running time* algoritma *Boyer-Moore* pada kasus terburuk adalah $O(nm + |\Sigma|)$.

BAB IV

IMPLEMENTASI DAN PENGUJIAN

Bab ini membahas mengenai implementasi fitur search dengan *auto-complete* pada *search* barang aplikasi LF-Pro menggunakan algoritma *Boyer-Moore*. Perancangan dan implementasi algoritma *Boyer-Moore* didasarkan pada analisis yang telah dilakukan pada bab sebelumnya. Bab ini juga akan membahas mengenai pengujian algoritma serta analisis dari hasil penerapan algoritma *Boyer-Moore* pada aplikasi LF-Pro Universitas Bakrie.

4.1 Implementasi Sistem

Implementasi sistem merupakan kumpulan dari elemen-elemen yang telah dirancang dalam bentuk pemrograman untuk menghasilkan suatu tujuan yang dibuat berdasarkan kebutuhan yang telah dibuat. Berikut adalah spesifikasi *hardware* dan *software* yang digunakan dalam tahapan implementasi.

1. Informasi *Hardware*

Informasi *hardware* yang digunakan dalam pengembangan aplikasi ini adalah sebagai berikut:

<i>Nama device</i>	: Laptop ASUS n46u
<i>Operating system</i>	: Windows 7 Home Premium
<i>Processor</i>	: Intel(R) Core (TM) i5-3210M CPU @ 2,5GHz 2.50 GHz
<i>Memory</i>	: 4.00 GB RAM

2. Informasi *Software*

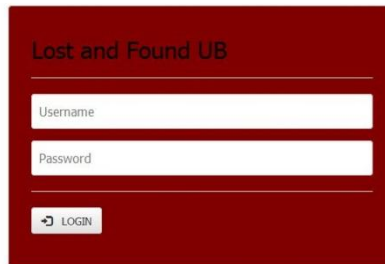
Informasi *Software* untuk pengembangan aplikasi adalah sebagai berikut:

- XAMPP version 3.2.2 sebagai *web server*, *database server*, dan *application server*
- Mozilla Firefox version 47.0.1 sebagai *web browser*

- Sublime Text *Copyright* 2006-2014 Sublime HQ Pty Ltd untuk membangun aplikasi

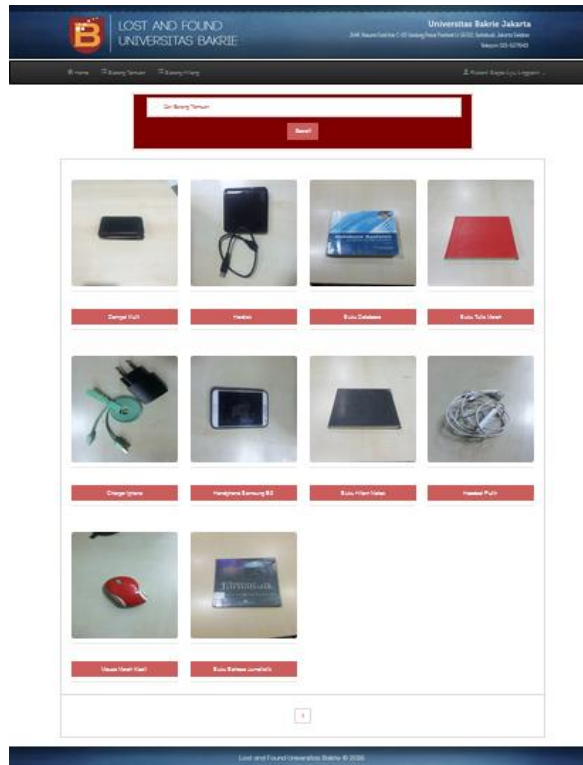
4.2 Implementasi Perancangan Antarmuka

Berdasarkan hasil perancangan *user interface* pada tahap sebelumnya, implementasi dari *user interface* dilakukan dengan menggunakan *front-end framework Hyper Text Markup Language* (HTML) *Cascading Style Sheet* (CSS) yaitu *Bootstrap*. Berikut tampilan hasil rancangan yang telah dibuat:




Gambar 1 Halaman *Login*

Gambar 4.1 di atas menggambarkan halaman awal bagi *user* untuk dapat mengakses aplikasi LF-Pro. *User* harus memasukkan “nomor induk” dan “*password*” dengan benar sehingga *user* dapat akses masuk ke dalam aplikasi.



Gambar 2 Halaman Awal Barang Temuan

User dapat melihat semua data barang yang telah di-submit dengan keterangan barang temuan. Gambar 4.2 menampilkan data barang temuan dalam bentuk Tabel dan Gambar sehingga dapat memudahkan *user* untuk melihat barang yang akan dicari.

<div>  <div> LOST AND FOUND UNIVERSITAS BAKRIE </div> <div> Universitas Bakrie Jakarta JHR Rasuna Said Kav C-22 Gedung Pasca Festival Lt G/22, Setiabudi, Jakarta Selatan Telepon 021-527643 </div> </div>												
<div> Home Barang Temuan Barang Hilang </div> <div> Ristanti Septa Ayu Anggrini </div>												
<div> <input type="text" value="Cari Barang Hilang"/> <div> Search + Barang Hilang </div> </div>												
ID	Nama Barang	Jumlah Barang	Tanggal Hilang	Lokasi Hilang	Status Barang	Kondisi Barang	Tipe Barang	Nama Pemilik	Jenis Kelamin	No Hp	Email Pemilik	Actions
73642	Kartu Ujian Semester Ganjil	1	2016-07-30	Ruang 6B	Hilang	Baik	Lain-Lain	Rosdiana	Perempuan	085672452411	rosdianacantik@yahoo.com	
25064	Tumbler	1	2016-08-08	Ruang 16	Hilang	Baik	Lain-Lain	Ristanti	Perempuan	085330000811	ristantiseptas@gmail.com	
16785	Akimaster Ukuran II	1	2016-07-27	Ruang 1	Hilang	Baik	Lain-Lain	Desna	Perempuan	08533723211	desna30@hotmail.com	
81679	Kacamata Wama Biru	1	2016-08-01	Ruang Beca Perpustakaan	Hilang	Baik	Lain-Lain	Anindya Naya	Perempuan	0897261253	aninn@yahoo.co.id	
59410	Kamera Xioayi Putih	1	2016-07-17	Ruang 10	Hilang	Baik	Elektronik	Damar Alam Raja	Laki-Laki	086541231420	damareja@gmail.com	
34780	Pajung Wama Wami	1	2016-08-09	SL Dalam	Hilang	Baik	Lain-Lain	Dini Novita	Perempuan	08533271610	dininovita12@yahoo.com	
40921	Tempat Pensil Batik	1	2016-08-09	Ruang 14	Hilang	Baik	Lain-Lain	Donita Sari	Perempuan	08761524611	donitasari@yahoo.com	
07351	Buku Binder Kuning	1	2016-07-16	Ruang 11	Hilang	Baik	Buku	Lain-lain	Perempuan	087635251411	ainasliyo@gmail.com	
63210	Flashdisk Biru	1	2016-08-09	Kantor Admisi UB	Hilang	Baik	Elektronik	Andika	Laki-Laki	08764325421	andika.pratama@yahoo.com	
19602	Sepatu Nike Hijau	1	2016-07-14	SL Dalam	Hilang	Baik	Lain-Lain	Santo	Laki-Laki	089726352412	santoo@gmail.com	
<div> 1 </div>												
Lost and Found Universitas Bakrie © 2016												

Gambar 3 Tampilan Menu Barang Hilang

User dapat melihat semua data barang hilang yang telah di-submit dengan keterangan barang temuan. Gambar 4.3 menampilkan data barang temuan dalam bentuk Tabel sehingga dapat memudahkan user untuk melihat barang yang akan dicari.



A. Data Barang Hilang

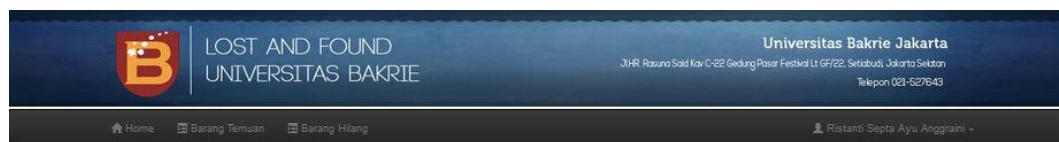
Nama Barang	<input type="text"/>
Jumlah Barang	<input type="text"/>
Tanggal Hilang	<input type="text"/>
Lokasi Hilang	<input type="text"/>
Kondisi Barang	<input type="radio"/> Baik <input type="radio"/> Rusak
Tipe Barang	<input type="text"/>

B. Data Pemilik

Nama Pemilik	<input type="text"/>
Jenis Kelamin	<input type="radio"/> Laki-Laki <input type="radio"/> Perempuan
Email	<input type="text"/>
No. Handphone	<input type="text"/>
<input type="button" value="Simpan Data"/> <input type="button" value="HOME"/>	

Gambar 4 Tambahkan Data Barang Temuan atau Hilang

Gambar 4.4 menampilkan halaman *insert* data barang hilang maupun temuan ke dalam *database*.



Nama Pemilik	<input type="text"/>
Jenis Kelamin	<input type="radio"/> Laki-Laki <input type="radio"/> Perempuan
Tanggal konfirmasi	<input type="text"/>
Email	<input type="text"/>
<input type="button" value="KONFIRMASI"/> <input type="button" value="HOME"/>	

Gambar 5 Formulir Konfirmasi Barang


Gambar 4.5 menampilkan formulir konfirmasi barang yang akan diisi sesuai data *user* yang benar.

