­­DETEKSI PLAGIARISME PADA DOKUMEN

SKRIPSI BERDASARKAN TINGKAT KESAMAAN

DENGAN MENGGUNAKAN METODE *LONGEST COMMON SUBSEQUENCE*

SKRIPSI

Digunakan Sebagai Syarat Maju Ujian Diploma IV Politeknik Negeri Malang

Oleh:

IMAM NAWAWI NIM. 1541180020



PROGRAM STUDI TEKNIK INFORMATIKA

JURUSAN TEKNOLOGI INFORMASI

POLITEKNIK NEGERI MALANG

2019

# HALAMAN JUDUL

# HALAMAN PENGESAHAN

# HALAMAN PERNYATAAN

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

Malang, 17 Juni 2019

Imam Nawawi

# ABSTRAK

**Nawawi, Imam.** “Deteksi Plagiarisme Pada Dokumen Skripsi Berdasarkan Tingkat Kesamaan Dengan Menggunakan Metode *Longest Common Subsequence*”.

**Pembimbing: (1) Putra Prima Arhandi, S.T., M.Kom. (2) Dr.Eng. Faisal Rahutomo, ST., M.Kom.**

**Skripsi, Program Studi Teknik Informatika, Jurusan Teknologi Informasi, Politeknik Negeri Malang, 2019.**

Plagiarisme adalah tindakan menyalin, mengambil karangan atau pendapat orang lain tanpa adanya izin tertulis dan menjadikanya seolah-olah pendapatnya sendiri. Hal ini masih menjadi fenomena yang sering terjadi pada instansi akademik atau non akademik. Namun pada jurusan Teknologi Informasi Politeknik Negeri Malang belum ada aplikasi yang dapat digunakan untuk mendeteksi plagiarisme.

Berdasarkan permasalahan diatas, maka dibuatlah aplikasi deteksi plagiarisme pada dokumen Tugas Akhir / Skripsi yang bernama *Document plagiarism Detection* (Doristec) dengan menggunakan metode *Longest Common Subsequence (LCS)* dengan membuat modifikasi untuk mencapai hasil yang sesuai dengan perancangan. Aplikasi ini dibuat dengan tujuan untuk mengetahui seberapa besar tingkat plagiarisme pada dokumen Tugas Akhir / Skripsi yang nantinya akan diketahui oleh mahasiswa dan panitia Laporan Akhir dan Skripsi.

Berdasarkan penelitian yang telah dilakukan, Metode *Longest Common Subsequence* dapat digunakan untuk deteksi plagiarisme dengan perbandingan dua atau lebih dokumen. Hal ini dapat menjadi alternatif bagi mahasiswa jurusan Teknologi Informasi Politeknik Negeri Malang dalam melakukan pengujian terhadap penelitiannya dan bagi panitia Laporan Akhir dan Skripsi dapat memonitoring Laporan Akhir dan Skripsi.

**Kata Kunci :** Kemiripan, Plagiarisme, *Longest Common Subsequence,* Teknologi Informasi

# *ABSTRACT*

**Nawawi, Imam.** “*Plagiarism Detection in Essay Documents Based on the Level of Similarity Using the Method* *Longest Common Subsequence*”.

***Advisors*: (1) Putra Prima Arhandi, S.T., M.Kom. (2) Dr.Eng. Faisal Rahutomo, ST., M.Kom.**

***Essay, Informatics Engineering Study Programme, Department of Information Technology, State Polytechnic of Malang, 2019*.**

*Plagiarism is the act of copying, taking essays or opinions of others without written permission and making it as if his own opinion. This is still a phenomenon that often occurs in academic or non academic institutions. But in the Department of Information Technology, Malang State Polytechnic there are no applications that can be used to detect plagiarism.*

*Based on the above problems, a plagiarism detection application was made in the Final Project / Essay document called Document plagiarism Detection (Doristec) using the Longest Common Subsequence (LCS) method by making modifications to achieve results that are in accordance with the design. This application was made with the aim to find out how much the level of plagiarism in the Final Project / Thesis document which will later be known by students and the Final Report and Thesis committee.*

*Based on research that has been done, the Longest Common Subsequence Method can be used to detect plagiarism by comparing two or more documents. This can be an alternative for students majoring in Information Technology at the State Polytechnic of Malang in testing their research and for the Final Report and Thesis committee can monitor the Final Report and Essay..*

***Keywords* :** *Similarities, Plagiarism, Longest Common Subsequence, Information Technology*

# KATA PENGANTAR

Puji Syukur penulis panjatkan kehadirat Allah AWT atas segala rahmat dan hidayah-Nya penulis dapat menyelesaikan skripsi dengan judul “Deteksi Plagiarisme Dokumen Skripsi Menggunakan Metode *Longest Common Subsequence*”.

Penulis menyadari tanpa adanya dukungan dan kerja sama dari berbagai pihak, kegiatan laporan akhir ini tidak akan dapat berjalan baik. Untuk itu, penulis ingin menyampaikan rasa terima kasih kepada:

1. Allah SWT yang telah memberikan kemampuan dan kemudahan selama pengerjaan skripsi ini.
2. Kedua orang tua penulis, Bapak Tu’in Agus Priyanto dan Ibu Kofifa, dan keluarga besar penulis, yang telah memberikan dukungan baik materi maupun non-materi dan doa yang tiada henti sehingga penulis dapat menyelesaikan studi dengan baik.
3. Putra Prima Arhandi, S.T., M.Kom. selaku Dosen Pembimbing I.
4. Dr.Eng. Faisal Rahutomo, ST., M.Kom. selaku Dosen Pembimbing II.
5. Bapak Rudi Ariyanto, ST., MCs. selaku Ketua Jurusan Teknologi Informasi.
6. Imam Fahrur Rozi, ST, MT. selaku Ketua Program Studi Teknik Informatika.
7. Dan seluruh pihak yang telah membantu dan mendukung lancarnya pembuatan skripsi dari awal hingga akhir yang tidak dapat penulis sebutkan satu persatu.

Penulis menyadari bahwa dalam penyusunan skripsi ini masih banyak terdapat kekurangan dan kelemahan yang dimiliki penulis baik itu sistematika pemulisan maupun penggunaan bahasa. Untuk itu penulis mengharapkan saran dan kritik dari berbagai pihak yang bersifat membangun demi penyempurnaan skripsi ini. Semoga skripsi ini berguna bagi pembaca secara umum dan penulis secara khusus.

Akhir kata, penulis ucapkan banyak terima kasih.

Malang, 17 Juni 2019

Penulis

# DAFTAR ISI

**Halaman**

[HALAMAN JUDUL i](#_Toc13060970)

[HALAMAN PENGESAHAN ii](#_Toc13060971)

[HALAMAN PERNYATAAN iii](#_Toc13060972)

[ABSTRAK iv](#_Toc13060973)

[*ABSTRACT* v](#_Toc13060974)

[KATA PENGANTAR vi](#_Toc13060975)

[DAFTAR ISI viii](#_Toc13060976)

[DAFTAR GAMBAR x](#_Toc13060977)

[DAFTAR TABEL xii](#_Toc13060978)

[DAFTAR LAMPIRAN xiv](#_Toc13060979)

[BAB I. PENDAHULUAN 1](#_Toc13060980)

[1.1 Latar Belakang 1](#_Toc13060981)

[1.2 Rumusan Masalah 2](#_Toc13060982)

[1.3 Tujuan Penelitian 2](#_Toc13060983)

[1.4 Batasan Masalah 2](#_Toc13060984)

[1.5 Sistematika Penulisan 3](#_Toc13060985)

[BAB II. LANDASAN TEORI 5](#_Toc13060986)

[2.1 Penelitian Terdahulu 5](#_Toc13060987)

[2.2 Model View Controller (MVC) 6](#_Toc13060988)

[2.3 Metode *Longest Common Subsequence* 7](#_Toc13060989)

[2.4.1 Mendapatkan Panjang LCS 7](#_Toc13060990)

[2.4.2 Mendapatkan LCS 8](#_Toc13060991)

[2.4 Codeigniter 9](#_Toc13060992)

[2.5 Bootstrap 9](#_Toc13060993)

[2.6 MySQL 9](#_Toc13060994)

[2.7 Kalimat 10](#_Toc13060995)

[2.8 Diagram *Use Case* 11](#_Toc13060996)

[2.9 *Framework* 11](#_Toc13060997)

[BAB III. METODOLOGI PENELITIAN 12](#_Toc13060998)

[3.1 Metode Pengambilan Data 12](#_Toc13060999)

[3.2 Metode Pengolahan Data 12](#_Toc13061000)

[3.2.1 Preprocessing 13](#_Toc13061001)

[3.2.2 Pencarian Dokumen Terpanjang 14](#_Toc13061002)

[3.2.3 Longest Common Subsequences (LCS) 14](#_Toc13061003)

[3.3 Metode Pengujian 15](#_Toc13061004)

[BAB IV. ANALISIS DAN PERANCANGAN SISTEM 16](#_Toc13061005)

[4.1 Analisis Sistem 16](#_Toc13061006)

[4.2 Analisis Kebutuhan 17](#_Toc13061007)

[4.2.1 Deskripsi Umum Sistem 18](#_Toc13061008)

[4.2.2 Analisis Kebutuhan Data 18](#_Toc13061009)

[4.2.3 Analisis Pengguna 18](#_Toc13061010)

[4.2.4 Analisis Kebutuhan Fungsional 19](#_Toc13061011)

[4.2.5 Kebutuhan Non Fungsional 19](#_Toc13061012)

[4.3 Perancangan Sistem 20](#_Toc13061013)

[4.3.1 User Flow 20](#_Toc13061014)

[4.3.2 Diagram Use case 21](#_Toc13061015)

[4.3.3 Database Schema 44](#_Toc13061016)

[4.3.4 Work Breakdown Structure 44](#_Toc13061017)

[4.3.5 Upload Dokumen 46](#_Toc13061018)

[4.3.6 Filter Kandidat Pembanding 47](#_Toc13061019)

[4.3.7 Metode Longest Common Subsequence 48](#_Toc13061020)

[4.4 Desain Antar Muka Pengguna 48](#_Toc13061021)

[4.4.1 Halaman Register dan Login 49](#_Toc13061022)

[4.4.2 Tampilan Halaman Mahasiswa 49](#_Toc13061023)

[4.4.3 Tampilan Halaman Panitia 53](#_Toc13061024)

[BAB V. IMPLEMENTASI DAN PENGUJIAN 56](#_Toc13061025)

[5.1 Implementasi Basis Data 56](#_Toc13061026)

[5.1.1 Gambar Tabel Users 57](#_Toc13061027)

[5.1.2 Gambar Tabel Tahun 57](#_Toc13061028)

[5.1.3 Gambar Tabel Antrian\_*dataset* 58](#_Toc13061029)

[5.1.4 Gambar Tabel *Dataset* 59](#_Toc13061030)

[5.1.5 Gambar Tabel Dokumen\_user 59](#_Toc13061031)

[5.1.6 Gambar Tabel Antrian\_dokumen 60](#_Toc13061032)

[5.2 Implementasi Sistem 61](#_Toc13061033)

[5.2.1 Implementasi *Convert* Dokumen ke Text 61](#_Toc13061034)

[5.2.2 Implementasi *Filter* Pada Kandidat Pembanding 63](#_Toc13061035)

[5.2.3 Implementasi Metode LCS Pada Python 65](#_Toc13061036)

[5.2.4 Implementasi *Output* dan *Ranking* Pada Hasil Uji Plagiarisme 70](#_Toc13061037)

[5.3 Implementasi Antar Muka 72](#_Toc13061038)

[5.3.1 Halaman Registrasi dan *Login* 72](#_Toc13061039)

[5.3.2 Halaman Mahasiswa 74](#_Toc13061040)

[5.3.3 Halaman Panitia 79](#_Toc13061041)

[5.4 Pengujian 85](#_Toc13061042)

[5.4.1 Pengujian Fungsionalitas Sistem 85](#_Toc13061043)

[5.4.2 Pengujian Metode 89](#_Toc13061044)