**LOST AND FOUND
UNIVERSITAS BAKRIE**

Universitas Bakrie Jakarta
Jl. R. Rasuna Said Kav C-02 Gedung Pasar Festival Lt. G/02, Setiabudi, Jakarta Selatan
Telepon 021-527843

Home Barang Temuan Barang Hilang Ristand Septa Ayu Anggrini

A. Data Barang

Nama Barang	Dompet Kulit
Jumlah Barang	1
Tanggal Ditemukan	2016-08-02
Lokasi Ditemukan	Sl. Luar
Kondisi Barang	Belk
Tipe Barang	Lain-Lain
Gambar Barang	
Status Barang	Disimpan

B. Data Penemu

Nama Penemu	Ridho
Jenis Kelamin	Laki-Laki
Email	ridhogillang@gmail.com

Konfirmasi Barang BACK

Gambar 6 Tampilan *Detail* Barang Temuan

Gambar 4.6 menampilkan data barang yang belum di ambil oleh pemilik, perbedaannya adalah dengan adanya status barang pada *detail* barang tersebut. Status barang yang belum diambil diberikan “Disimpan” sedangkan yang telah diambil adalah “Sudah Diambil”.



Universitas Bakrie Jakarta
Jl. HR. Rasuna Said Kav. C-22 Gedung Pasar Festival Lt. GF/22, Setiabudi, Jakarta Selatan
Telepon 021-527643

Home | Barang Temuan | Barang Hilang | Ristanti Septa Ayu Anggraini

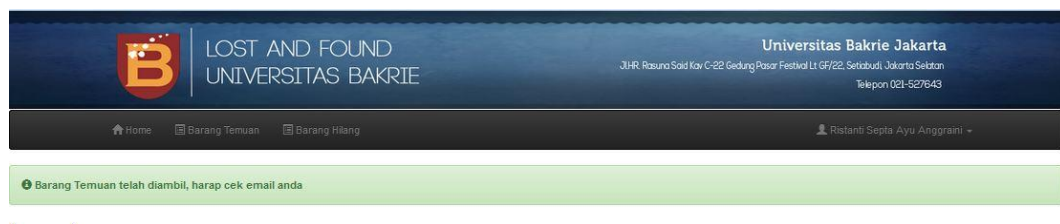
Data Pemilik

Nama Pemilik	tina
Jenis Kelamin	laki-laki
Tanggal Konfirmasi	2018-08-02
Email	ristantiseptaa@gmail.com

[← BACK](#)

Gambar 7 Tampilan Data Pemilik

Gambar 4.7 menampilkan data pemilik yang telah mengambil barang temuan.



Gambar 8 Notifikasi Barang Telah Diambil

Gambar 4.8 menampilkan hasil dari konfirmasi barang yang telah diambil dan disarankan pemilik mengecek *e-mail* yang telah didaftarkan untuk pemberitahuan lebih lengkap.



Gambar 9 Pesan Konfirmasi Barang Hilang via E-mail

Pada Gambar 4.9 merupakan contoh *e-mail* dari bagian *security* untuk konfirmasi bahwa barang telah ditemukan dan dapat diambil di bagian *security* dengan batas waktu yang telah ditentukan.



Gambar 10 Konfirmasi Pengambilan Barang via E-mail

Pada Gambar 4.10 merupakan contoh *e-mail* dari bagian *security* untuk konfirmasi pengambilan barang.

4.3 Implementasi Data

Data yang digunakan dalam penelitian ini adalah data barang temuan yang diambil sampelnya dari salah satu jenis barang. Kumpulan data tersebut dimasukkan dalam *database* mysql menggunakan perangkat bantuan phpmyadmin. *File* data ini bernama “dbpdo.sql”, *file* ini di-*import* ke dalam *database* “dbpdo”. Data yang digunakan sebagai sampel dalam implementasi *search* ini adalah data dari barang-barang yang sering dilaporkan hilang oleh pihak *security*. Jenis ini dipilih sebagai tes data karena memiliki jumlah informasi terbesar dibandingkan dengan jenis lainnya.

4.4 Implementasi Algoritma *Boyer-Moore* pada Fitur *Auto-complete*

Fitur *auto-complete* digunakan pada pengisian nama barang untuk proses pencarian barang. Dengan adanya fitur *auto-complete*, pengguna tidak perlu memasukkan nama lengkap barang untuk mencari jika barang tersebut memang ada pada basis data yang ada.

Algoritma *Boyer-Moore* berperan sebagai mencari *pattern* (P) yang dicari pada *full-text* (S) nama barang yang ada di Tabel *tb_found* dan *tb_lost*. Dalam pengaplikasiannya, saat pengguna memasukkan minimal satu karakter pada *search* form, program JavaScript akan melakukan panggilan fungsi PHP dengan Asynchronous Javascript and XML (AJAX) seperti yang digambarkan pada Gambar 4.11. Program ini bergantung pada *library* jQuery untuk menghasilkan bentuk *auto-complete* yang menerima daftar *string* yang akan ditampilkan dalam format *JavaScript Object Notation* (JSON).

```
$(function() {
    $("#topic_title").autocomplete({ //Library jQuery autocomplete
        source: "autocomplete.php", //Program PHP yang dipanggil
        minLength: 1, //berapa huruf minimal untuk mulai operasi
        html: true,
        open: function(event, ui) {
            $(".ui-autocomplete").css("z-index", 1000); //CSS
        }
    });
});
```

Gambar 11 Program JavaScript untuk Fitur *Auto-complete*

Program JavaScript pada Gambar 4.11 menggunakan metode GET untuk melakukan *request* ke program PHP pada *server side*. Program PHP `autocomplete.php` akan menjalankan panggilan fungsi untuk menjalankan fungsi algoritma *Boyer-Moore* yang ada pada program `boyermoore.php`. Gambar 4.12 memberikan Gambaran mengenai kode pada `autocomplete.php`

```
<?php
if (!isset($_GET['term'])) {
    die();
}

$keyword = $_GET['term'];
// $data = searchForKeyword($keyword);
$search = new search();
$data = $search->cariTemuan($keyword); //ada di database.php
// $data->array semua barang yang ditemui
echo json_encode($data);

?>
```

Gambar 12 Kode `autocomplete.php`

Program `class.boyermoore.php` bertugas untuk mengambil data nama barang dari *database* dan mengolahnnya dengan algoritma *Boyer-Moore* untuk menemukan *pattern* (P) tertentu sesuai yang diinginkan oleh pengguna. Gambar 4.13 menggambarkan bagaimana program mengakses *database* dan

mengubahnya menjadi *string* yang memiliki pola yang ditunjukkan pada Gambar 4.14.

```
function cariTemuan($key) {
    $barang = new barang();
    $hasil = '';
    $table = 'tb_found';
    $res= $barang->getNamaBarangFrom($table);
    $size = sizeof($res);
    for ($i = 0; $i < $size; $i++) {
        foreach ($res[$i] as $a => $b) {
            $hasil .= ($b . "*");
        }
    }

    $cari = $this->setBoyerMoore($hasil, $key);
    return $cari;
}
```

Gambar 13 Fungsi `cariTemuan()` dalam `class.BoyerMoore.php`

```
Nama barang 1*nama barang 2* nama barang 3*...*
```

Gambar 14 Bentuk *String* yang Akan Diolah

Sebelum melakukan pengolahan dengan *Boyer-Moore*, program PHP pada `class.boyermoore.php` khususnya fungsi `makechartable()` membuat sebuah Tabel dalam bentuk array dari *pattern* (P) dengan pola array(2) { [`'karakter pertama'`] => `int(i)` [`karakter kedua`] => `int(0)` ... }. Gambar 4.15 menunjukkan potongan kode pada fungsi `makechartable()`.


```

function makeCharTable($string) {
    $len = strlen($string);
    $table = array();
    for ($i=0; $i < $len; $i++) {
        $table[strtolower($string[$i])] = $len - $i - 1;
    }
    return $table;
}

```

Gambar 15 Fungsi makechartable ()

Fungsi paling krusial yang berperan sebagai pencari *pattern* (P) pada *string* adalah fungsi Boyer-moore(\$text, \$pattern) yang ditunjukkan pada Lampiran. Selain menggunakan *Boyer-Moore* untuk menentukan dimana indeks *string* saat *pattern* ditemukan, fungsi ini juga menggandakan seluruh nama barang sehingga walaupun *pattern* ditemukan di tengah nama barang, seperti menemukan *pattern* “ge” dalam “Buku Management, nama lengkap dari barang akan diambil dan hasilnya dapat dilihat dalam fitur *auto-complete*.

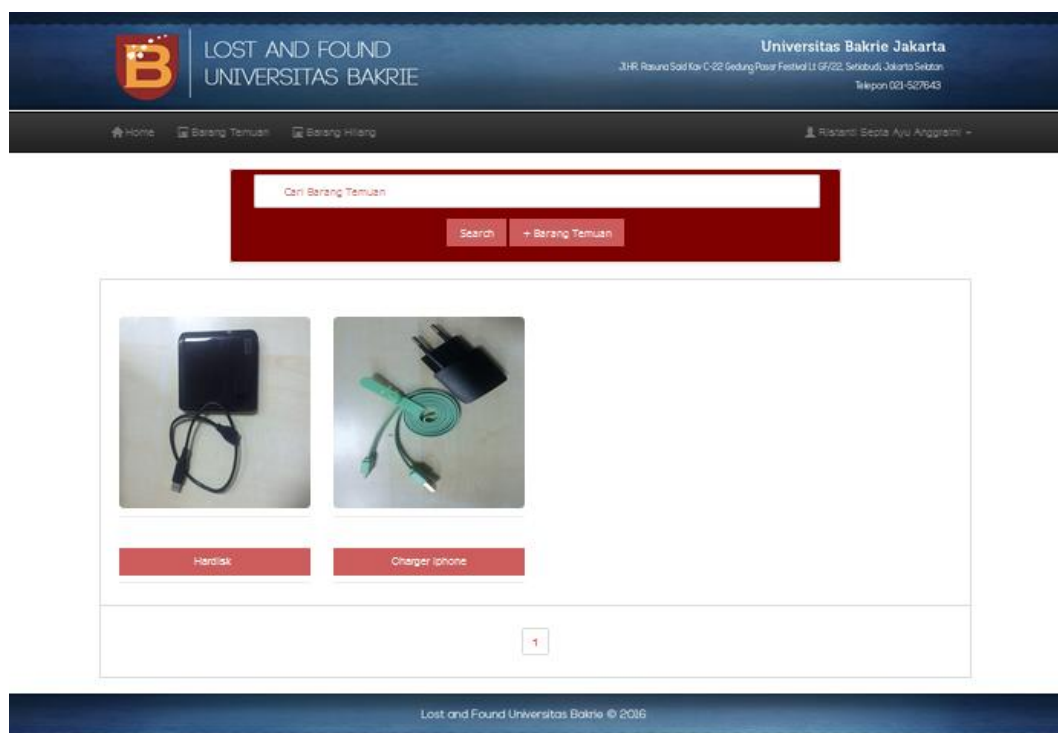
4.5 Hasil Pencarian Berdasarkan Kata Kunci

Bagian ini menjelaskan mengenai hasil pencarian *string* yang dilakukan *user* pada *field search*. Kata kunci yang dimasukkan akan ditampilkan dalam beberapa kondisi. Kondisi pertama adalah hasil pencarian yang terdapat pada *database* dan ditampilkan oleh fitur *auto-complete*. Hasil yang ditampilkan merupakan semua data yang ada di *database* dan merupakan *string* dari yang telah di-*input*-kan. Kondisi kedua adalah hasil pencarian yang tidak terdapat pada *database* dan tidak dapat ditampilkan oleh fitur *auto-complete* karena tidak sesuai dengan *string* yang ada di *database*.



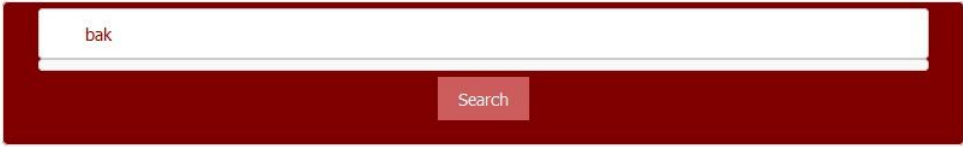
Gambar 16 Search dengan Fitur Auto-Complete Kondisi Pertama

Pada Gambar 4. 16 merupakan hasil pencarian data nama barang dengan menggunakan string “har” dan *auto-complete* dapat menampilkan semua hasil dari string “har” yang ada di *database*.




Gambar 17 Hasil Search Pada Kondisi Pertama

Pada Gambar 4.17 menampilkan hasil dari pencarian kata “har” pada kondisi pertama, terdapat 2 data barang yang tersimpan dengan kata yang mengandung *string* “har”.



Gambar 18 Search dengan Fitur Auto-Complete Kondisi Kedua

Pada Gambar 4. 18 merupakan hasil pencarian data nama barang dengan menggunakan string “bak” dan *auto-complete* tidak dapat menampilkan semua hasil dari *string* “bak” yang dari *database*.



Gambar 19 Hasil dari Pencarian String Kondisi Kedua

Pada Gambar 4.19 menampilkan hasil dari pencarian kata yang mengandung “bak” dan sistem tidak dapat menemukan nama barang yang cocok dengan yang di-*input*-kan pada kolom *search*.

4.6 Pengujian Algoritma

Berdasarkan rencana pengujian algoritma bagian 3.1.5 di atas, pengujian akan dilakukan dengan memakai MPE. Penulis mengadopsi *draft* pengujian mengacu pada penelitian (Mahardika, 2012) yang menerapkan MPE sebagai metode pengujian penentuan ranking. Spesifikasi pengujian yang diterapkan pada algoritma *boyer-moore* dan *brute-force* adalah:

Jenis Data Pencarian	: Karakter huruf a-z 1-9 dan tanda baca
Ukuran data	: Minimal 1 Karakter dan Maksimal 25 karakter
Item pengujian	: 10 sampel data barang temuan
<i>Record</i>	: Data yang ditampilkan pada <i>autocomplete</i> hanya berupa data yang ada pada <i>database</i> , dan tidak bisa digabungkan.
Banyak <i>Pattern</i>	: 4 <i>pattern</i>

Berikut adalah penjelasan pengujian algoritma *Boyer-Moore* berdasarkan MPE :

4.7.1 Menentukan *Pattern* pada Teks

Dalam pengujian kali ini, diambil sepuluh sampel data dari aplikasi LF-Pro untuk dicocokkan dengan pencocokan *string* menggunakan algoritma *Boyer-Moore* dan *Brute-Force*. Dalam proses pencocokannya dibagi menjadi dua komponen yaitu *pattern* dan teks. Penentuan *pattern* dan teks yang akan digunakan untuk menganalisa data diambil dari analisa Tabel 4.1.

Tabel 4. 1 Penentuan *Pattern* dan Teks Setelah Jumlah Hurufnya Disamakan

Proses Ke-	Pattern	Teks Setelah Dipotong	Teks di Database
1	h	h	<i>Handphone</i> Samsung
		h	<i>Headset</i> Putih
		h	Buku Hitam Notes
		h	<i>Charger</i> Iphone
		h	<i>Hardisk</i>
		h	Buku Tulis Merah
		b	<i>Mouse</i> Merah Kecil
		d	Dompot Kulit
		b	Buku Bahasa Jurnalistik
		b	Buku Database
2	ha	ha	<i>Handphone</i> Samsung
		ho	<i>Handphone</i> Samsung
		he	<i>Headset</i> Putih

Proses Ke-	Pattern	Teks Setelah Dipotong	Teks di Database
		hi	Buku Hitam Notes
		ha	<i>Charger Iphone</i>
		ho	<i>Charger Iphone</i>
		ha	<i>Hardisk</i>
		hi	Buku Tulis Merah
		h-	<i>Mouse Merah Kecil</i>
		do	Dompot Kulit
		ha	Buku Bahasa Jurnalistik
		bu	Buku Database
3	har	han	<i>Handphone Samsung</i>
		hon	<i>Handphone Samsung</i>
		hea	<i>Headset Putih</i>
		hit	Buku Hitam Notes
		har	<i>Charger Iphone</i>
		hon	<i>Charger Iphone</i>
		har	<i>Hardisk</i>
		h--	Buku Tulis Merah
		h-k	<i>Mouse Merah Kecil</i>
		dom	Dompot Kulit
		has	Buku Bahasa Jurnalistik
		buk	Buku Database
4	hard	hand	<i>Handphone Samsung</i>
		hone	<i>Handphone Samsung</i>
		head	<i>Headset Putih</i>
		hita	Buku Hitam Notes
		harg	<i>Charger Iphone</i>
		hone	<i>Charger Iphone</i>
		hard	<i>Hardisk</i>
		h---	Buku Tulis Merah
		h-ke	<i>Mouse Merah Kecil</i>
		domp	Dompot Kulit
		hasa	Buku Bahasa Jurnalistik
		buku	Buku Database

4.7.2 Proses Pencarian Algoritma

Setelah menentukan *pattern* dan teks maka proses selanjutnya adalah pencocokan karakter. Dalam proses pengujian ini, digunakan sebuah pembanding algoritma yaitu algoritma *brute-force* yang melakukan pencarian dari *string* paling

kiri ke kanan. Berikut adalah Tabel yang menggambarkan ilustrasi pencocokan karakter yang dilakukan oleh Algoritma *Brute Force*.

Tabel 4. 2 Simulasi Cara Kerja Algoritma Brute Force

Proses ke-	Iterasi Ke-	Patter n	Teks	Teks di Database	Pencocokan Brute Force	Hasil Sugesti
1	1	h	h	<i>Handphone</i> Samsung	h = h	<i>Handphone</i> Samsung
	2		h	<i>Headset</i> Putih	h = h	<i>Headset</i> Putih
	3		h	Buku Hitam Notes	h = h	Buku Hitam Notes
	4		h	<i>Charger</i> Iphone	h = h	<i>Charger</i> Iphone
	5		h	<i>Hardisk</i>	h = h	<i>Hardisk</i>
	6		h	Buku Tulis Merah	h = h	Buku Tulis Merah
	7		h	<i>Mouse</i> Merah Kecil	h = h	<i>Mouse</i> Merah Kecil
	8		d	Dompot Kulit	h ≠ d	-
	9		h	Buku Bahasa Jurnalistik	h = h	Buku Bahasa Jurnalistik
	10		b	Buku Database	h ≠ b	-
2	1	ha	ha	<i>Handphone</i> Samsung	h = h	-
	2				a = a	<i>Handphone</i> Samsung
	5		he	<i>Headset</i> Putih	h = h	-
	6				h ≠ e	-
	7		hi	Buku Hitam Notes	h = h	-
	8				h ≠ h	-
	9		ha	<i>Charger</i> Iphone	h = h	-
	10				a = a	<i>Charger</i> Iphone
	11		ha	<i>Hardisk</i>	h = h	-
	12				a = a	<i>Hardisk</i>
	13		h-	Buku Tulis Merah	h = h	-
	14				a ≠ -	-
	15		h-	<i>Mouse</i> Merah Kecil	h = h	-
	16				a ≠ -	-
	17		do	Dompot Kulit	h ≠ d	-
	18		bu	Buku Bahasa Jurnalistik	h ≠ b	-
	19		bu	Buku Database	h ≠ b	-
3	1	har	han	<i>Handphone</i> Samsung	h = h	-
	2				a = a	-
	3				r ≠ n	-