[BAB VI. HASIL DAN PEMBAHASAN 99](#_Toc13061045)

[6.1 Hasil Pengujian 99](#_Toc13061046)

[6.1.1 Hasil Pengujian Kasus Pertama 99](#_Toc13061047)

[6.1.2 Hasil Pengujian Kasus Kedua 102](#_Toc13061048)

[6.1.3 Hasil Pengujian Kasus Ketiga 104](#_Toc13061049)

[6.2 Pembahasan 106](#_Toc13061050)

[6.2.1 Pembahasan Pengujian Validitas 107](#_Toc13061051)

[6.2.2 Pembahasan Pengujian Sistem 107](#_Toc13061052)

[6.2.3 Pembahasan Pengujian Metode 107](#_Toc13061053)

[BAB VII. KESIMPULAN DAN SARAN 110](#_Toc13061054)

[7.1 Kesimpulan 110](#_Toc13061055)

[7.2 Saran 111](#_Toc13061056)

[DAFTAR PUSTAKA 112](#_Toc13061057)

[LAMPIRAN 114](#_Toc13061058)

# DAFTAR GAMBAR

**Halaman**

[Gambar 2. 1 Model View Controller 6](#_Toc13061059)

[Gambar 3. 1 Arsitektur Sistem 13](#_Toc12873005)

[Gambar 4. 1 Diagram Blok Perancangan 16](#_Toc13061060)

[Gambar 4. 2 Gambaran Umum Sistem 20](#_Toc13061061)

[Gambar 4. 3 User Flow Aplikasi 21](#_Toc13061062)

[Gambar 4. 4 Diagram use case pengguna mahasiswa 22](#_Toc13061063)

[Gambar 4. 5 Diagram use case pengguna panitia 23](#_Toc13061064)

[Gambar 4. 6 Database schema dari aplikasi 44](#_Toc13061065)

[Gambar 4. 7 work breakdown structure pada sistem 45](#_Toc13061066)

[Gambar 4. 8 Upload dokumen skripsi dan tugas akhir 46](#_Toc13061067)

[Gambar 4. 9 Flowchart filter kandidat pembanding 47](#_Toc13061068)

[Gambar 4. 10 Flowchart proses LCS 48](#_Toc13061069)

[Gambar 4. 11 Halaman Registrasi dan Login 49](#_Toc13061070)

[Gambar 4. 12 Halaman Dokumen 50](#_Toc13061071)

[Gambar 4. 13 Halaman Upload Dokumen 50](#_Toc13061072)

[Gambar 4. 14 Halaman Uji Plagiasi 51](#_Toc13061073)

[Gambar 4. 15 Halaman Filter 51](#_Toc13061074)

[Gambar 4. 16 Halaman Proses Deteksi 52](#_Toc13061075)

[Gambar 4. 17 Halaman Laporan Hasil Uji Plagiasi 52](#_Toc13061076)

[Gambar 4. 18 Halaman Dashboard 53](#_Toc13061077)

[Gambar 4. 19 Halaman Data Mahasiswa D3 Manajemen Informatika 53](#_Toc13061078)

[Gambar 4. 20 Halaman Data Mahasiswa D4 Teknik Informatika 54](#_Toc13061079)

[Gambar 4. 21 Halaman Dokumen Dataset 54](#_Toc13061080)

[Gambar 4. 22 Halaman Upload Dokumen Pembanding 55](#_Toc13061081)

[Gambar 5. 1 Basis Data Aplikasi 56](#_Toc13061082)

[Gambar 5. 2 Tabel Users 57](#_Toc13061083)

[Gambar 5. 3 Tabel Tahun 58](#_Toc13061084)

[Gambar 5. 4 Tabel Antrian Dataset 58](#_Toc13061085)

[Gambar 5. 5 Tabel Dataset 59](#_Toc13061086)

[Gambar 5. 6 Tabel Dokumen User 60](#_Toc13061087)

[Gambar 5. 7 Tabel Antrian Dokumen 61](#_Toc13061088)

[Gambar 5. 8 Halaman Utama Website 73](#_Toc13061089)

[Gambar 5. 9 Lanjutan Halaman Utama Website 73](#_Toc13061090)

[Gambar 5. 10 Lanjutan Halaman Utama Website 74](#_Toc13061091)

[Gambar 5. 11 Lanjutan Halaman Utama Website 74](#_Toc13061092)

[Gambar 5. 12 Halaman Utama Admin Mahasiswa 75](#_Toc13061093)

[Gambar 5. 13 Halaman Lihat Data 75](#_Toc13061094)

[Gambar 5. 14 Halaman Lihat teks keseluruhan 76](#_Toc13061095)

[Gambar 5. 15 Halaman Edit Data 76](#_Toc13061096)

[Gambar 5. 16 Halaman Upload Dokumen Baru 77](#_Toc13061097)

[Gambar 5. 17 Halaman Uji Plagiasi 78](#_Toc13061098)

[Gambar 5. 18 Halaman Pilih Dokumen Uji 78](#_Toc13061099)

[Gambar 5. 19 Halaman Hasil Filter Kandidat pembanding 79](#_Toc13061100)

[Gambar 5. 20 Halaman Hasil Uji 79](#_Toc13061101)

[Gambar 5. 21 Halaman Dashboard Panitia 80](#_Toc13061102)

[Gambar 5. 22 Halaman Mahasiswa D3 Manajemen Informatika 81](#_Toc13061103)

[Gambar 5. 23 Halaman mahasiswa D4 teknik informatika 81](#_Toc13061104)

[Gambar 5. 24 Halaman mahasiswa D3 yang sudah upload 82](#_Toc13061105)

[Gambar 5. 25 Halaman mahasiswa D4 yang sudah upload 82](#_Toc13061106)

[Gambar 5. 26 Halaman dataset 83](#_Toc13061107)

[Gambar 5. 27 Halaman Lihat Data Lengkap 83](#_Toc13061108)

[Gambar 5. 28 Halaman Lihat Teks Keseluruhan 84](#_Toc13061109)

[Gambar 5. 29 Halaman Edit Data 84](#_Toc13061110)

[Gambar 5. 30 Halaman Tahun 85](#_Toc13061111)

[Gambar 5. 31 Rumus LCS 90](#_Toc13061112)

[Gambar 6. 1 Hasil Pemilihan Kandidat Pembanding Kasus Pertama 100](#_Toc13061113)

[Gambar 6. 2 Hasil Pengujian Pada Kasus Pertama 100](#_Toc13061114)

[Gambar 6. 3 Tampilan Detail Hasil Setiap Kandidat Pembanding 101](#_Toc13061115)

[Gambar 6. 4 Hasil Pemilihan Kandidat Pembanding Kasus Kedua 102](#_Toc13061116)

[Gambar 6. 5 Hasil Pengujian Pada Kasus Kedua 103](#_Toc13061117)

[Gambar 6. 6 Hasil Pemilihan Kandidat Pembanding Kasus Ketiga 104](#_Toc13061118)

[Gambar 6. 7 Hasil Pengujian Pada Kasus Ketiga 105](#_Toc13061119)

[Gambar 6. 8 Tampilan Detail Hasil Setiap Kandidat Pembanding 105](#_Toc13061120)

[Gambar 6. 9 Perhitungan Kasus Pertama Pada Sistem 107](#_Toc13061121)

[Gambar 6. 10 Perhitungan Kasus Kedua Pada Sistem 108](#_Toc13061122)

[Gambar 6. 11 Perhitungan Kasus Ketiga Pada Sistem 109](#_Toc13061123)

# DAFTAR TABEL

**Halaman**

[Tabel 2. 1 Hasil Kalkulasi LCS 8](#_Toc13061124)

[Tabel 2. 2 Hasil Penurutan Balik Untuk Mendapatkan LCS 8](#_Toc13061125)

[Tabel 4. 1 Tabel Kebutuhan Perangkat Keras 19](#_Toc13061126)

[Tabel 4. 2 Tabel Kebutuhan Perangkat Lunak 20](#_Toc13061127)

[Tabel 4. 3 Tabel fungsi aktor use case 23](#_Toc13061128)

[Tabel 4. 4 Definisi Setiap Use Case Mahasiswa 24](#_Toc13061129)

[Tabel 4. 5 Tabel Deskripsi Use Case Login 26](#_Toc13061130)

[Tabel 4. 6 Deskripsi Use Case Logout 27](#_Toc13061131)

[Tabel 4. 7 Deskripsi Use Case Registrasi 28](#_Toc13061132)

[Tabel 4. 8 Deskripsi Use Case Profil 29](#_Toc13061133)

[Tabel 4. 9 Deskripsi Use Case Daftar Dokumen User 30](#_Toc13061134)

[Tabel 4. 10 Deskripsi Use Case Upload Dokumen 31](#_Toc13061135)

[Tabel 4. 11 Definisi Use Case Daftar Dokumen Telah Diuji 32](#_Toc13061136)

[Tabel 4. 12 Deskripsi Use Case Filter Kandidat Pembanding 33](#_Toc13061137)

[Tabel 4. 13 Deskripsi Use Case Proses Uji Plagiarisme 34](#_Toc13061138)

[Tabel 4. 14 Deskripsi Use Case Hasil Uji Plagiarisme 35](#_Toc13061139)

[Tabel 4. 15 Deskripsi Use Case Login 36](#_Toc13061140)

[Tabel 4. 16 Deskripsi Use Case Logout 37](#_Toc13061141)

[Tabel 4. 17 Deskripsi Use Case Registrasi 38](#_Toc13061142)

[Tabel 4. 18 Deskripsi Use Case Profil 39](#_Toc13061143)

[Tabel 4. 19 Deskripsi Use Case Daftar Mahasiswa 40](#_Toc13061144)

[Tabel 4. 20 Deskripsi Use Case Daftar Mahasiswa Telah Upload 41](#_Toc13061145)

[Tabel 4. 21 Deskripsi Use Case Lihat Hasil Deteksi 42](#_Toc13061146)

[Tabel 4. 22 Deskripsi Use Case Upload Dataset 43](#_Toc13061147)

[Tabel 5. 1 Contoh Pemisahan Kata Kunci Berdasarkan Koma 64](#_Toc13061148)

[Tabel 5. 2 Pengujian Sistem Pada Halaman Awal Aplikasi 86](#_Toc13061149)

[Tabel 5. 3 Pengujian Sistem Pada Level Mahasiswa 86](#_Toc13061150)

[Tabel 5. 4 Pengujian Sistem Pada Level Panitia 87](#_Toc13061151)

[Tabel 5. 5 Kasus Pertama Pengujian Metode 90](#_Toc13061152)

[Tabel 5. 6 Perhitungan Manual Pada Studi Kasus Pertama 91](#_Toc13061153)

[Tabel 5. 7 Kasus Kedua Pengujian Metode 92](#_Toc13061154)

[Tabel 5. 8 Perhitungan Manual Pada Studi Kasus Kedua 95](#_Toc13061155)

[Tabel 5. 9 Kasus Ketiga Pengujian Metode 95](#_Toc13061156)

[Tabel 5. 10 Perhitungan Manual Pada Studi Kasus Ketiga 98](#_Toc13061157)

[Tabel 6. 1 Tabel Matrik Recall Precision 99](#_Toc13061158)

[Tabel 6. 2 Recall Precision sesuai skenario pertama 101](#_Toc13061159)

[Tabel 6. 3 Hasil Pengujian Pada Kasus Kedua 102](#_Toc13061160)

[Tabel 6. 4 Recall Precision sesuai skenario kedua 103](#_Toc13061161)

[Tabel 6. 5 Hasil Pengujian Pada Kasus Ketiga 104](#_Toc13061162)

[Tabel 6. 6 Recall Precision sesuai skenario ketiga 106](#_Toc13061163)

[Tabel 6. 7 Perhitungan Kasus Pertama Manual 108](#_Toc13061164)

[Tabel 6. 8 Perhitungan Kasus Kedua Manual 108](#_Toc13061165)

[Tabel 6. 9 Perhitungan Kasus Ketiga Manual 109](#_Toc13061166)

# DAFTAR LAMPIRAN

[Lampiran 1 Script setup.py](#_Toc12873129)

[Lampiran 2 Script lcs.py](#_Toc12873130)

Lampiran 3 Lembar Bimingan Skripsi Pembimbing 1

Lampiran 4 Lembar Bimingan Skripsi Pembimbing 2

Lampiran 5 Form Revisi Penguji 1

Lampiran 6 Form Revisi Penguji 2

[Lampiran 7 Profil Penulis](#_Toc12873131)

BAB I. PENDAHULUAN

1. Latar Belakang

Tugas akhir atau skripsi merupakan penelitian yang harus dilakukan oleh mahasiswa sebagai syarat kelulusan. Tuntutan bagi setiap mahasiswa untuk membuat dan menyelesaikan tugas akhir atau skripsi sebagai bukti dari hasil belajar selama di perguruan tinggi. Membuat laporan tugas akhir atau skripsi adalah langkah yang harus dilakukan oleh mahasiswa untuk mendokumentasikan penelitiannya. Membaca tugas akhir atau skripsi terdahulu diperlukan sebagai referensi bagi mahasiswa dan gambaran dari laporan penelitian. Dalam penulisan laporan penelitian tidak semua mahasiswa mampu menulis berdasarkan argumennya sendiri melainkan dapat mengambil dari laporan penelitian terdahulu, ketika mengambil dari penelitian terdahulu namun tidak dilakukan modifikasi maka akan menimbulkan adanya pelanggaran yaitu plagiarisme. Situasi seperti ini dapat menghambat perkembangan dari aktivitas kreatif mahasiswa maka diperlukan aplikasi yang dapat mendeteksi adanya plagiarisme pada laporan tugas akhir atau skripsi.