Proses ke-	Iterasi Ke-	Patter n	Teks	Teks di Database	Pencocokan Brute Force	Hasil Sugesti
	4		hon		$h = h$	-
	5				$a \neq o$	-
	6		hea	Headset Putih	$h = h$	-
	7				$a \neq e$	-
	8		hit	Buku Hitam Notes	$h = h$	-
	9				$a \neq i$	-
	10		har	Charger Iphone	$h = h$	-
	11				$a = a$	-
	12				$r = r$	Charger Iphone
	13		har	Hardisk	$h = h$	-
	14				$a = a$	-
	15				$r = r$	Hardisk
	16		h--	Buku Tulis Merah	$h = h$	-
	17				$a \neq -$	-
	18		h-k	Mouse Merah Kecil	$h = h$	-
	19				$h \neq -$	-
	20		dom	Dompot Kulit	$h \neq d$	-
	21		has	Buku Bahasa Jurnalistik	$h = h$	-
	22				$a = a$	-
	23				$r \neq s$	-
	24		buk	Buku Database	$h \neq b$	-
4	1	hard	hand	Handphone Samsung	$h = h$	-
	2				$a = a$	-
	3				$r \neq n$	-
	4		hone		$h = h$	
	5				$a \neq o$	
	6		head	Headset Putih	$h = h$	-
	7				$a \neq e$	-
	8		hita	Buku Hitam Notes	$h = h$	-
	9				$a \neq i$	-
	10		harg	Charger Iphone	$h = h$	-
	11				$a = a$	-
	12				$r = r$	-
	13				$d \neq g$	-
	14		hone		$h = h$	-
	15				$a \neq o$	-
	16		hard	Hardisk	$h = h$	-
	17				$a = a$	-
	18				$r = r$	-
	19				$d = d$	Hardisk

Proses ke-	Iterasi Ke-	Pattern	Teks	Teks di Database	Pencocokan Brute Force	Hasil Sugesti
	20		h---	Buku Tulis Merah	$h = h$	-
	21				$a \neq -$	-
	22		mous	Mouse Merah Kecil	$h \neq m$	-
	23		domp	Dompot Kulit	$h \neq d$	-
	24		hasa	Buku Bahasa Jurnalistik	$h = h$	-
	25				$a = a$	-
	26				$r \neq s$	-
	27		buku	Buku Database	$h \neq b$	-

Tabel 4. 3 Simulasi Cara Kerja Algoritma *Boyer-Moore*

Proses ke	Iterasi Ke-	Pattern	Teks	Teks di Database	Pencocokan Boyer Moore	Hasil Sugesti
1	1	h	h	Handphone Samsung	$h = h$	Handphone Samsung
	2		h	Headset Putih	$h = h$	Headset Putih
	3		h	Buku Hitam Notes	$h = h$	Buku Hitam Notes
	4		h	Charger Iphone	$h = h$	Charger Iphone
	5		h	Hardisk	$h = h$	Hardisk
	6		h	Buku Tulis Merah	$h = h$	Buku Tulis Merah
	7		h	Mouse Merah Kecil	$h = h$	Mouse Merah Kecil
	8		d	Dompot Kulit	$h \neq d$	-
	9		b	Buku Bahasa Jurnalistik	$h \neq b$	-
	10		b	Buku Database	$h \neq b$	-
2	1	ha	ha	Handphone Samsung	$a = a$	-
	2				$h = h$	Handphone Samsung
	3		he	Headset Putih	$a \neq e$	-
	4		hi	Buku Hitam Notes	$a \neq i$	-
	5		ha	Charger Iphone	$a = a$	-
	6				$h = h$	Charger Iphone
	8		ha	Hardisk	$a = a$	-
	9				$h = h$	Hardisk
	10		h-	Buku Tulis Merah	$a \neq -$	-
	11		bu	Mouse Merah Kecil	$a \neq u$	-
	12		do	Dompot Kulit	$a \neq o$	-
	13		ha		$a = a$	-

Proses ke	Iterasi Ke-	Pattern	Teks	Teks di Database	Pencocokan Boyer Moore	Hasil Sugesti
	14			Buku Bahasa Jurnalistik	$h = h$	Buku Bahasa Jurnalistik
	15		bu	Buku Database	$a \neq u$	-
3	1	har	han	<i>Handphone</i>	$r \neq n$	-
	2		hon	Samsung	$r \neq n$	-
	3		hea	<i>Headset</i> Putih	$r \neq a$	-
	4		hit	Buku Hitam Notes	$r \neq t$	-
	5		har	<i>Charger</i> Iphone	$r = r$	-
	6				$a = a$	-
	7				$h = h$	<i>Charger</i> Iphone
	9		har	<i>Hardisk</i>	$r = r$	-
	10				$a = a$	-
	11				$h = h$	<i>Hardisk</i>
	12		h--	Buku Tulis Merah	$r \neq -$	-
	13		h-k	<i>Mouse</i> Merah Kecil	$r \neq k$	-
	14		dom	Dompot Kulit	$r \neq m$	-
	15		has	Buku Bahasa Jurnalistik	$r \neq s$	-
	16		buk	Buku Database	$r \neq k$	-
4	1	hard	hand	<i>Handphone</i> Samsung	$d = d$	-
	2				$r \neq n$	-
	3		hone		$d \neq e$	-
	4		head	<i>Headset</i> Putih	$d = d$	-
	5				$r \neq a$	-
	6		hita	Buku Hitam Notes	$d \neq a$	-
	7		harg	<i>Charger</i> Iphone	$d \neq g$	-
	8		hone		$d \neq e$	-
	9		hard	<i>Hardisk</i>	$d = d$	-
	10				$r = r$	-
	11				$a = a$	-
	12				$h = h$	<i>Hardisk</i>
	13		h---	Buku Tulis Merah	$d \neq -$	-
	14		h-ke	<i>Mouse</i> Merah Kecil	$d \neq e$	-
	15		domp	Dompot Kulit	$d \neq p$	-
	16		hasa	Buku Bahasa Jurnalistik	$d \neq a$	-
	17		buku	Buku Database	$d \neq u$	-

4.7.3 Menentukan Bobot Kriteria

Penentuan bobot kriteria dijelaskan pada Tabel berikut:

Tabel 4. 4 Pembobotan Kriteria

Kriteria	Persentase pengaruh kriteria	Bobot range (0-1)	Keterangan
Jumlah iterasi algoritma	70%	0,7	Tingkat pengaruh iterasi algoritma terhadap kecepatan sangat tinggi karena semakin banyak perulangan/iterasi maka akan semakin lambat suatu algoritma menyelesaikan masalah
Jumlah huruf pada <i>pattern</i>	30%	0,3	Jumlah huruf pada <i>pattern</i> merupakan kriteria yang juga mempengaruhi kecepatan, namun tidak lebih berpengaruh dari pada relasi

4.7.4 Pemberian Nilai Pada Setiap Kriteria

Tahap ini adalah tahap dimana setiap kriteria yang telah terbentuk diberi nilai. Untuk dapat memberikan nilai, berikut adalah contoh hasil simulasi *auto-complete* Universitas Bakrie yang diambil dari pembahasan analisa pada bagian 4.7.2.

Tabel 4. 5 Pemberian Nilai Pada Setiap Kriteria

Alternatif	Proses ke-	Jumlah Iterasi Algoritma	Jumlah Huruf Pada Pattern
<i>Boyer-Moore</i>	1	10	1
	2	15	2
	3	16	3
	4	17	4
<i>Brute Force</i>	1	10	1
	2	19	2
	3	24	3
	4	27	4

4.7.5 Menghitung Skor

Nilai pada kriteria dimasukkan dalam nilai bobot yang telah ditentukan. Tahapan selanjutnya adalah dengan melakukan perhitungan menggunakan rumus Metode Perbandingan Eksponensial.

Tabel 4. 6 Simulasi Perhitungan Analisa Menggunakan Perhitungan Perbandingan Eksponensial

Proses Ke-	Kriteria						Total Nilai BF	Total Nilai BM
	Jumlah Iterasi			Jumlah Huruf				
	B	BF	BM	B	BF	BM		
		N	N		N	N		
1	0,7	10	10	0,3	1	1	6,012	6,012
2	0,7	19	15	0,3	2	2	9,086	7,888
3	0,7	24	16	0,3	3	3	10,641	8,355
4	0,7	27	17	0,3	4	4	11,561	8,782
TOTAL							48,245	38,587

Keterangan

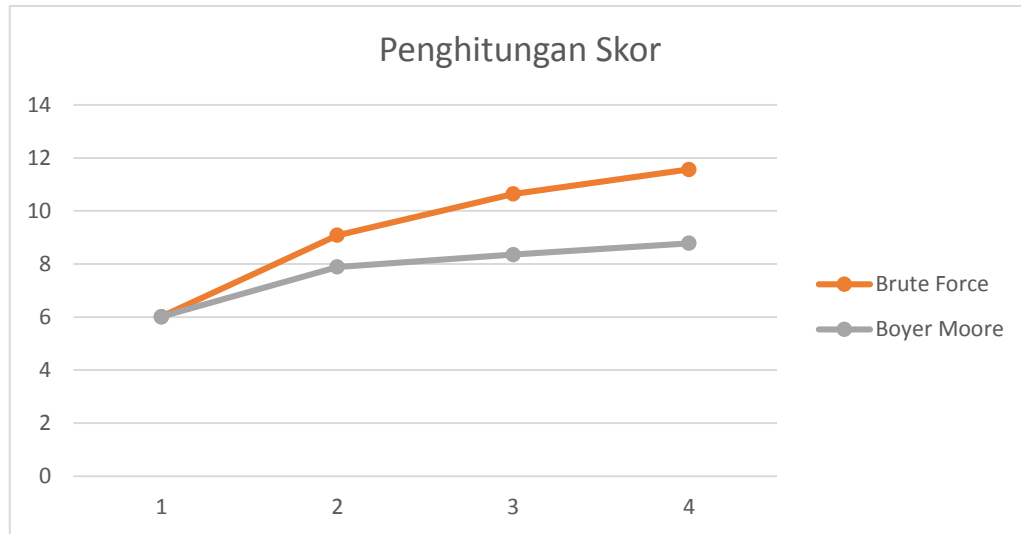
1. BF : Algoritma *Brute Force*
2. BM : Algoritma *Boyer-Moore*
3. B : Nilai Bobot
4. N : Nilai Dari Kriteria
5. Total Nilai : $\sum(N)^B$

Contoh perhitungan :

Nilai pada proses 2.

$$\begin{aligned}
 \text{Nilai BF} &= (19)^{0,7} + (2)^{0,3} \\
 &= 7,855 + 1,231 \\
 &= 9,086
 \end{aligned}$$

$$\begin{aligned}
 \text{Nilai BM} &= (15)^{0,7} + (2)^{0,3} \\
 &= 6,657 + 1,231 \\
 &= 7,888
 \end{aligned}$$



Gambar 20 Grafik Perhitungan Skor

Pada Gambar 4.20 menunjukkan bahwa grafik perhitungan skor algoritma *boyer-moore* lebih rendah daripada algoritma *brute-force*, dalam kasus ini proses pengujian hanya dilakukan untuk mengetahui perbandingan keefektifan algoritma *boyre-moore* dan *brute-force* pada fitur *auto-complete*. Hasil yang diperoleh dalam empat kali percobaan dengan urutan karakter yang dicari sebanyak 1-4 huruf dan dengan panjang maksimal teks sebanyak 25 karakter adalah algoritma *boyer-moore* memiliki jumlah iterasi yang lebih kecil dari pada algoritma *brute-force* dengan hasil total pencapaian algoritma *boyer-moore* adalah 38,587 dan algoritma *brute-force* sebanyak 48,245.