Berdasarkan sumber yang penulis amati dalam [1] yang dilakukan oleh Alamsyah, Nur membahas mengenai deteksi plagiarisme pada judul skripsi, metode yang digunakan adalah algoritma Winnowing. Pada penelitian tersebut, aplikasi yang dibangun dapat memberikan informasi berupa hasil plagiarisme pada judul skripsi. Namun, pengujian yang dapat dilakukan pada penelitian tersebut, terbatas pada teks judul skripsi.

Penelitian yang dilakukan oleh Widiastuti, Inta pada tahun 2014 yang berjudul “Aplikasi Pendeteksi Kemiripan Dokumen Menggunakan Algoritma Rabin Karp” [2]. Dalam penelitiannya mengenai deteksi kemiripan antara dua dokumen dengan dilakukan perbandingan pattern pada dokumen teks, metode yang digunakan untuk mendeteksi kemiripan dokumen adalah Algoritma Rabin Karp. Penelitian ini berhasil menciptakan aplikasi deteksi plagiarisme dokumen yang bekerja dengan baik serta memperlihatkan hasil similarity-nya. Namun, pada penelitian ini hanya dapat melakukan pengujian antara dua dokumen.

Dikarenakan pada Jurusan Teknologi Informasi (JTI) belum ada aplikasi deteksi plagiarisme untuk laporan tugas akhir atau skripsi, oleh karena itu dalam skripsi ini penulis membuat sistem deteksi plagiarisme dokumen skripsi menggunakan metode *Longest Common Subsequence* (LCS). Proses deteksi menggunakan perbandingan pada dokumen yang di *upload* mahasiswa dengan *dataset* yang ada pada *database*. Dengan adanya aplikasi ini, diharapkan dapat membantu untuk mendeteksi adanya plagiarisme dokumen tugas akhir atau skripsi.

1. Rumusan Masalah

Berdasarkan latar belakang di atas, maka rumusan masalah yang dapat diambil adalah sebagai berikut:

1. Bagaimana menciptakan aplikasi yang dapat mendeteksi adanya plagiarisme pada dokumen tugas akhir atau skripsi?
2. Bagaimana mendeteksi plagiarisme pada dokumen tugas akhir atau skripsi menggunakan metode *Longest Common Subsequence*?
3. Tujuan Penelitian

Tujuan penelitian ini adalah membuat sistem yang dapat mendeteksi plagiarisme pada dokumen tugas akhir atau skripsi berdasarkan tingkat kesamaan dengan menggunakan metode *Longest Common Subsequence* dan memberikan kemudahan kepada panitia skripsi dalam menentukan penelitian mahasiswa tidak mengandung plagiarisme terhadap penelitian lainnya.

1. Batasan Masalah

Agar skripsi penulis yang berjudul Deteksi Plagiarisme Pada Dokumen Berdasarkan Tingkat Kesamaan Dokumen Dengan Metode *Longest Common Subsequence* dapat berjalan sesuai dengan rencana dan tujuan awal, maka penulis memberikan batasan-batasan masalah yaitu :

1. Deteksi yang dilakukan pada file dokumen bereksensi pdf dan docx/doc.
2. Aplikasi ini menggunakan *dataset* lokal Jurusan Teknologi Informasi.
3. Data yang di uji merupakan skripsi dan tugas akhir.
4. Sistematika Penulisan

Dalam penulisan Laporan Skripsi ini menggunakan tujuh bab dengan sistematika penulisan sebagai berikut:

**BAB I PENDAHULUAN**

Pada bab ini dijelaskan mengenai latar belakang, rumusan masalah, batasan masalah, tujuan dan sistematika penulisan.

**BAB II LANDASAN TEORI**

Pada bab ini berisi teori-teori yang mendasari dan berkaitan dengan masalah perencanaan dan pembuatan sistem yang digunakan untuk memudahkan pemahaman dan pemecahan terhadap masalah yang ada.

**BAB III METODOLOGI PENELITIAN**

Pada bab ini menjelaskan mengenai tahapan yang dilakukan untuk menyelesaikan masalah pada skripsi yang bersumber dari proses dalam perencanaa skripsi. Metode penelitian berisi urauian tentang metode pengmbilan data, metode pengembangan sistem, fasefase pengembangan sistem.

**BAB IV ANALISIS DAN PERANCANGAN**

Bab analisis dan perancangan menjelaskan tentang analisis sistem yang akan di terapkan dan perancangan yang digunakan dan membahas langkah-langkah dalam mengimplementasikan metode pada sistem deteksi plagiarisme menggunakan metode *Longest Common Subsequence*.

**BAB V IMPLEMENTASI DAN PENGUJIAN**

Bab ini membahas mengenai implementasi sistem dan implementasi antar muka dari perancangan yang telah dibahas pada bab iv untuk menciptakan sistem yang sesuai dengan perencanaan.

**BAB VI HASIL DAN PEMBAHASAN**

Pada bab ini membahas mengenai hasil uji coba dari aplikasi yang telah dibuat secara keseluruhan. Mulai dari pengujian sistem sampai penerimaan pengguna.

**BAB VI KESIMPULAN DAN SARAN**

Bab ini berisi kesimpulan dan saran yang diperoleh setelah melakukan analisa, desain, implementasi, dan pengujian terhadap pembuatan aplikasi deteksi plagiarisme.

BAB II. LANDASAN TEORI

Pada bab ini akan diuraikan kajian pustaka dan dasar teori yang mendukung penelitian ini. Dasar teori tersebut diperoleh dari beberapa referensi yang relevan dengan topik yang diangkat dalam penelitian ini. Dalam bab ijni akan dijelaskan mengenai kajian pustaka dan metode *Longest Common Subsequence* (LCS) yang merupakan topik utama diadakannya penelitian ini.

1. Penelitian Terdahulu

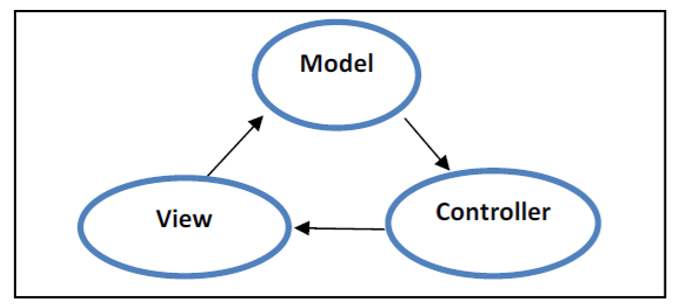
Berdasarkan hasil penelitian terdahulu, ditemukan penelitian sejenis tentang deteksi kemiripan dalam referensi [1] yang membahas mengenai deteksi plagiarisme pada judul skripsi, metode yang digunakan adalah algoritma Winnowing. Pada penelitian tersebut, aplikasi yang dibangun dapat memberikan informasi berupa hasil plagiarisme pada judul skripsi. Namun, pengujian yang dapat dilakukan pada penelitian tersebut, terbatas pada teks judul skripsi.

Penelitian yang dilakukan dalam referensi [2] mengenai deteksi kemiripan antara dua dokumen dengan dilakukan perbandingan pattern pada dokumen teks, metode yang digunakan untuk mendeteksi kemiripan pada dokumen adalah Algoritma Rabin Karp. Penelitian ini berhasil menciptakan aplikasi deteksi plagiarisme yang bekerja dengan baik serta memperlihatkan hasil similarity-nya. Namun, pada penelitian ini hanya dapat melakukan pengujian antara dua dokumen.

Berdasarkan kedua referensi yang sudah dijelaskan tersebut, maka pada penelitian skripsi ini permasalahan yang dibahas adalah pembuatan aplikasi pendeteksi plagiarisme pada dokumen tugas akhir atau skripsi berdasarkan tingkat kesamaan dengan menggunakan metode *Longest Common Subsequence* untuk mencari nilai *similarity* yang terdapat pada satu atau lebih dari satu kandidat pembanding yang dibandingkan. Persamaan penelitian ini dengan kedua referensi penelitian tersebut yaitu sistem pendeteksi kemiripan pada dokumen.

2. Model View Controller (MVC)

Menurut [17], *Model View Controller* (MVC) merupakan *framework* yang menerapkan model MVC, model ini diterapkan guna dapat memberi dampak kedisiplinan dalam menuliskan kode program serta mempermudah dalam pengembangan aplikasi karena telah terstruktur dengan baik, model MVC dibagi menjadi 3 bagian yaitu : model, *view,* dan *controller* seperti pada gambar 2.1.



Gambar 2. 1 Model View Controller

Bagian-bagian tersebut memiliki arti sebagai berikut:

* *Model* merupakan bagian yang berinteraksi dengan *database*.
* *View* merupakan *pages* yang tampil di halaman *web*.
* *Controller* akan mengatur interaksi antar *pages*/ mengkontrol hubungan *view* dan *model*.

Keuntungan MVC adalah sebagai berikut:

* *Division/testability*

*Cotrollers* mengatur interaksi antara *User Interface* dan data (*model*).

* *Flexibility*

Setiap individual *layer* mudah dibuang tanpa mempengaruhi *layer* lainnya karena dapat di *costumize* sehingga *user interface* dapat diganti atau menggunakan *database* yang berbeda.

* *Maintability*

Walaupun dapat di *costumize*, tapi *projects* harus tetap memiliki code yang teratur.

1. Metode *Longest Common Subsequence*

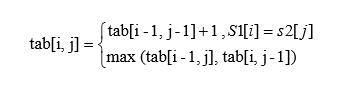
*Longest Common Subsequences* merupakan masalah dalam mencari *subsequence* terpanjang dari beberapa *sequence* (biasanya hanya 2 buah *sequence*), dimana sebuah *subsequence* merupakan sekumpulan kata dari *sequence* yang urutan kemunculannya sama [7]. *Subsequence* dari sebuah *string* A adalah sekumpulan kata yang ada pada A dengan urutan kemunculan kata yang sama.

Dalam rangkaian Z yang merupakan *subsequence* dari X (sebagai *sequence*) dimana X = <x1,x2,…,xm> , jika terdapat urutan menaik <i1,i2,…,in> yang merupakan indeks X untuk semua j=1,2,…,k, yang memenuhi xi=zj. misal pada *common subsequences* dari 2 rangkaian string1 = tadi saya sedang makan dikantin sekolah pada jam istirahat dan string2 = saya sedang mengerjakan tugas seni budaya pada jam istirahat, maka saya sedang, dan pada jam istirahat adalah *common subsequences* dari string1 dan string2. *Longest Common Subsequences* adalah *common subsequences* dari rangkaian hasil yang paling panjang pada 2 rangkaian *string* (string1 dan string2), sehingga *Longest Common Subsequences* yang terpilih adalah pada jam istirahat.