4.7.6 Menentukan Prioritas Keputusan

Setelah total nilai dari setiap alternatif dihitung, maka tahap selanjutnya adalah menentukan prioritas keputusan dari total nilai setiap alternatif. Secara rinci hasil prioritas keputusan dapat dilihat sebagai berikut.

Tabel 4. 7 Prioritas Keputusan

Alternatif	Total Nilai	Rangking
Algoritma Boyer Moore	31,037	1
Algoritma Brute Force	37,299	2

Pada Tabel 4.7 terlihat bahwa total nilai dari alternatif terendah yang memperoleh ranking pertama, hal ini dikarenakan semakin kecil usaha yang dilakukan, maka semakin sedikit iterasi yang diperlukan. Berdasarkan analisa tersebut maka Algoritma *Boyer-Moore* lebih efektif dibandingkan dengan algoritma *brute-force* dalam pencarian menggunakan fitur *auto-complete*. Proses pengujian ini hanya didasarkan oleh penerapan cara kerja algoritma berdasarkan jumlah iterasi, bukan dengan kecepatan algoritma.

BAB V

SIMPULAN DAN SARAN

5.1 Simpulan

Berdasarkan hasil dari analisis, perancangan, implementasi, dan pengujian yang telah dilakukan, dapat diambil kesimpulan bahwa Algoritma *Boyer-Moore* dapat diterapkan pada fitur *auto-complete* aplikasi LF-Pro berbasis *web*. Algoritma *Boyer-Moore* juga dapat berjalan dengan baik pada fitur *auto-complete* pencarian barang dengan melakukan pencarian dari karakter paling kanan ke karakter paling kiri dengan menggunakan pergeseran *pattern good-suffix shift rule* dan *bad character rule*. Dari hasil pengujian menggunakan Metode Perbandingan Eksponensial, grafik perhitungan skor menunjukkan bahwa jumlah iterasi yang dilakukan oleh algoritma *Boyer-Moore* lebih sedikit daripada algoritma *Brute Force* yang dijadikan sebagai algoritma pembanding untuk fitur pencarian menggunakan fitur *auto-complete*. Hasil pencarian algoritma *Boyer-Moore* juga bergantung pada dua hal, yaitu panjang *pattern* yang diinputkan dan kata kunci yang di-input-kan oleh *user*. Apabila *user* memasukkan jumlah *pattern* sedikit maka peluang untuk sistem menampilkan hasil pencarian lebih lama, semakin banyak jumlah *pattern* yang dimasukkan maka semakin kecil sistem mendapatkan sugesti kemiripannya.

5.2 Saran

Berdasarkan hasil penulisan yang telah dilakukan, maka beberapa saran untuk pengembangan penulisan selanjutnya adalah sebagai berikut.

1. Untuk meningkatkan akurasi dari hasil pencarian *auto-complete* menggunakan algoritma *Boyer-Moore*, maka diharapkan *user* melakukan beberapa hal dalam proses pencarian, yang pertama adalah *user* memasukkan data dengan nama barang yang spesifik disertai dengan tipe barang agar menghindari kesamaan barang yang disimpan. Kedua, diharapkan *user* melakukan pencarian dengan memasukkan nama barang yang spesifik (*pattern* yang panjang) supaya hasil pencarian lebih akurat.

2. Aplikasi LF-Pro ini dikembangkan untuk meningkatkan jumlah keakurasian dalam pencarian data menggunakan *auto-complete* dengan menambahkan metode pencocokan *string* lainnya.
3. Aplikasi LF-Pro belum memiliki pengendalian sistem keamanan, sehingga diperlukan penelitian lebih lanjut untuk menyempurnakan sistem dan aplikasi LF-Pro.

DAFTAR PUSTAKA

- Abdeen, R. A. (2011). An Algorithm for String Searching Based on Brute-Force Algorithm . *IJCSNS International Journal of Computer Science and Network Security*, 11(7), 24-28.
- Al-Fedaghi, S. (2011). Developing Web Applications. *International Journal of Software Engineering and Its Applications Vol. 5 No.2*, 5(2), 57-68.
- Argakusumah, K. W., & Hansun, S. (2014). Implementasi Algoritma Boyer-Moore pada Aplikasi Kamus Kedokteran Berbasis Android. *Ultimatics Vol. VI, No. 2 ISSN 2085-4552*, 71.
- Aulia, R. (2008). Analisis Algoritma Knuth Morris Pratt dan Algoritma Boyer Moore dalam Proses Pencarian String . *Strategi Algoritmik* , 1-5.
- Chandran, S. P., & Angepat, M. (2011). Comparison between ASP.NET and PHP - Implementation of a Real Estate Web Application . *Master Thesis in Software Engineering, Malardalens hogskola Eskilstuna Vasteras*, 34-35.
- Chiquita, C. (2012). Penerapan Algoritma Boyer-Moore Dynamic Programming untuk Layanan Auto-Complete dan Auro Correct. *Makalah IF3051 Strategi Algoritma*, 1-6.
- French, A. M. (2011). Web Development Life Cycle: A New Methodology for Developing Web Applications . *Journal of Internet Banking and Commerce*, vol 16 no.2 , 5.
- Hidayani, N., Sari, J. N., & Suharman, R. (2012). Perancangan dan Implementasi Metode Brute Force untuk Pencarian String pada Website PCR. *Program Studi Informatika dan Multimedia, Politeknik Caltex Riau*, 1-10.
- Januardi, A. (2013). Analisa Perbandingan Algoritma Brute Force dan Boyer Moore dalam Pencarian Word Suggestion Menggunakan Metode Perbandingan Eksponensial. *Pelita Informatika Budi Darma ISSN: 2301-9425, IV(1)*, 18 - 24.

- Kamatchi, Iyer, J., & Singh, S. (2013). Software Engineering: Web Development Life Cycle. *International Journal of Engineering Research & Technology (IJERT)* ISSN:2278-0181, 2(3), 1-4.
- Kent. (t.thn.). *Boyer-Moore Algorithm*. Dipetik April 4, 2016, dari Personal Kent: <http://www.personal.kent.edu/~rmuhamma/Algorithms/MyAlgorithms/StringMatch/boyerMoore.htm>
- Kristanto, S., Rachmat, A., & Santosa, R. G. (2013). Implementasi Algoritma Boyer-Moore Pada Permainan Word Search Puzzle. *Conference Paper*, <https://www.researchgate.net/publication/263810958>, 1-9.
- Kumara, G. H. (2009). Visualisasi Beberapa Algoritma Pencocokan String dengan Java. *Jurusan Teknik Informatika, Fakultas Elektro Informatika, Institut Teknologi Bandung*, 1-14.
- Mahardika, D. (2012). Sistem Pendukung Keputusan Promosi Kenaikan Jabatan Manager dengan Metode Perbandingan Eksponensial Pada PT Texmaco Perkasa Engineering Kendal. 1-14.
- Marimin. (2015). *Teknik dan Aplikasi Pengambilan Keputusan dengan Kriteria Majemuk*. Jakarta: Grasindo.
- Mountaines, P. E. (2013). Pengembangan Aplikasi Berbasis Web untuk Menampilkan Absensi dan Nilai Akhir Peserta Didik. *Tugas Akhir Program Studi Sistem Komputer, Fakultas Teknik, Universitas Diponegoro*, 3.
- Mujumdar, A., Masiwal, G., & Chawan, P. (2012). Analysis of various Software Process Models. *International Journal of Engineering Research and Applications (IJERA)* ISSN: 2248 - 9622 vol.2, Issue 3, 2015 - 2021.
- Munassar, N. M., & Govardan, A. (2012). Comparison Study Between Traditional and Object-Oriented Approaches to Develop All Projects in Software Engineering. *International Journal of Computer Science and Information Technologies* , 3(1), 3022-3028.
- Munir, R. (2004-2007). Diktat Kuliah Strategi Algoritmik. *Program Studi Teknik Informatika ITB*.

- Nagila, R. (2013). Comparison of Web Development Technologies - ASP.NET & PHP. *Master Thesis in Web Development Malarden University* , 12.
- O'Brien, J. A., & Marakas, G. M. (2010). *Management Information System 10e*. New York : McGraw-Hill/Irwin .
- Pratiwi, H. R., Syarif, D., & Wibowo, A. (2012). Prototype Aplikasi SMS Content Filtering Menggunakan Metode String Matching (Studi Kasus : Content Iklan). *Jurnal Teknik Informatika* , 1, 1-8.
- Pressman, R. S. (2010). *Software Engineering, A Practitioner's Approach, Seventh Edition* . New York : McGraw-Hill .
- Sagita, V., & Prasetyowati, M. I. (2013). Studi Perbandingan Implementasi Algoritma Boyer-Moore, Turbo Boyer-Moore, dan Tuned Boyer-Moore dalam Pencarian String . *ULTIMATICS, ISSN 2085-4552, IV(1)*, 31-37.
- Suprawoto. (2008). Akurat, Cepat, Mudah, dan Merata Sebuah Praktik Pengelolaan Informasi Publik. *Bali: Konferensi Perpustakaan Digital Indonesia ke-1*, 1-11.
- Supriyanto. (2014). Perancangan dan Pembuatan Sistem Informasi Kehilangan Berbasis Web. *Jurusan Elektro, Fakultas Teknik Universitas Muhammadiyah Surakarta*, 1-12.
- Utomo, D., Harjo, E. W., & Handoko. (2011). Perbandingan Algoritma String Searching Brute Force, Knuth Morris Pratt, Boyer Moore, dan Karp Rabin pada Teks Alkitab Bahasa Indonesia. *Techne Jurnal Ilmiah Elektronika* , 7(1), 1-13.
- Whitten, J. L., & Bentley, L. D. (2007). *System Analysis and Design Methods 7/E*. New York : McGraw-Hill.
- Wibisono, D. A., Priharsari, D., & Muttaqin, A. (2012). Rancang Bangun Sistem Informasi Pencarian Benda Hilang 'Lost and Found' Berbasis Website di Universitas Brawijaya. *Pemrograman Teknologi Informasi dan Ilmu Komputer, Informatika, Universitas Brawijaya*, 1-8.

LAMPIRAN

Lampiran 1 Software Requirement Specification

SOFTWARE REQUIREMENT SPECIFICATION (SRS)

Implementasi Algoritma Boyer-Moore dalam Aplikasi LF-Pro (*Lost and Found Property*) di Universitas Bakrie

1. Pendahuluan

Software Requirement Specification (SRS) adalah dokumen yang menjelaskan rincian kebutuhan spesifik dalam perancangan aplikasi LF-Pro di Universitas Bakrie berbasis *web* yang terdiri dari kebutuhan fungsional dan non-fungsional sesuai dengan kriteria sistem yang akan dirancang. *Software Requirement Specification* (SRS) ini dibuat untuk membantu spesifikasi aplikasi LF-Pro. Kriteria yang diperlukan dijelaskan secara rinci dan sistematis karena digunakan sebagai evaluasi hasil analisa aplikasi LF-Pro yang dirancang, agar menjawab permasalahan sesuai dengan keperluan *user* dan tidak menyimpang dari tujuan pembuatan sistem informasi tersebut. Sistem informasi yang sesuai dengan kriteria yang telah ditentukan sebelumnya akan berjalan lebih lancar dalam pengoperasiannya. Dan *Software Requirement Specification* (SRS) dapat digunakan sebagai acuan evaluasi dalam pengerjaan sistem informasi agar dapat berjalan sesuai dengan kriteria yang diinginkan.

1.1 Latar Belakang

Kehilangan barang-barang pribadi di Universitas Bakrie merupakan hal yang sudah sering terjadi. Sebagian besar mahasiswa yang merasa kehilangan atau menemukan barang di lingkungan Universitas Bakrie akan segera melapor ke petugas *security* terdekat. Hal ini yang membuat *staff security* merasa kesulitan dalam melakukan penyimpanan dan pencarian barang yang telah ditemukan atau barang yang dicari karena jangka waktu penemuan terlalu lama ataupun catatan yang sudah menumpuk. Dalam satu minggu, pihak *security* bisa menerima laporan kehilangan atau penemuan sebanyak 10 kali laporan, jumlah ini termasuk cukup banyak mengingat jumlah mahasiswa di Universitas Bakrie yang sangat banyak. Dari permasalahan tersebut, penulis merasa perlu adanya sebuah sistem atau sarana khusus yang

dapat menampung data temuan barang dan lokasi penyimpanan barang tersebut. Salah satunya dengan membangun aplikasi LF-Pro berbasis *web* yang dapat diakses oleh *staff security* di berbagai pos keamanan di Universitas Bakrie. Dengan adanya aplikasi temuan barang ini akan memudahkan *staff security* untuk mengelola temuan barang dan laporan kehilangan secara efektif.

1.2 Tujuan

Software Requirement Specification (SRS) ini diharapkan menjadi acuan evaluasi dalam pengerjaan pengembangan aplikasi LF-Pro. *Software Requirement Specification* (SRS) ini digunakan sebagai standar pembuatan mengacu pada penjelasan rincian kebutuhan spesifik dalam perancangan LF-Pro yang terdiri dari kebutuhan fungsional dan non-fungsional sesuai dengan kriteria aplikasi LF-Pro yang akan dirancang. Dalam pengerjaannya diharapkan aplikasi LF-Pro tidak menyimpang dari standar dan kriteria pembuatan sehingga dapat menghasilkan sistem informasi yang memuaskan dan bermanfaat bagi *user*.

1.3 Batasan

Batasan yakni ruang lingkup kebutuhan yang dibutuhkan dalam perancangan aplikasi LF-Pro. Dalam dokumen ini dijelaskan agar dalam pengembangan aplikasi LF-Pro tidak keluar dari topik masalah yang ada maka terdapat batasan sistem informasi yang dibuat. Aplikasi LF-Pro hanya melakukan pencatatan dan dokumentasi pada laporan civitas akademika yang menemukan barang dan kehilangan barang. Semua kegiatan dalam sistem informasi membutuhkan *login* untuk otorisasi dan autentikasi pengguna. *User* dapat mengelola data penemuan barang dan laporan kehilangan barang yang disesuaikan dengan sistem yang berlaku di *security* Universitas Bakrie. Secara teknis, aplikasi LF-Pro berbasis *web* dengan *database* MySQL. Pengujian algoritma pada aplikasi LF-Pro menggunakan Metode Perbandingan Eksponensial.

1.4 Definisi, Istilah dan Singkatan

- **Software Requirement Specification (SRS):** Dokumen yang menggambarkan secara jelas dan rinci spesifikasi kebutuhan *software* dalam pembuatan aplikasi LF-Pro berbasis *web* pada bagian *security*.
- **Software:** Perangkat lunak yang dijalankan di komputer.
- **Hardware:** Perangkat keras yang secara fisik digunakan dalam pengoperasian komputer.
- **Interface:** Tampilan pada layar yang ditampilkan ke pengguna.
- **Web Server:** Pusat komputer/sistem yang mengelola data yang terhubung dengan jaringan.

1.5 Tinjauan

Software Requirement Specification (SRS) ini diharapkan menjadi acuan evaluasi dalam pengerjaan pengembangan aplikasi LF-Pro. *Software Requirement Specification (SRS)* ini digunakan sebagai standar pembuatan mengacu pada penjelasan rincian kebutuhan spesifik dalam perancangan aplikasi LF-Pro yang terdiri dari kebutuhan fungsional dan non-fungsional sesuai dengan kriteria aplikasi LF-Pro yang dirancang. Dalam pengerjaannya diharapkan aplikasi LF-Pro tidak menyimpang dari standar dan kriteria pembuatan sehingga dapat menghasilkan sistem informasi yang memuaskan dan bermanfaat bagi *user*.

2. Deskripsi Keseluruhan

Berdasarkan studi awal yang telah dilakukan, didapatkan hasil bahwa kebutuhan pengguna merujuk pada pencatatan laporan kehilangan dan penemuan barang yang memiliki fungsi antara lain melakukan pencatatan laporan kehilangan barang pada *form* kehilangan barang, pencatatan laporan penemuan barang pada *form* penemuan barang, konfirmasi barang dan pencarian data barang (*searching*). Berdasarkan fungsi tersebut, akan dibuat sebuah aplikasi LF-Pro berbasis *web* yang akan digunakan oleh *security* Universitas Bakrie.

2.1 Perspektif Produk

Menurut penjelasan sebelumnya, bagian *security* Universitas Bakrie membutuhkan sistem pencatatan laporan dan dokumentasi barang hilang yang dapat meningkatkan kinerja untuk pengelolaan data barang hilang dan laporan kehilangan barang. Dengan adanya permintaan *user* dan batasan dalam perancangan aplikasi LF-Pro, ada beberapa fungsi yang dikembangkan yakni:

1. Sistem dapat digunakan untuk melakukan pencarian barang hilang dan barang temuan
2. Sistem dapat menampilkan data barang temuan dan barang hilang
3. Sistem dapat menyimpan data barang temuan dan barang hilang
4. Sistem dapat mengedit data barang hilang
5. Sistem dapat melakukan konfirmasi barang
6. Sistem dapat mengirim konfirmasi barang via email
7. Terdapat menu *login* untuk dapat masuk ke dalam sistem
8. Terdapat menu *logout* untuk keluar dari sistem

2.2 Karakteristik Pengguna

User	Action
<i>Security</i>	<ol style="list-style-type: none"> 1. <i>Security</i> dapat melakukan <i>login</i> ke aplikasi LF-Pro 2. <i>Security</i> dapat melihat <i>form</i> penemuan dan kehilangan barang 3. <i>Security</i> dapat menginputkan data penemuan dan laporan kehilangan barang 4. <i>Security</i> dapat melakukan pencarian data barang hilang atau laporan kehilangan barang 5. <i>Security</i> dapat mengupdate data barang hilang 6. <i>Secuirty</i> dapat melakukan <i>logout</i>

2.3 Batasan

Pada SRS ini, batasan dari pengerjaan aplikasi LF-Pro berbasis *web* pada bagian *security* Universitas Bakrie antara lain:

1. Aplikasi LF-Pro dikembangkan dengan bahasa pemrograman PHP dan berbasis *web*.
2. *Database* yang digunakan adalah MySQL
3. Untuk memudahkan implementasi maka diawali dengan data sampel
4. Pengujian menggunakan Metode Perbandingan Eksponensial dengan membandingkan cara kerja algoritma *Boyer-Moore* dengan *Brute-Force* berdasarkan jumlah iterasi
5. Tidak membahas keamanan data dan keamanan jaringan
6. Pengguna aplikasi LF-Pro adalah *staff security* Universitas Bakrie.