1. Mendapatkan Panjang LCS

Untuk menjabarkan dalam mempermudah dalam menjelaskan, akan dijabarkan sebagai berikut :

Misal *m* adalah panjang string1, *n* adalah panjang string2, dan tab adalah matriks berukuran *m* x *n,* dan tab[*i, j*] adalah panjang LCS dari *subsequences* pertama yang terdiri dari *i* kata pada string1 dengan *subsequences* kedua yang terdiri dari *j* kata pada string2, maka untuk 1 ≤ *i ≤ m* dan 1 ≤ *j ≤ n,* sesuai dengan fungsi rekursif diatas, maka :



Teks yang dibentuk yaitu string1 = tadi saya sedang makan dikantin sekolah pada jam istirahat dan string2 = saya sedang mengerjakan tugas seni budaya pada jam istirahat, tabel matrik yang terbentuk berukuran 9 x 7 yang isinya sebagai berikut :

Tabel 2. 1 Hasil Kalkulasi LCS

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|  |  | tadi | saya | sedang | makan | dikantin | sekolah | pada | jam | istirahat |
| 1 | saya | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 2 | sedang | 0 | 1 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
| 3 | mengerjakan | 0 | 1 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
| 4 | tugas | 0 | 1 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
| 5 | pada | 0 | 1 | 2 | 2 | 2 | 2 | 3 | 3 | 3 |
| 6 | jam | 0 | 1 | 2 | 2 | 2 | 2 | 3 | 4 | 4 |
| 7 | istirahat | 0 | 1 | 2 | 2 | 2 | 2 | 3 | 4 | 5 |

Dan sesuai dengan arti notasi dari setiap tab[*i, j*] yang berisi pada tabel 2.1 bahwa LCS dari string1 dan string2 adalah tab[*m, n*] atau dalam kasus ini adalah ditemukan 5 kata.

1. Mendapatkan LCS

Sedangkan untuk mendapatkan LCS yang dimaksud, kita bisa merunut balik pada tabel 2.1 yang dimulai dari tab[*m, n*]. Dalam runut balik ini, yang kita lakukan sebenarnya hanyalah merunut balik pilihan yang dilakukan pada saat kalkulasi tab[i, j].

Untuk setiap 1 ≤ *i ≤ m* dan 1 *≤ j ≤ n*, jika string1[i] sama dengan string2[j], maka kata itu pasti terdapat pada LCS. Selain itu, cek, dari mana mendapatkan LCS saat itu, lakukan pemilihan yang sama. Lakukan hal tersebut dimulai dari *i = m* dan *j = n*. Setelah hal itu dilakukan maka akan terbentuk tabel di bawah ini. Di mana cell yang dihitamkan menandakan kata yang masuk ke dalam LCS.

Tabel 2. 2 Hasil Penurutan Balik Untuk Mendapatkan LCS

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|  |  | tadi | saya | sedang | makan | dikantin | sekolah | pada | jam | istirahat |
| 1 | Saya | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 2 | Sedang | 0 | 1 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
| 3 | Mengerjakan | 0 | 1 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
| 4 | Tugas | 0 | 1 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
| 5 | Pada | 0 | 1 | 2 | 2 | 2 | 2 | 3 | 3 | 3 |
| 6 | Jam | 0 | 1 | 2 | 2 | 2 | 2 | 3 | 4 | 4 |
| 7 | Istirahat | 0 | 1 | 2 | 2 | 2 | 2 | 3 | 4 | 5 |

Berdasarkan pada tabel 2.2 dapat dilihat bahwa hasil LCS yang ditemukan adalah saya sedang, pada jam istirahat.

1. Codeigniter

CodeIgniter adalah *framework* open source yang powerful yang dibangun menggunakan bahasa pemrograman PHP yang terkenal sebagai PHP *framework* yang mudah dikuasai [15], codeigneter dibangun untuk PHP programmers yang membutuhkan toolkit sederhana dan untuk membuat full-featured web applications. CodeIgniter menggunakan konsep MVC *framework* yang di *design* untuk mempermudah pengguna dan ketika dalam pengembangan aplikasi. Pada penelitian ini codeigniter digunakan untuk membangun aplikasi yang berjudul deteksi plagiarisme pada dokumen skripsi berdasarkan tingkat kesamaan dengan menggunakan metode *Longest Common Subsequence.*

1. Bootstrap

Bootsrap adalah *framework* css yang dapat digunakan untuk membangun tampilan *web* [16]. *Framework* ini menggunakan coding CSS dan JavaScript, namun tetap bisa membuat *website* yang powerfull mengikuti perkembangan *browser*. *Website* yang menggunakan bootstrap akan menjadi *website* yang fleksibel dengan penataan *layout* standard dari bootstrap yang dinamakan grid, sehingga nyaman dan tentu saja cepat baik dalam proses pembuatan maupun ketika *maintenance.* Tidak hanya penataan layoutnya saja yang mudah, bootstrap juga menyediakan komponen-komponen yang siap digunakan untuk mendesain sebuah template.

Komponen yang disediakan sangat banyak dan memiliki beberapa variasi, seperti *dropdowns, button, input, navbar, label, panels, alerts* dan masih banyak lagi yang dapat di *explore* melalui *website* resminya https://getbootstrap.com. Pada penelitian ini desain templatenya menggunakan bootstrap versi 3.

1. MySQL

MySQL adalah *database product* yang bersifat *open source* [19]. MySQL digunakan secara gratis dan hanya menghabiskan sedikit biaya bagi periklanannya namun sangat populer. Tidak seperti *commercial database*, MySQL sangat terjangkau dan mudah digunakan.

Fitur yang terdapat di MySQL antara lain:

* *Openess*

MySQL sangat terbuka dalam setiap hubungan. SQL *dialect* yang digunakan adalah ANSI SQL2 sebagai pembangunnya. *Database engine* menjalankan *platform* yang tidak terhitung, termasuk Windows 2000, Mac OS X, Linux, FreeBSD, dan Solaris. Jika *binary* tidak tersedia untuk *platform*, user dapat mengaksesnya dari *source* untuk di *compile* ke *platform* tersebut.

* *Application Support*

MySQL memiliki API untuk semua bahasa pemrograman. *User* bisa menulis aplikasi *database* yang mengakses MySQL di C, C++, Eiffel, Java, Perl, PHP, Phyton, dan Tcl.

* *Cross-database joins*

User bisa mengkonstruksi MySQL queries yang bisa menggabungkan tabel dari database yang berbeda.

* *Outer join support*

MySQL mendukung kiri dan kanan outer joins menggunakan ANSI dan sintaks ODBC.

1. Kalimat

Menurut [12, pp. 317-319] Kalimat adalah satuan bahasa terkecil, dalam wujud lisan atau tulisan, yang mengungkapkan pikiran yang utuh. Dalam wujud lisan, kalimat diucapkan dengan suara naik turun dan keras lembut, disela jeda dan diakhiri dengan intonasi akhir yang diikuti oleh kesenyapan yang mencegah terjadinya perpaduan ataupun asimilasi bunyi ataupun proses fonologis lainnya.

Dilihat dari segi bentuknya, kalimat dapat dirumuskan sebagai konstruksi sintaksis terbesar yang terdiri atas dua kata atau lebih. Hubungan structural antara kata dan kata, atau kelompok kata dan kelompok kata yang lain, berbeda-beda. Sementara itu, kedudukan tiap kata atau kelompok kata dalam kalimat itu berbeda-beda pula. Ada kata atau kelompok kata yang dapat dihilangkan dengan menghasilkan bentuk yang tetap berupa kalimat (1b), dan ada pula yang tidak seperti pada (2b).

1. a. Ibu pergi ke pasar.

b. Ibu pergi.

1. a. Masalah itu menyangkut masa depan kita.

b. Masalah itu menyangkut.

1. Diagram *Use Case*

*Use case* diagram adalah model diagram dari *UML (Unified Modeling Language)* yang digunakan untuk menggambarkan permintaan secara fungsional yang diharapkan dari sebuah sistem. *Use case* diagram menekankan pada “siapa” melakukan “apa” dalam lingkungan sistem perangkat lunak akan dibangun. *Use-case* diagram sebenarnya terdiri dari dua bagian besar; yang pertama adalah *use case* diagram (termasuk gambar use case *dependencies*) dan *use case* *description*.

*Use case* diagram merupakan gambaran graphical dari beberapa atau semua *actor*, *use case*, dan interaksi [13] diantara komponen-komponen tersebut yang memperkenalkan suatu sistem yang akan dibangun.*Use case* diagram menjelaskan manfaat suatu sistem jika dilihat menurut pandangan orang yang berada di luar sistem. Diagram ini menunjukkan fungsionalitas suatu sistem atau kelas dan bagaimana sistem tersebut berinteraksi dengan dunia luar.

1. *Framework*

Menurut [17], dengan menggunakan framework, kita tidak perlu membuat program dari awal, tetapi kita sudah diberikan library fungsi-fungsi yang sudah diorganisasikan untuk dapat membuat suatu program dengan cepat. Dalam [18], framework merupakan kerangka kerja yang memudahkan programmer untuk membuat sebuah aplikasi sehingga programmer akan lebih mudah melakukan perubahan (customize) terhadap aplikasinya dan dapat memakainya kembali untuk aplikasi lain yang sejenis. Berdasakan penjelasan di atas framework merupakan kerangka kerja yang memudahkan programmer untuk membuat aplikasi dengan library fungsi-fungsi yang sudah diorganisasikan untuk dapat membuat suatu program dengan cepat.

BAB III. METODOLOGI PENELITIAN

Dalam metodologi penelitian ini akan dijelaskan proses tahap-tahapan yang dilakukan untuk membangun aplikasi Deteksi Plagiarisme Dokumen Skripsi Menggunakan Metode *Longest Common Subsequence* dijabarkan sebagai berikut :

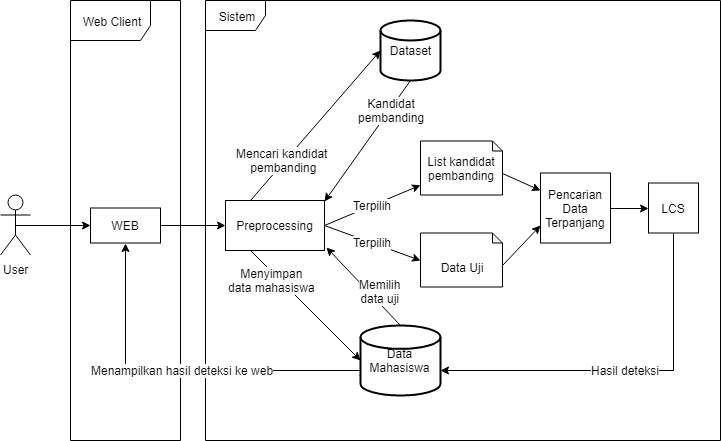
1. Metode Pengambilan Data

Metode pengambilan data yang dilakukan untuk melaksanakan penelitin ini yaitu mencari referensi berupa penelitian terdahulu dan jurnal. penelitian yang berjudul Aplikasi Pendeteksi Kemiripan Dokumen Menggunakan Algoritma Rabin Karp tersebut penulis gunakan sebagai referensi utama. Setelah referensi berhasil dipelajari kemudian mengumpulkan *dataset* dari dokumen – dokumen tugas akhir dan skripsi terdahulu, *dataset* tersebut berupa dokumen word atau pdf skripsi mahasiswa, dari dokumen tersebut akan dilakukan proses konversi dengan mengambil teks dan menghilangkan semua elemen lain seperti gambar dan simbol – simbol, kemudian teks tersebut di simpan kedalam *database* sebagai *dataset*, sehingga penulis memutuskan untuk menggunakan metode *Longest Common Subsequences* (LCS) sebagai metode untuk mencari kesamaan dari dokumen yang di uji. Metode tersebut diambil dari referensi jurnal yang berjudul *Plagiarism detection using document similarity based on distributed representation.*

1. Metode Pengolahan Data

Data yang didapat dari data tugas akhir dan skripsi pada Jurusan Teknologi Informasi merupakan data yang diambil secara manual dari website digilib.jti.polinema.ac.id. data tersebut akan dijadikan sebagai *dataset* sistem, data tersebut perlu dilakukan pengolahan agar mempermudah dalam proses deteksi oleh sistem, pengolahan yang dilakukan yaitu ketika proses *upload* data tugas akhir dan skripsi akan dilakukan *convert* dari dokumen pdf/docx/doc menjadi teks, hal ini dilakukan untuk mengambil teks didalam dokumen, sehingga gambar dan objek-objek lain tidak diambil, setelah berhasil mengambil keseluruhan teks kemudian data disimpan ke *database,* proses *convert* ini juga berlaku ketika proses *upload* dokumen yang akan diuji, proses ini disebut dengan *preprocessing*. Ada juga yang terdapat dalam proses *preprocessing* selain *convert* dokumen adalah *filter* data-data

pada dataset berdasarkan kata kunci yang akan menjadi kandidat pembanding. Tahapan tersebut akan digambarkan pada arsitektur sistem, seperti pada gambar 3.1.