2.4 Asumsi dan Ketergantungan

Aplikasi LF-Pro yang akan dibangun sangat tergantung pada koneksi internet dan *server* yang digunakan. Apabila koneksi internet yang digunakan oleh pengguna lambat, maka kinerja aplikasi LF-Pro juga akan lambat. *Server down* atau terjadi kerusakan juga akan membuat aplikasi LF-Pro tidak dapat diakses oleh pengguna.

2.5 Metode Pengembangan

Pengembangan aplikasi LF-Pro ini menggunakan WDLC (*Web Development Life Cycle*). Perancangan aplikasi LF-Pro diperlukan dalam pengembangan perangkat lunak yang bertujuan agar aplikasi tersebut dapat memenuhi kebutuhan klien. Pemahaman klien dan kebutuhan klien seringkali berkembang, konsekuensi dari kenyataan ini yakni sistem melampaui spesifikasi yang dirancang dan digunakan dalam konteks yang lebih luas. Hal ini mempersulit kemampuan untuk secara jelas menentukan sistem persyaratan. Metode WDLC dipilih karena metode ini adalah metode pengembangan dari metode *prototyping* dan SDLC (*Software Development Life Cycle*). WDLC menggunakan komponen dari masing-masing metodologi,

menggabungkan ke dalam sebuah pendekatan baru yang akan mengurangi waktu pengembangan, menambahkan struktur untuk masalah yang tidak terstruktur dan menjaga pengguna yang terlibat dalam seluruh siklus hidup pengembangan.

3. Spesifikasi Kebutuhan

Pembuatan aplikasi LF-Pro didasarkan kepada kebutuhan pengguna yang diperoleh melalui proses elisitasi. Proses elisitasi kebutuhan dilakukan melalui wawancara dengan beberapa pihak yang akan menggunakan sistem informasi inventaris ini.

3.1 Fungsional

3.1.1 Aplikasi LF-Pro yang terintegrasi

= Menampilkan tampilan yang barang hilang dan temuan yang beroperasi dengan baik dan bersesuaian dengan fungsi-fungsi yang ditampilkan.

3.1.2 Menampilkan Master Menu *Home*

= Menampilkan tampilan yang dapat melihat semua master menu.

3.1.3 Menampilkan Master Menu Data Barang Hilang

= Menampilkan tampilan yang dapat melihat semua data laporan barang hilang beserta menu yang memfasilitasi tampilan data barang hilang beserta menu yang memfasilitasi tampilan tersebut baik menambah, mengedit, dan menghapus data.

3.1.4 Menampilkan Master Menu Data Barang Temuan

= Menampilkan tampilan yang dapat melihat semua data laporan barang hilang beserta menu yang memfasilitasi tampilan data barang hilang beserta menu yang memfasilitasi tampilan tersebut baik menambah, mengedit, dan menghapus data.

3.1.5 Menampilkan Master Menu Penambahan Data Barang Hilang

= Menampilkan *form* dalam menambahkan data laporan barang hilang.

3.1.6 Menampilkan Menu *Search* untuk Pencarian

= Menampilkan fasilitas pencarian nama barang berdasarkan label barang.

3.1.7 Menampilkan Menu *Edit* Data Barang Hilang

= Menampilkan *form edit* untuk mengubah keterangan barang hilang.

3.1.8 Menampilkan Menu *Paging*

= Menampilkan *paging* data agar data tertata berdasar *paging*.

3.1.9 Menampilkan Fasilitas untuk *Login User*

= Menampilkan *form* untuk masuk ke dalam aplikasi LF-Pro.

3.1.10 Menampilkan nama *user* pada halaman *home*

= Terdapat keterangan nama pengguna *user* berdasarkan dari *privilage login*

3.1.11 Menampilkan Nama dan alamat Universitas Bakrie

= Terdapat keterangan nama dan alamat Universitas di bagian *header* dan *footer*.

3.1.12 Menyediakan fasilitas *logout*

= Terdapat *button* untuk mengakhiri sesi dalam aplikasi LF-Pro.

3.2 Non-fungsional

3.2.1 Tampilan sistem *user friendly*

= Tampilan LF-Pro yang mudah dipahami.

3.2.2 Tampilan sistem simpel dan menarik

= Tampilan LF-Pro simpel dan menarik bagi pengguna

3.3 Logical View

Client Layer : *Layer* ini merupakan tampilan halaman *web* dari aplikasi LF-Pro yang akan dibuat. Pengguna dapat melakukan akses ke aplikasi LF-Pro ini dengan jaringan internet. Fungsi yang akan ditampilkan bergantung kepada hak akses yang dimiliki oleh pengguna.

Business Layer: Aplikasi LF-Pro akan bertugas untuk menerima *request* yang telah dikirimkan melalui *client layer*. Sistem informasi inventaris akan menerima *query* yang telah diinputkan oleh pengguna untuk diteruskan ke *database layer*. Setelah *query* dieksekusi, hasil akan ditampilkan kembali ke *client layer*.

Database Layer: Penyimpanan data akan dilakukan pada *layer* ini agar data dapat diakses oleh aplikasi LF-Pro untuk memenuhi *request* dari pengguna melalui *client layer*.

3.4 Antarmuka

Kebutuhan antarmuka yang didefinisikan pada dokumen ini mencakup antarmuka pengguna, antarmuka perangkat keras, dan antarmuka komunikasi.

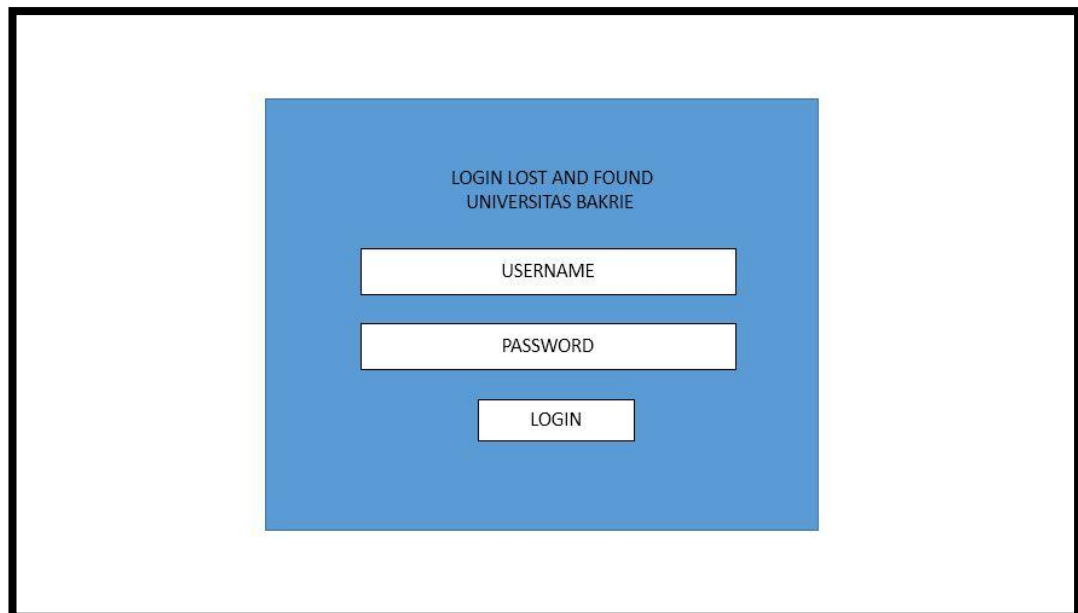
3.4.1 Antarmuka Pengguna

User interface dari aplikasi LF-Pro menggunakan desain antarmuka yang merupakan bagian dari LF-Pro yang memiliki peran penting dan membantu pengguna untuk melakukan kegiatan dengan aplikasi LF-Pro tersebut.

Perancangan aplikasi sangat dibutuhkan dalam sebuah pengembangan software. Adapun model perancangan aplikasi adalah sebagai berikut :

1. *User Interface*

Berikut adalah rancangan *Graphic User Interface* (GUI) dari aplikasi LF-Pro yang akan dibuat.



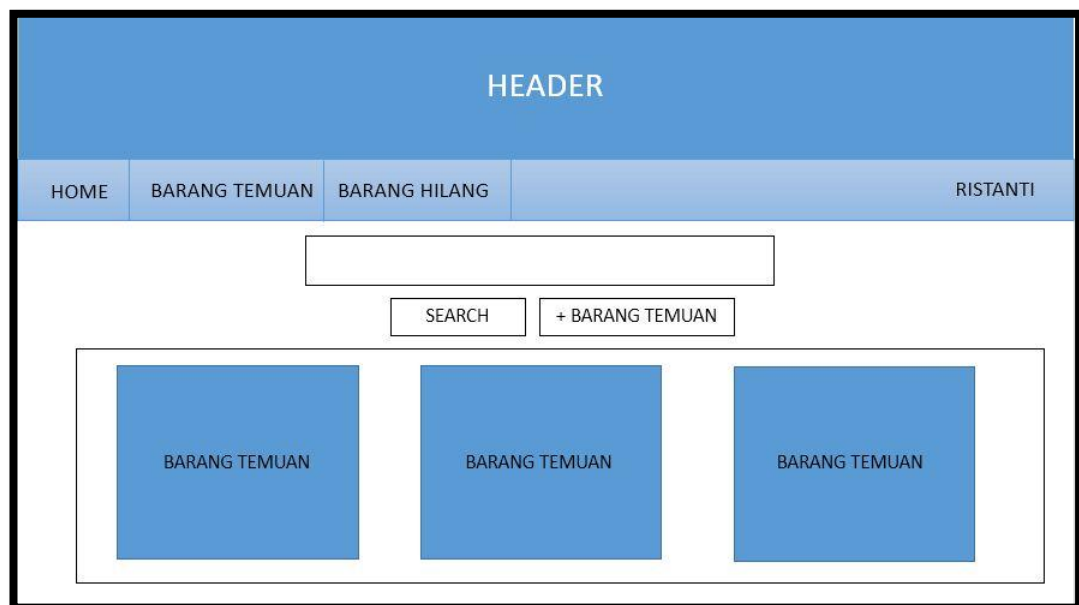
Gambar 21. GUI Halaman Awal User

Gambar 1 menampilkan halaman ketika *user* membuka aplikasi saat pertama kali dan klik menu 'LOGIN'. Terdapat beberapa menu navigasi yang dapat dipilih.



Gambar 22. Halaman *Home*

Gambar 2 menggambarkan halaman *home*, ketika *user* sudah melakukan *login*.



Gambar 23. Halaman Barang Temuan

Gambar 3 menggambarkan halaman awal setelah *user* melakukan ‘klik’ Barang Temuan pada navigasi.

HEADER

HOME BARANG TEMUAN BARANG HILANG RISTANTI

SEARCH + BARANG HILANG

TABEL DATA BARANG HILANG

Gambar 24 Tampilan Barang Hilang

Gambar 4 menggambarkan tampilan barang hilang ketika *user* melakukan ‘klik’ barang hilang pada navigasi.

HEADER

HOME BARANG TEMUAN BARANG HILANG RISTANTI

A. Data Barang Temuan

Nama Barang

Jumlah Barang

Tanggal Ditemukan

Lokasi Ditemukan

Gambar Barang

Kondisi Barang

Tipe Barang

B. Data Penemu

Nama Penemu

Jenis Kelamin

Email

SIMPAN DATA HOME

Gambar 25 *Insert Data Barang*

Gambar 5 Menggambarkan pengisian form barang temuan yang kemudian disimpan ke dalam *database*.

Gambar 26 Konfirmasi Barang

Gambar 6 menggambarkan form konfirmasi barang apabila telah cocok dengan barang yang dilaporkan hilang.

3.4.2 Antarmuka Perangkat Keras

Antarmuka perangkat keras yang dibutuhkan untuk membantu kelengkapan dari pembangunan LF-Pro yang sedang dirancang meliputi:

- Keyboard*, merupakan salah satu alat yang digunakan dalam proses input informasi berbentuk papan yang terdiri atas tombol-tombol seperti huruf alfabet (A-Z) serta simbol untuk pengetikan.
- Mouse*, perangkat yang mendeteksi gerakan *input* dari pengguna dengan melakukan *click*, *drag*, dll.
- Monitor, *display* adalah tampilan *visual* elektronik untuk komputer yang memudahkan pengguna melihat hasil *output* dari sistem.

3.4.3 Antarmuka Komunikasi

Dalam pengembangan *web* ini, dibutuhkan perangkat lunak untuk mendukung pengembangan *web*. Perangkat lunak untuk mendukung pengembangan *web*. Perangkat lunak tersebut antara lain:

a. Sistem Operasi

Sistem Operasi (Server) : Apache (xampp)

Sistem Operasi (Client) : Windows 7

b. Bahasa Pemrograman

Bahasa : PHP dan HTML

Aplikasi : Sublime Text

c. RDBMS

Nama RDBMS : MySQL

Aplikasi : xampp

3.4.4 Antramuka Komunikasi

Desain antarmuka komunikasi dari aplikasi LF-Pro akan dibangun menggunakan *Apache webserver* sebagai penghubung antara *server* dengan komputer pengguna. Dengan menggunakan jaringan internet, pengguna dapat melakukan akses terhadap aplikasi LF-Pro kapanpun dan dimanapun.

3.5 Persyaratan Perijinan

Implementasi dan instalasi aplikasi LF-Pro ini akan terdistribusi sesuai dengan ketentuan operasional yang berlaku.

3.6 Hukum, Hak Cipta dan Pemberitahuan Lainnya

Hak cipta aplikasi LF-Pro merupakan milik pengembang proyek dan bagian *security* Universitas Bakrie. Masing-masing pihak tidak dapat mendistribusikan aplikasi LF-Pro kepada pihak lain tanpa adanya kesepakatan bersama.

3.7 Applicable Standards

- XAMPP v3.2.2
- Sublime Text

4. Informasi Pendukung

Dokumen-dokumen yang terkait dalam pembuatan *Software Requirement Specification* (SRS) ini meliputi:

- Transkrip wawancara
- Surat keterangan penelitian dengan koordinator *security* Universitas Bakrie

Lampiran 2 Elisitasi LF-Pro

Terlampir di bawah ini Elisitasi LF-Pro yang mendeskripsikan kebutuhan *functional* dan *non-functional* aplikasi LF-Pro.

<i>Functional</i>	
1.	Aplikasi LF-Pro yang terintegrasi sebagai <i>working prototype</i>
2.	Menampilkan Master Menu <i>Home</i>
3.	Menampilkan Master Menu Data Barang Hilang
4.	Menampilkan Master Menu Data Barang Temuan
5.	Menampilkan Master Menu Penambahan Data Barang Hilang
6.	Menampilkan Menu <i>Search</i> untuk pencarian
7.	Menampilkan Menu <i>Edit</i> Data Barang Hilang
8.	Menampilkan Menu Paging
9.	Menampilkan Fasilitas untuk <i>Login User</i>
10.	Menyediakan fasilitas <i>Logout</i>
11.	Menampilkan Nama <i>user</i> pada halaman <i>home</i>
12.	Menampilkan Nama dan alamat Universitas Bakrie
<i>Non-Functional</i>	
1.	Tampilan sistem simple dan menarik
2.	Tampilan sistem <i>user friendly</i>

Lampiran 3. Rencana Kegiatan Penelitian

No	Rancangan Penelitian	2016																											
		Februari				Maret				April				Mei				Juni				Juli				Agustus			
		1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
1	Pengamatan dan Observasi																												
2	Menentukan Rumusan Masalah																												
3	Studi Literatur																												
4	Penulisan Bab I - Pendahuluan																												
5	Penulisan Bab II - Landasan Teori																												
6	Penulisan Bab III - Metodologi Penelitian																												
7	Pengajuan Proposal																												
8	Seminar Proposal																												
9	Revisi Proposal																												
10	Perancangan dan pembangunan aplikasi																												
11	Testing																												
12	Penulisan Bab IV - Pembahasan																												
13	Penulisan Bab V - Kesimpulan dan Saran																												
14	Pengajuan Sidang TA																												
15	Sidang TA																												
16	Revisi Laporan TA																												

Lampiran 4. Surat Keterangan Penelitian

Terlampir di belakang halaman ini surat keterangan penelitian yang telah ditanda tangani oleh Bapak Surayo selaku Koordinator *Security* Universitas Bakrie

Lampiran 5. Hasil Wawancara

Terlampir di belakang halaman ini Transkrip Wawancara yang dilakukan kepada Bapak Surayo selaku Koordinator *Security* Universitas Bakrie.

Lampiran 6. Algoritma Boyer-Moore

```

class search
{

public function setBoyerMoore($text, $pattern) {

    $textLower = strtolower($text);
    $patternLower = strtolower($pattern);

    $patlen = strlen($pattern);
    $textlen = strlen($text);
    $table = $this->makeCharTable($patternLower);

    $p = 0;
    $q = 0;
    $arr = array(""); //hasil akan disimpan di array ini
    $i=$patlen-1;

    while ($i < $textlen) {
        $t = $i;
        for ($j=$patlen-1; $patternLower[$j] == $textLower[$i]; $j--,$i--) {
            if($j == 0) {
                //untuk mundur mengambil satu nama barang jika
                ditemukan
                //pattern pada tengah kata nama barang
                if ($i == 0 || $text[$i - 1] == '*') { //'*' sebagai
                pembatas nama barang
                    $q = $i; //i adalah index dimana pattern ditemukan
                }
                else {
                    $x = $i;
                    for ($x = $i; $text[$x] != '*'; $x--) {
                        if ($x == 0)
                            break;
                    }
                    $q = ($x == 0 ? $x : $x + 1);
                }
                $str = '';

                while ($text[$q] != '*') { //gandakan string untuk
                hasil temuan
                    $str .= $text[$q];
                    $q++;
                    $t = $q;
                }
                $arr[$p] = $str;
                $p++;

                break;
            }
        }

        $i = $t;
        if(array_key_exists($textLower[$i], $table)) { //jika
        karakter pada text ditemukan pada pattern
    
```

```

        $i = $i + max($table[$textLower[$i]], 1); //maju sejauh
index pada ditemukannya karakter itu,
        //atau jika diawal pattern, maju satu langkah
    }
    else { //karakter tidak ada di pattern, maka akan dimajukan
sejauh panjang pattern
        $i += $patlen;
    }
}

return $arr;
}

function makeCharTable($string) {
    $len = strlen($string);
    $table = array();
    for ($i=0; $i < $len; $i++) {
        $table[$string[$i]] = $len - $i - 1;
    }
    return $table;
}

function cariTemuan($key) {
    $barang = new barang();
    $hasil = '';
    $table = 'tb_found';
    $res= $barang->getNamaBarangFrom($table);
    $size = sizeof($res);
    for ($i = 0; $i < $size; $i++) {
        foreach ($res[$i] as $a => $b) {
            $hasil .= ($b . "*");
        }
    }

    $cari = $this->setBoyerMoore($hasil, $key);
    return $cari;
}

function cariHilang($key) {
    $barang = new barang();
    $hasil = '';
    $table = 'tb_lost';
    $res= $barang->getNamaBarangFrom($table);
    $size = sizeof($res);
    for ($i = 0; $i < $size; $i++) {
        foreach ($res[$i] as $a => $b) {
            $hasil .= ($b . "*");
        }
    }

    $cari = $this->setBoyerMoore($hasil, $key);
    return $cari;
}
?>

```

Lampiran 7. Surat Pengujian Aplikasi

Terlampir di belakang halaman ini surat pengujian aplikasi yang telah ditandatangani oleh Bapak Surayo selaku Koordinator *Security* Universitas Bakrie sebagai *user* dari penelitian ini.