Gambar 3. 1 Arsitektur Sistem

1. Preprocessing

Tahapan dalam *preprocessing* sebagai berikut :

1. *Convert* Dokumen Menjadi Teks

*Convert* dokumen bertujuan untuk mempermudah dalam penyimpanan ke *database* dan meminimalisir penggunaan logika yang terlalu banyak, sehingga proses ini dapat mempersingkat waktu karna ketika proses deteksi dijalankan tidak perlu *convert* dokumen berulang-ulang.

1. *Filter* Kandidat Pembanding

*Filter* Kandidat Pembanding merupakan sebuah fungsi yang terdapat didalam sistem yang bertujuan untuk mencari data-data didalam *dataset,* pencarian dilakukan berdasarkan kata kunci yang sesuai dengan data uji. Sehingga data uji nantinya dilakukan proses perbandingan dengan kandidat pembanding saja, karna data yang tidak memiliki kata kunci yang sama tidak akan diambil menjadi kandidat pembanding. Proses ini memiliki dampak yang besar karna sistem tidak perlu melakukan perbandingan data uji dengan data-data pada *dataset,* serta dapat mempersingkat waktu yang dibutuhkan dalam proses deteksi menggunakan LCS.

1. Pencarian Dokumen Terpanjang

Pencarian dokumen terpanjang merupakan proses yang dilakukan sebelum menjalankan LCS, karena data parameter yang dibutuhkan LCS, parameter 1 harus lebih panjang dari parameter 2, sehingga dibuat sebuah fungsi pencarian data terpanjang untuk mempersiapkan variabel yang akan dimasukkan pada parameter LCS. Sehingga sistem ini benar-benar berjalan secara dinamis.

|  |  |  |
| --- | --- | --- |
| Jika X > Y  Maka  S1 = X  S2 = Y | Atau | Jika Y > X  Maka  S1 = Y  S2 = X |

Keterangan :

S1 = variabel yang menyimpan yang panjang

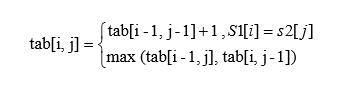
S2 = variabel yang menyimpan lebih pendek

X = data uji

Y = kandidat pembanding

1. Longest Common Subsequences (LCS)

*Longest Common Subsequences* merupakan cara yang digunakan untuk menghitung dan mencari setiap rangkaian kata dalam deteksi plagiarisme. Pencarian ini dilakukan dengan cara membuat tabel matriks yang menyimpan hasil penghitungan LCS, matrik tersebut akan dibuat dengan rumus :



Keterangan :

tab[*i, j*] = panjang LCS yang ditemukan

S1 = data uji

S2 = kandidat pembanding

Karena pada dasar metode LCS hanya mengambil 1 hasil LCS dengan rangkaian kata terpanjang didalam 1 dokumen, maka dibuat sebuah kondisi perulangan untuk menemukan lebih dari 1 rangkaian kata yang plagiat. Kondisi pada perulangan ini dibatasi dengan jumlah kata hasil LCS lebih dari 2 kata, dalam kondisi ini diambil berdasarkan struktur kalimat terbaik adalah kalimat yang memiliki lebih dari 2 rangkaian kata. Dengan adanya kondisi ini akan membuat sistem dapat melakukan pencarian secara berulang-ulang dalam 1 dokumen pembanding, sehingga dalam 1 dokumen pembanding sistem dapat mencari dan mendapatkan lebih dari 1 hasil LCS. Selain itu metode juga dibuat secara dinamis dengan dibuat sebuah kondisi untuk mengetahui jumlah data kandidat pembanding, sehingga sistem dapat berjalan secara berulang-ulang sebanyak data kandidat pembanding.

1. Metode Pengujian

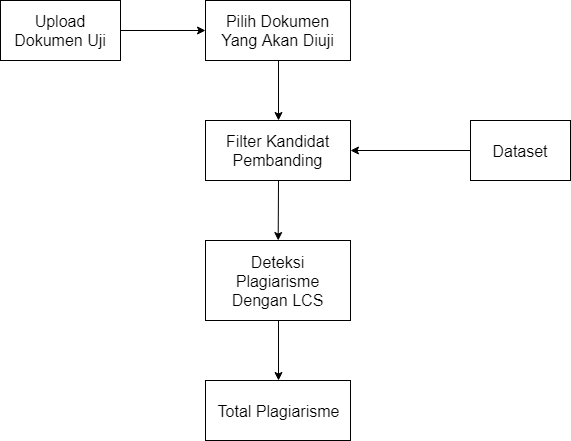
Menguji sistem merupakan tahapan yang dilakukan ketika sistem sudah menjadi perangkat lunak yang siap pakai. Pengujian yang dilakukan menggunakan beberapa cara yaitu *White Box, Black Box*, pengujian arsitektur untuk memastikan perangkat lunak yang sudah jadi sesuai atau tidak dengan perencanaan aplikasi (manual) serta pengujian tingkat keakurasian dari hasil perangkat lunak dengan menggunakan metode pengujian *recall precision*. Jika perangkat lunak sudah sesuai dengan tujuan awal maka sistem sudah dapat digunakan. Namun jika perangkat lunak belum sesuai dengan tujuan awal maka harus mengulang pada tahap dimana letak kesalahan pada aplikasi, kemudian dilakukan pengujian ulang.

BAB IV. ANALISIS DAN PERANCANGAN SISTEM

Pada bab ini, akan dibahas secara detail dan terperinci mengenai analisis dan perancangan deteksi plagiarisme pada dokumen skripsi berdasarkan tingkat kesamaan dengan menggunakan metode *Longest Common Subsequence* yang akan diimplementasikan dengan menerapkan kerangka konsep dan metode penelitian yang telah dijelaskan pada Bab III.

1. Analisis Sistem

Analisis sistem merupakan penjabaran mengenai sistem yang akan dirancang dan dibangun pada penelitian ini yaitu sistem deteksi plagiarisme pada dokumen skripsi berdasarkan tingkat kesamaan dengan menggunakan metode *Longest Common Subsequence* untuk mendapatkan nilai aktual. Berikut ini adalah diagram blok terdapat pada gambar 4.1.



Gambar 4. 1 Diagram Blok Perancangan

Berdasarkan gambar 4.1 mengenai diagram blok perancangan akan dijelaskan tentang proses kerja sistem yang dirancang sebagi berikut:

1. *Upload* dokumen uji adalah proses yang harus dilakukan oleh mahasiswa untuk mengunggah dokumennya kedalam sistem sebelum lanjut ke proses kedua. Dokumen yang telah diunggah akan ditampilkan pada daftar yang disediakan dalam sistem.
2. Pilih dokumen yang akan diuji, adalah proses yang harus dilakukan untuk memilih dokumen yang diinginkan sebagai data yang akan diuji. Dokumen yang dapat dipilih adalah dokumen yang sudah tersedia dalam daftar.
3. *Filter* kandidat pembanding adalah proses pemilihan data pada *dataset* yang sesuai berdasarkan kata kunci yang ada pada dokumen uji. Proses *filter* akan mengambil data-data yang memiliki satu atau lebih kata kunci yang sama. Data yang terpilih akan dijadikan sebagai kandidat pembanding.
4. Deteksi plagiarisme dengan LCS adalah proses dilakukan pendeteksian oleh sistem menggunakan metode LCS dengan membandingkan dokumen uji dengan kandidat pembanding. Waktu yang dibutuhkan dalam satu kali pendeteksian dipengaruhi oleh banyaknya kandidat pembanding terpilih dan jumlah kata dalam satu kandidat pembanding.
5. Total Plagiarisme adalah sebuah tampilan *website* yang menampilkan hasil berupa nilai - nilai plagiarisme seperti banyaknya kata yang ditemukan, total prosentase yang didapatkan dari keseluruhan kandidat pembanding, nilai prosentase yang ditemukan pada tiap-tiap kandidat pembanding, tampilan detail pada setiap *dataset* dengan menampilkan kalimat-kalimat yang diperoleh, sehingga mahasiswa akan mengetahui seberapa besar tingkat plagiarisme yang ditemukan.
6. Analisis Kebutuhan

Dalam analisis kebutuhan akan dilakukan proses penguraian kerangka konseptual yang utuh kedalam bagian-bagian disetiap komponennya dan mengumpulkan kebutuhan dengan dilakukan analisis terhadap permasalahan yang ada, mulai dari mendefinisikan kebutuhan perangkat, sistem, dan data yang berkaitan dengan metode yang digunakan, pada Bab III telah dijelaskan mengenai metode yang digunakan untuk menyelesaikan permasalahan, maka pada bab ini dilakukan penjabaran dari tiap tahapannya, berikut ini adalah penjelasan dari tiap-tiap tahap.

1. Deskripsi Umum Sistem

Deteksi plagiarisme pada dokumen skripsi berdasarkan tingkat kesamaan dengan menggunakan metode *Longest Common Subsequence* digunakan untuk membantu panitia skripsi dalam menentukan penelitian mahasiswa tidak mengandung plagiarisme terhadap penelitian lainnya.

Deteksi akan dilakukan oleh sistem ketika mahasiswa telah *upload* dokumennya lalu masuk dalam proses filter dengan memilih penelitian yang akan dilakukan pengujian kemudian sistem memberikan respon berupa daftar kandidat pembanding terpilih untuk masuk pada proses pengujian plagiarisme, pada proses pengujian di implementasikan metode *Longest Common Subsequence* untuk mendapatkan jawaban yang benar.

1. Analisis Kebutuhan Data

Analisis kebutuhan data bertujuan untuk dapat mengetahui tentang kebutuhan data yang diperlukan oleh sistem baik data masukan maupun data yang harus ada didalam sistem sebagai *dataset* agar hasil keluaran sesuai dengan tujuan awal dibuatnya sistem ini.

1. Analisis Pengguna

Berdasarkan deskripsi umum sistem yang dibuat, maka dapat diketahui pengguna dari deteksi plagiarisme pada dokumen skripsi berdasarkan tingkat kesamaan dengan menggunakan metode *Longest Common Subsequence* yaitu :

1. Panitia Skripsi

Panitia skripsi adalah orang yang memonitoring terhadap penelitian mahasiswa yang telah melakukan proses pengujian terhadap penelitiannya, panitia memiliki hak untuk menerima dan menolak (agar dilakukan perbaikan) penelitian yang memiliki tingkat plagiarisme tinggi.

1. Mahasiswa

Mahasiswa adalah orang yang melakukan penelitian dan submit penelitiannya kepada sistem deteksi plagiarisme pada dokumen skripsi berdasarkan tingkat kesamaan dengan menggunakan metode *Longest Common Subsequence* kemudian melakukan pengujian terhadap penelitiannya, sehingga dapat mengetahui apakah penelitiannya mengandung plagiarisme dengan penelitian lainnya.

1. Analisis Kebutuhan Fungsional

Kebutuhan fungsional adalah kebutuhan yang berupa proses-proses yang harus dimiliki oleh sistem. Hal ini pengguna dapat menggunakan layanan-layanan pada sistem, antara lain :

1. Sistem dapat melakukan *load dataset* dari *database.*
2. Sistem dapat melakukan convert pdf/docx dan menyimpan ke *database.*
3. Sistem dapat melakukan *filter* berdasarkan kata kunci.
4. Sistem dapat melakukan perhitungan LCS.
5. Sistem dapat menyimpan hasil proses deteksi ke *database*.
6. Sistem dapat menampilkan kembali hasil deteksi dari *database*.
7. Kebutuhan Non Fungsional

Kebutuhan non fungsional adalah kebutuhan yang tidak secara langsung terkait dengan fitur yang ada dalam sistem. Pada sistem ini terdapat dua kebutuhan non-fungsional yaitu :

1. Kebutuhan Perangkat Keras

Spesifikasi perangkat keras yang digunakan penulis dalam pembuatan sistem ini adalah sebagai berikut ditunjukkan pada tabel 4.1:

Tabel 4. 1 Tabel Kebutuhan Perangkat Keras

|  |  |  |
| --- | --- | --- |
| **No** | **Nama Komponen** | **Spesifikasi** |
| 1 | *Processor* | Intel(R) Core(TM) i7-4720HQ CPU @ 2.60GHz~2.59GHz |
| 2 | *Memory* (RAM) | 8GB |
| 3 | Graphic | NVIDIA Geforce GTX 950M - 2GB |
| 4 | *Storage* | SSD 250GB |
| 5 | HDD 1TB |

1. Kebutuhan Perangkat Lunak

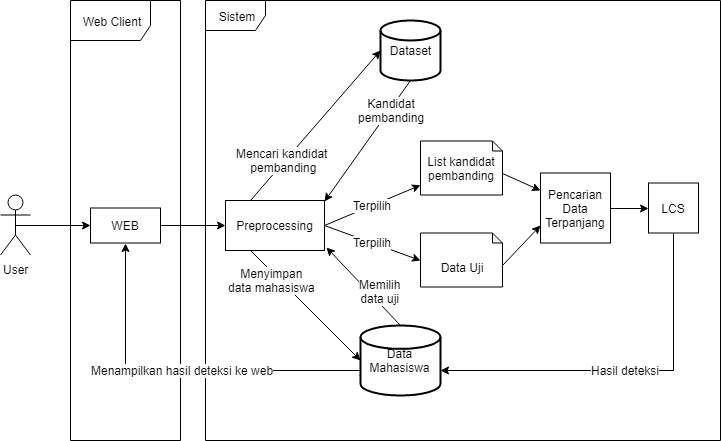
Spesifikasi perangkat lunak yang digunakan penulis dalam menunjang kebutuhan pada pembuatan sistem ini adalah sebagai berikut ditunjukkan pada tabel 4.2 :

Tabel 4. 2 Tabel Kebutuhan Perangkat Lunak

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Nama Komponen** | **Spesifikasi** | **Versi** |
| 1 | *Operating System* | Windows 10 Pro | 1809 |
| 2 | *Tools* Pemrograman Web | Atom | 1.36.1 |
| Wingware IDE 101 | 6.1.5-1 |
| 3 | Web *Browser* | Google Chrome | 69.0.3497.100 |
| Microsoft Edge | 76.0.167.1 |
| 4 | *Local Server* | Xampp | 3.2.2 |
| 5 | Bahasa Pemrograman Web | PHP | 7.1.1 |
| Python | 3.7.2 |
| 6 | DBMS | MySQL-MariaDB | 10.1.21 |

1. Perancangan Sistem

Perancangan sistem adalah proses pembuatan sistem yang diperoleh berdasarkan dari hasil analisis kebutuhan sistem. Perancangan sistem ini memuat uraian-uraian yang berupa *input*, proses dan *output*. Dengan gambaran umum sistem seperti pada gambar 4.2.



Gambar 4. 2 Gambaran Umum Sistem

1. User Flow

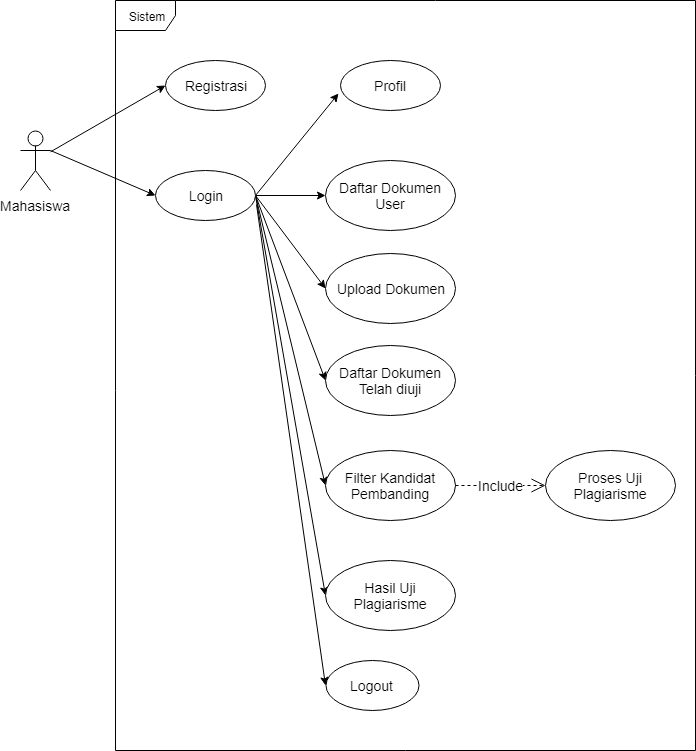
*User flow* adalah bentuk alur yang harus diikuti dan dilakukan oleh pengguna untuk dapat menyelesaikan tugas. *User* *flow* menunjukkan awal titik masuk pada sistem hingga menuju hasil akhir atau tindakan akhir yang dapat dilakukan oleh pengguna. Pada sistem ini digambarkan sebuah alur pada *web client* mahasiswa, ketika mahasiswa mengunjungi domain utama maka langsung diarahkan pada halaman main. Halaman main memiliki 2 *form* yang digunakan untuk registrasi dan *login*, pada kondisi pengguna melakukan *login* sistem akan otomatis memeriksa pengguna yang akan masuk telah terdaftar didalam sistem atau belum, jika belum maka pengguna harus registrasi terlebih dahulu, jika telah terdaftar maka akan dapat masuk kedalam sistem dan dapat menggunakan fitur-fitur yang disediakan. berikut ini ditunjukkan pada gambar 4.3:



Gambar 4. 3 User Flow Aplikasi

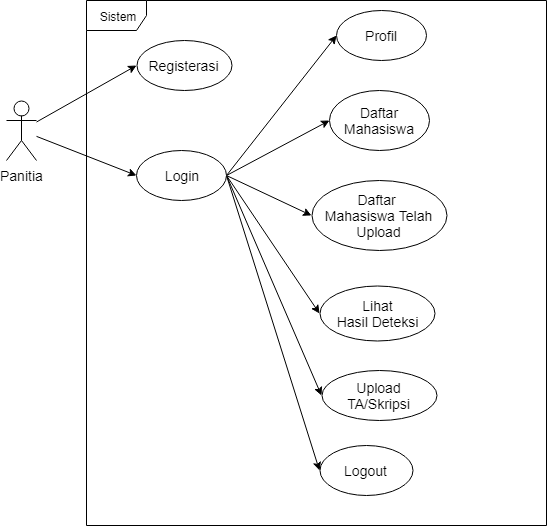
1. Diagram Use case

*Use case* adalah diagram yang menggambarkan tentang keterkaitan satu sama lain antara *use case* dan aktor sehingga membentuk sebuah sistem yang teratur, serta dapat dengan mudah dalam melakukan pengawasan oleh pengguna atau disebut aktor. Dalam perancangan diagram *use case* ini memiliki 2 aktor yang berbeda yaitu mahasiswa dan panitia. Pada *use case* dengan aktor panitia memiliki beberapa komponen *use case* yang berbeda dari aktor mahasiswa, aktor mahasiswa dapat ditunjukkan pada gambar 4.4 yang terhubung dengan bebrapa *use case*. *Use case* dengan aktor mahasiswa ini memiliki beberapa berbedaan dengan *use case* aktor panitia, karena pada *use case* ini memiliki bagian *filter* kandidat pembanding dan proses uji plagiarisme.



Gambar 4. 4 Diagram use case pengguna mahasiswa

Selain aktor mahasiswa juga terdapat aktor panitia yang ditunjukkan pada gambar 4.5 yang terhubung dengan beberapa *use case*, *use case* aktor panitia juga memiliki perbedaan dengan *use case* mahasiswa yaitu pada daftar mahasiswa, daftar mahasiswa telah *upload,* upload dataset (TA/Skripsi).



Gambar 4. 5 Diagram use case pengguna panitia

Deskripsi aktor pada sistem deteksi plagiarisme pada dokumen skripsi berdasarkan tingkat kesamaan dengan menggunakan metode *longest common subsequenc* disajikan pada tabel 4.3.

Tabel 4. 3 Tabel fungsi aktor use case

|  |  |  |
| --- | --- | --- |
| **No** | **Aktor** | **Kebutuhan** |
| 1 | Panitia | Pengguna yang bertugas dalam mengelola *dataset,* menentukan tahun, dan monitoring hasil deteksi. |
| 2 | Mahasiswa | Pengguna yang melakukan *upload* dokumen uji dan melakukan pengujian skripsi atau tugas akhir |

Selanjutnya, mendefinisikan *use case* dijelaskan dengan detail dan terperinci sebagaimana telah disajikan pada tabel 4.4. Pada tabel tersebut akan dijelaskan fitur-fitur dan menu yang ada pada aplikasi yang dibangun.

Tabel 4. 4 Definisi Setiap Use Case Mahasiswa

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Nama *Use Case*** | **Deskripsi** | **Pengguna** |
| 1 | *Login* | Berisi form yang terdiri dari *email* dan *password* digunakan untuk masuk kedalam sistem. | Mahasiswa |
| 2 | *Logout* | Berisi fungsi yang akan menampilkan *pop up* digunakan untuk keluar dari sistem. | Mahasiswa |
| 3 | Registrasi | Berisi form yang terdiri dari nama, *email*, jenis kelamin, tanggal lahir, *password*, alamat, program studi, nomor hp, level pengguna yang akan disimpan kedalam *database*. | Mahasiswa |
| 4 | Profil | Berisi data lengkap dari profil pengguna yang sudah tersimpan sebelumnya pada *database.* | Mahasiswa |
| 5 | Daftar Dokumen *User* | Berisi data - data dari dokumen yang telah di *upload* sebelumnya oleh pengguna dan sudah tersimpan pada *database.* | Mahasiswa |
| 6 | *Upload* Dokumen | Berisi form yang terdisi dari judul, tahun, *keyword*, program studi, ringkasan, file penelitian yang akan disimpan kedalam *database*. | Mahasiswa |

Lanjutan tabel 4.4 Definisi setiap *use case* mahasiswa

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Nama *Use Case*** | **Deskripsi** | **Pengguna** |
|  | Daftar Dokumen telah diuji | Berisi data - data dari dokumen yang telah di lakukan pengujian plagiarisme oleh sistem berdasarkan permintaan pengguna dan sudah disimpan pada *database* ketika pengujian selesai. | Mahasiswa |
| 8 | *Filter* Kandidat Pembanding | Berisi form pilihan data - data dari dokumen yang telah tersimpan pada *database.* | Mahasiswa |
| 9 | Proses Uji Plagiarisme | Berisi data - data dari proses *filter* kandidat pembanding dan data dokumen uji, tahap ini dilakukan proses uji plagiarisme dengan hasil di simpan pada *database.* | Mahasiswa |
| 10 | Hasil Uji Plagiarisme | Berisi data - data dari proses *uji* plagiarisme yang telah tersimpan pada *database.* | Mahasiswa |

Setelah melakukan pendefinisian terhadap *use case* secara satu-persatu, Langkah selanjutnya adalah mendeskripsikan pada setiap *use case* mahasiswa secara lengkap, sehingga dapat lebih mudah dipahami. Deskripsi dari tiap-tiap *use case* akan disajikan pada tabel-tabel berikut ini, yaitu ditampilkan pada tabel 4.5, tabel 4.6, tabel 4.7, tabel 4.8, tabel 4.9, tabel 4.10, tabel 4.11, tabel 4.12, tabel 4.13, tabel 4.14.

Tabel 4. 5 Tabel Deskripsi Use Case Login

|  |  |  |
| --- | --- | --- |
| *Use Case Name:*  *Login* | ID: UC.01  Siklus1 | *Importance Level:*  *High* |
| *Primary Actor:*  Mahasiswa | *Use Case Type:* | |
| *Stakeholder and Interest:*  Mahasiswa masuk kedalam sistem untuk mengolah data yang ada. | | |
| *Brief Description:*  Menjelaskan proses inisialisasi pengguna | | |
| *Trigger:* Mahasiswa masuk kedalam sistem untuk mengelola data dokumen yang ada, *upload* dokumen baru, uji dokumen.  *Type: internal* | | |
| *Relationship*:  *Association:* Mahasiswa  *Include*: | | |
| *Extend:*  *Generalization:* | | |
| *Normal flow of event:*   1. Mahasiswa memasukkan *email* dan *password* dan menekan tombol masuk. 2. Jika data yang dimasukkan sesuai, maka *login user* diterima.   Sistem menampilkan halaman yang sesuai dengan hak akses yang diberikan. | | |
| Subflows:  1a. Jika *email* dan *password* yang diinputkan kosong, maka akan muncul alert “*Email* atau *Password* salah”  1b. Jika salah satu *email* atau *password* yang diinputkan salah, maka akan muncul alert “*Email* atau *Password* salah” | | |

Tabel 4. 6 Deskripsi Use Case Logout

|  |  |  |
| --- | --- | --- |
| *Use Case Name:*  *Logout* | ID: UC.02  Siklus2 | *Importance Level:*  *High* |
| *Primary Actor:*  Mahasiswa | *Use Case Type:* | |
| *Stakeholder and Interest:*  Mahasiswa keluar dari sistem setelah selesai melakukan aktivitas yang diperlukan. | | |
| *Brief Description:*  Menjelaskan proses inisialisasi pengguna | | |
| *Trigger:* Mahasiswa kembali ke halaman utama.  *Type: internal* | | |
| *Relationship*:  *Association:* Mahasiswa  *Include*: *Login* | | |
| *Extend:*  *Generalization:* | | |
| *Normal flow of event:*   1. Mahasiswa menekan tombol *Logout* untuk keluar dari sistem. 2. Sistem akan menampilkan halaman yang berisi *form login* dan registrasi. | | |
| Subflows:  1a. Ketika tombol *logout* ditekan, akan muncul *pop up* konfirmasi yang berisi “keluar atau tidak”  2a. Setelah menekan tombol Ya pada *pop up* konfirmasi, maka akan muncul halaman depan yang berisi informasi, *form login,* dan *form* registrasi. | | |

Tabel 4. 7 Deskripsi Use Case Registrasi

|  |  |  |
| --- | --- | --- |
| *Use Case Name:*  Registrasi | ID: UC.03  Siklus3 | *Importance Level:*  *High* |
| *Primary Actor:*  Mahasiswa | *Use Case Type:* | |
| *Stakeholder and Interest:*  Mahasiswa mengisi *form* registrasi berdasarka data pribadi sesuai dengan identitas KTM atau KTP. | | |
| *Brief Description:*  Menjelaskan proses inisialisasi pengguna | | |
| *Trigger:* Mahasiswa mengisi *form* registrasi untuk disimpan ke *database*.  *Type: internal* | | |
| *Relationship*:  *Association:* Mahasiswa  *Include*: | | |
| *Extend:*  *Generalization:* | | |
| *Normal flow of event:*   1. Mahasiswa mengisi *form* registrasi berdasarka data pribadi. 2. Mahasiswa menekan tombol simpan untuk menyimpan ke *database*. | | |
| Subflows:  1a. Mahasiswa wajib mengisi *form* nama, *email*, nomor hand phone, password sesuai dengan identitas priadi, seperti KTM atau KTP.  2a. Ketika salah satu *form input* pada nomor 1a tidak terisi, maka proses simpan tidak akan dijalankan.  2b. Ketika *form* telah terisi semua termasuk pada nomor 1a, maka proses simpan ke *database* akan dilakukan. | | |

Tabel 4. 8 Deskripsi Use Case Profil

|  |  |  |
| --- | --- | --- |
| *Use Case Name:*  Profil | ID: UC.04  Siklus4 | *Importance Level:*  *High* |
| *Primary Actor:*  Mahasiswa | *Use Case Type:* | |
| *Stakeholder and Interest:*  Untuk melihat data yang sudah tersimpan di *database*, dan *form* *reset password*. | | |
| *Brief Description:*  Menjelaskan proses Mahasiswa melihat data yang ada | | |
| *Trigger:* Mahasiswa melihat data yang ada, dan *form* *reset password.*  *Type: internal* | | |
| *Relationship*:  *Association:* Mahasiswa  *Include*: *Login* | | |
| *Extend:*  *Generalization:* | | |
| *Normal flow of event:*   1. Ketika Mahasiswa menekan tombol profil, makan otomatis akan ditampilkan halaman berisi data pribadi yang tersimpan pada *database*. 2. Mahasiswa dapat melihat *form* *reset* *password.* 3. Ketika mahasiswa menekan tombol *edit* profil, maka ditampilkan *form* *edit.* | | |
| Subflows:  2a. Ketika halaman profil ditampilkan, mahasiswa juga dapat melihat *form* *reset password.*  2b. Untuk mengganti *password* harus mengisi data *password* lama, *password* baru, dan ulangi *password* baru pada *form* ganti *password,* kemudian tekan tombol *Reset.* Jika *form reset password* tidakterisi dengan benar, maka proses *reset password* dibatalkan*.*  3c. Mahasiswa menekan tombol *edit* profil, akan diampilkan *form edit* sehingga mahasiswa dapat mengubah dan menyimpan kembali ke *database.* | | |

Tabel 4. 9 Deskripsi Use Case Daftar Dokumen User

|  |  |  |
| --- | --- | --- |
| *Use Case Name:*  Daftar Dokumen *User* | ID: UC.05  Siklus5 | *Importance Level:*  *High* |
| *Primary Actor:*  Mahasiswa | *Use Case Type:* | |
| *Stakeholder and Interest:*  Untuk melihat data yang sudah tersimpan di *database*. | | |
| *Brief Description:*  Menjelaskan proses Mahasiswa melihat data yang ada | | |
| *Trigger:* Mahasiswa melihat data yang ada*.*  *Type: internal* | | |
| *Relationship*:  *Association:* Mahasiswa  *Include*: *Login* | | |
| *Extend:*  *Generalization:* | | |
| *Normal flow of event:*   1. Ketika Mahasiswa berhasil *login,* maka otomatis akan tampil data-data dokumen yang telah tersimpan pada *database*. 2. Ketika mahasiswa menekan menu dokumen saya, maka otomatis akan tampil data-data dokumen yang telah tersimpan pada *database.* | | |
| Subflows:  *.* | | |

Tabel 4. 10 Deskripsi Use Case Upload Dokumen

|  |  |  |
| --- | --- | --- |
| *Use Case Name:*  *Upload* Dokumen | ID: UC.06  Siklus6 | *Importance Level:*  *High* |
| *Primary Actor:*  Mahasiswa | *Use Case Type:* | |
| *Stakeholder and Interest:*  Mahasiswa mengisi *form upload* dokumen untuk menambah data pada *database*. | | |
| *Brief Description:*  Menjelaskan proses inisialisasi pengguna. | | |
| *Trigger:* Mahasiswa menampilkan data-data yang ada*.*  *Type: internal* | | |
| *Relationship*:  *Association:* Mahasiswa  *Include*: *Login* | | |
| *Extend:*  *Generalization:* | | |
| *Normal flow of event:*   1. Mahasiswa menekan menu dokumen saya*,* maka otomatis akan tampil data-data dokumen yang telah tersimpan pada *database* dan tombol *upload* dokumen baru. 2. Mahasiswa mengisi *form upload,* kemudian menekan tombol simpan, data otomatis dimasukkan *database* dan ditampilkan kembali pada halaman dokumen saya*.* | | |
| Subflows:  *.* | | |

Tabel 4. 11 Definisi Use Case Daftar Dokumen Telah Diuji

|  |  |  |
| --- | --- | --- |
| *Use Case Name:*  Daftar Dokumen Telah Diuji | ID: UC.07  Siklus7 | *Importance Level:*  *High* |
| *Primary Actor:*  Mahasiswa | *Use Case Type:* | |
| *Stakeholder and Interest:*  Untuk melihat data yang sudah tersimpan di *database*. | | |
| *Brief Description:*  Menjelaskan proses Mahasiswa melihat data yang ada | | |
| *Trigger:* Mahasiswa melihat data yang ada*.*  *Type: internal* | | |
| *Relationship*:  *Association:* Mahasiswa  *Include*: *Login* | | |
| *Extend:*  *Generalization:* | | |
| *Normal flow of event:*   1. Ketika Mahasiswa selesai melakukan proses uji*,* maka otomatis akan tampil data-data dokumen yang telah tersimpan pada *database*. | | |
| Subflows:  *.* | | |

Tabel 4. 12 Deskripsi Use Case Filter Kandidat Pembanding

|  |  |  |
| --- | --- | --- |
| *Use Case Name:*  *Filter* Kandidat Pembanding | ID: UC.08  Siklus8 | *Importance Level:*  *High* |
| *Primary Actor:*  Mahasiswa | *Use Case Type:* | |
| *Stakeholder and Interest:*  Untuk pemilihan data-data yang sudah tersimpan di *database*. | | |
| *Brief Description:*  Menjelaskan proses Mahasiswa memilih calon data uji dan data-data dari *dataset* yang ada | | |
| *Trigger:* Mahasiswa menampilkan data hasil *filter dataset.*  *Type: internal* | | |
| *Relationship*:  *Association:* Mahasiswa  *Include*: *Login* | | |
| *Extend:*  *Generalization:* | | |
| *Normal flow of event:*   1. Mahasiswa menekan tombol uji dokumen*,* maka otomatis akan tampil pilihan untuk menentukan dokumen yang ingin diuji berdasarkan data-data dokumen yang telah tersimpan pada *database*. 2. Mahasiswa menekan tombol proses jika sudah menentukan dokumen pilihan. | | |
| Subflows:  2a. Ketika tombol proses ditekan, akan memanggil fungsi *filter dataset* untuk mengambil *dataset* yang sesuai dengan kata kunci pada dokumen uji.  2b. Ketika proses *filter* selesai, akan ditampilkan data-data yang telah didapatkan.*.* | | |

Tabel 4. 13 Deskripsi Use Case Proses Uji Plagiarisme

|  |  |  |
| --- | --- | --- |
| *Use Case Name:*  Proses Uji Plagiarisme | ID: UC.09  Siklus9 | *Importance Level:*  *High* |
| *Primary Actor:*  Mahasiswa | *Use Case Type:* | |
| *Stakeholder and Interest:*  Mahasiswa pemilihan data-data dari proses *filter* kandidat pembanding yang sudah tersimpan di *database*. | | |
| *Brief Description:*  Menjelaskan proses inisialisasi sistem. | | |
| *Trigger:* Mahasiswa menampilkan data hasil Uji*.*  *Type: internal* | | |
| *Relationship*:  *Association:* Mahasiswa  *Include*: *Login* | | |
| *Extend:*  *Generalization:* | | |
| *Normal flow of event:*   1. Mahasiswa menekan tombol uji sekarang*,* maka otomatis akan melakukan proses uji plagiarisme dan hasilnya dimasukkan kembali ke *database*. | | |
| Subflows:  1a. Ketika tombol uji sekarang ditekan, akan memanggil fungsi uji plagiarisme. Maka otomatis akan mengambil data uji dan kandidat pembanding untuk pencarian adanya plagiarisme. Proses ini akan dilakukan berulang-ulang hingga tidak ditemukan adanya plagiarisme.  2b. Ketika proses ujiselesai, hasil akan disimpan kembali ke *database.* | | |

Tabel 4. 14 Deskripsi Use Case Hasil Uji Plagiarisme

|  |  |  |
| --- | --- | --- |
| *Use Case Name:*  Hasil Uji Plagiarisme | ID: UC.010  Siklus10 | *Importance Level:*  *High* |
| *Primary Actor:*  Mahasiswa | *Use Case Type:* | |
| *Stakeholder and Interest:*  Untuk melihat data-data yang sudah tersimpan di *database*. | | |
| *Brief Description:*  Menjelaskan proses Mahasiswa melihat data yang ada | | |
| *Trigger:* Mahasiswa melihat data yang ada*.*  *Type: internal* | | |
| *Relationship*:  *Association:* Mahasiswa  *Include*: *Login* | | |
| *Extend:*  *Generalization:* | | |
| *Normal flow of event:*   1. Mahasiswa memilih menu uji plagiasi, maka otomatis akan tampil data-data hasil uji yang telah tersimpan pada *database*. 2. Mahasiswa memilih dan menekan tombol lihat hasil, maka otomatis akan menampilkan data secara lengkap berdasarkan data yang dipilih. | | |
| Subflows:  2a. Ketika tombol lihat hasil ditekan, maka otomatis akan menampilkan data lengkap dari hasil uji plagiarisme yang menujukkan berapa *dataset* yang diujikan, berapa kata yang ditemukan.  *.* | | |

Berikutnya mendeskripsikan pada setiap *use case* panitia secara lengkap. Deskripsi dari tiap-tiap *use case* akan disajikan pada tabel-tabel berikut ini, yaitu ditampilkan pada tabel 4.15, tabel 4.16, tabel 4.17, tabel 4.18, tabel 4.19, tabel 4.20, tabel 4.21, tabel 4.22.

Tabel 4. 15 Deskripsi Use Case Login

|  |  |  |
| --- | --- | --- |
| *Use Case Name:*  *Login* | ID: UC.11  Siklus11 | *Importance Level:*  *High* |
| *Primary Actor:*  Panitia | *Use Case Type:* | |
| *Stakeholder and Interest:*  Panitia masuk kedalam sistem untuk mengolah data yang ada. | | |
| *Brief Description:*  Menjelaskan proses inisialisasi pengguna | | |
| *Trigger:* Panitia masuk kedalam sistem untuk mengelola data-data yang ada.  *Type: internal* | | |
| *Relationship*:  *Association:* Panitia  *Include*: | | |
| *Extend:*  *Generalization:* | | |
| *Normal flow of event:*   1. Mahasiswa memasukkan *email* dan *password* dan menekan tombol masuk. 2. Jika data yang dimasukkan sesuai, maka *login user* diterima.   Sistem menampilkan halaman yang sesuai dengan hak akses yang diberikan. | | |
| Subflows:  1a. Jika *email* dan *password* yang diinputkan kosong, maka akan muncul alert “*Email* atau *Password* salah”  1b. Jika salah satu *email* atau *password* yang diinputkan salah, maka akan muncul alert “*Email* atau *Password* salah” | | |

Tabel 4. 16 Deskripsi Use Case Logout

|  |  |  |
| --- | --- | --- |
| *Use Case Name:*  *Logout* | ID: UC.12  Siklus12 | *Importance Level:*  *High* |
| *Primary Actor:*  Panitia | *Use Case Type:* | |
| *Stakeholder and Interest:*  Panitia keluar dari sistem setelah selesai melakukan aktivitas yang diperlukan. | | |
| *Brief Description:*  Menjelaskan proses inisialisasi pengguna | | |
| *Trigger:* Panitia kembali ke halaman utama.  *Type: internal* | | |
| *Relationship*:  *Association:* Panitia  *Include*: *Login* | | |
| *Extend:*  *Generalization:* | | |
| *Normal flow of event:*   1. Panitia menekan tombol *Logout* untuk keluar dari sistem. 2. Sistem akan menampilkan halaman yang berisi *form login* dan registrasi. | | |
| Subflows:  1a. Ketika tombol *logout* ditekan, akan muncul *pop up* konfirmasi yang berisi “keluar atau tidak”  2a. Setelah menekan tombol Ya pada *pop up* konfirmasi, maka akan muncul halaman depan yang berisi informasi, *form login,* dan *form* registrasi. | | |

Tabel 4. 17 Deskripsi Use Case Registrasi

|  |  |  |
| --- | --- | --- |
| *Use Case Name:*  Registrasi | ID: UC.13  Siklus13 | *Importance Level:*  *High* |
| *Primary Actor:*  Panitia | *Use Case Type:* | |
| *Stakeholder and Interest:*  Panitia mengisi *form* registrasi berdasarka data pribadi sesuai dengan identitas KTP. | | |
| *Brief Description:*  Menjelaskan proses inisialisasi pengguna | | |
| *Trigger:* Panitia mengisi *form* registrasi untuk disimpan ke *database*.  *Type: internal* | | |
| *Relationship*:  *Association:* Panitia  *Include*: | | |
| *Extend:*  *Generalization:* | | |
| *Normal flow of event:*   1. Panitia mengisi *form* registrasi berdasarka data pribadi. 2. Panitia menekan tombol simpan untuk menyimpan ke *database*. | | |
| Subflows:  1a. Panitia wajib mengisi *form* nama, *email*, nomor hand phone, password sesuai dengan identitas priadi, seperti KTM atau KTP.  2a. Ketika salah satu *form input* pada nomor 1a tidak terisi, maka proses simpan tidak akan dijalankan.  2b. Ketika *form* telah terisi semua termasuk pada nomor 1a, maka proses simpan ke *database* akan dilakukan. | | |

Tabel 4. 18 Deskripsi Use Case Profil

|  |  |  |
| --- | --- | --- |
| *Use Case Name:*  Profil | ID: UC.14  Siklus14 | *Importance Level:*  *High* |
| *Primary Actor:*  Panitia | *Use Case Type:* | |
| *Stakeholder and Interest:*  Untuk melihat data yang sudah tersimpan di *database*, dan *form* *reset password*. | | |
| *Brief Description:*  Menjelaskan proses Panitia melihat data yang ada | | |
| *Trigger:* Panitia melihat data yang ada, dan *form* *reset password.*  *Type: internal* | | |
| *Relationship*:  *Association:* Panitia  *Include*: *Login* | | |
| *Extend:*  *Generalization:* | | |
| *Normal flow of event:*   1. Ketika Panitia menekan tombol profil, makan otomatis akan ditampilkan halaman berisi data pribadi yang tersimpan pada *database*. 2. Panitia dapat melihat *form* *reset* *password.* 3. Ketika Panitia menekan tombol *edit* profil, maka ditampilkan *form* *edit.* | | |
| Subflows:  2a. Ketika halaman profil ditampilkan, mahasiswa juga dapat melihat *form* *reset password.*  2b. Untuk mengganti *password* harus mengisi data *password* lama, *password* baru, dan ulangi *password* baru pada *form* ganti *password,* kemudian tekan tombol *Reset.* Jika *form reset password* tidakterisi dengan benar, maka proses *reset password* dibatalkan*.*  3c. Panitia menekan tombol *edit* profil, akan diampilkan *form edit* sehingga mahasiswa dapat mengubah dan menyimpan kembali ke *database.* | | |

Tabel 4. 19 Deskripsi Use Case Daftar Mahasiswa

|  |  |  |
| --- | --- | --- |
| *Use Case Name:*  Daftar Mahasiswa | ID: UC.15  Siklus15 | *Importance Level:*  *High* |
| *Primary Actor:*  Panitia | *Use Case Type:* | |
| *Stakeholder and Interest:*  Untuk melihat data yang sudah tersimpan di *database*. | | |
| *Brief Description:*  Menjelaskan proses Panitia melihat data yang ada | | |
| *Trigger:* Panitia melihat data yang ada*.*  *Type: internal* | | |
| *Relationship*:  *Association:* Panitia  *Include*: *Login* | | |
| *Extend:*  *Generalization:* | | |
| *Normal flow of event:*   1. Ketika Panitia berhasil *login,* maka otomatis akan tampil data-data dokumen yang telah tersimpan pada *database*. 2. Ketika Panitia menekan menu D3 Manajemen Informatika atau D4 Teknik informatika, maka otomatis akan tampil data-data mahasiswa yang telah tersimpan pada *database.* | | |
| Subflows:  *.* | | |

Tabel 4. 20 Deskripsi Use Case Daftar Mahasiswa Telah Upload

|  |  |  |
| --- | --- | --- |
| *Use Case Name:*  Daftar Mahasiswa Telah *Upload* | ID: UC.16  Siklus16 | *Importance Level:*  *High* |
| *Primary Actor:*  Panitia | *Use Case Type:* | |
| *Stakeholder and Interest:*  Untuk melihat data yang sudah tersimpan di *database*. | | |
| *Brief Description:*  Menjelaskan proses Panitia melihat data yang ada | | |
| *Trigger:* Panitia melihat data yang ada*.*  *Type: internal* | | |
| *Relationship*:  *Association:* Panitia  *Include*: *Login* | | |
| *Extend:*  *Generalization:* | | |
| *Normal flow of event:*   1. Panitia memilih menu mahasiswa yang telah *upload,* maka otomatis akan tampil data-data mahasiswa yang telah *upload* dokumen tugas akhir atau skripsi telah tersimpan pada *database*. | | |
| Subflows:  *.* | | |

Tabel 4. 21 Deskripsi Use Case Lihat Hasil Deteksi

|  |  |  |
| --- | --- | --- |
| *Use Case Name:*  Lihat Hasil Deteksi | ID: UC.017  Siklus17 | *Importance Level:*  *High* |
| *Primary Actor:*  Panitia | *Use Case Type:* | |
| *Stakeholder and Interest:*  Untuk melihat data-data yang sudah tersimpan di *database*. | | |
| *Brief Description:*  Menjelaskan proses Panitia melihat data yang ada | | |
| *Trigger:* Panitia melihat data yang ada*.*  *Type: internal* | | |
| *Relationship*:  *Association:* Panitia  *Include*: *Login* | | |
| *Extend:*  *Generalization:* | | |
| *Normal flow of event:*   1. Panitia memilih menu Lihat hasil deteksi, maka otomatis akan tampil data-data hasil uji yang telah tersimpan pada *database*. 2. Panitia memilih dan menekan tombol lihat hasil, maka otomatis akan menampilkan data secara lengkap berdasarkan data yang dipilih. | | |
| Subflows:  2a. Ketika tombol lihat hasil ditekan, maka otomatis akan menampilkan data lengkap dari hasil uji plagiarisme yang menujukkan berapa *dataset* yang diujikan, berapa kata yang ditemukan.  *.* | | |

Tabel 4. 22 Deskripsi Use Case Upload Dataset

|  |  |  |
| --- | --- | --- |
| *Use Case Name:*  *Upload Dataset* | ID: UC.18  Siklus18 | *Importance Level:*  *High* |
| *Primary Actor:*  Panitia | *Use Case Type:* | |
| *Stakeholder and Interest:*  Panitia mengisi *form upload* *dataset* untuk menambah data pada *database*. | | |
| *Brief Description:*  Menjelaskan proses inisialisasi pengguna. | | |
| *Trigger:* Panitia menampilkan data-data yang ada*.*  *Type: internal* | | |
| *Relationship*:  *Association:* Panitia  *Include*: *Login* | | |
| *Extend:*  *Generalization:* | | |
| *Normal flow of event:*   1. Panita menekan menu *master* dan pilih *dataset,* maka otomatis akan tampil data-data *dataset* yang telah tersimpan pada *database* dan tombol *upload* *dataset* baru. 2. Panitia mengisi *form upload,* kemudian menekan tombol simpan, data otomatis dimasukkan *database* dan ditampilkan kembali pada halaman *dataset.* | | |
| Subflows:  *.* | | |

1. Database Schema

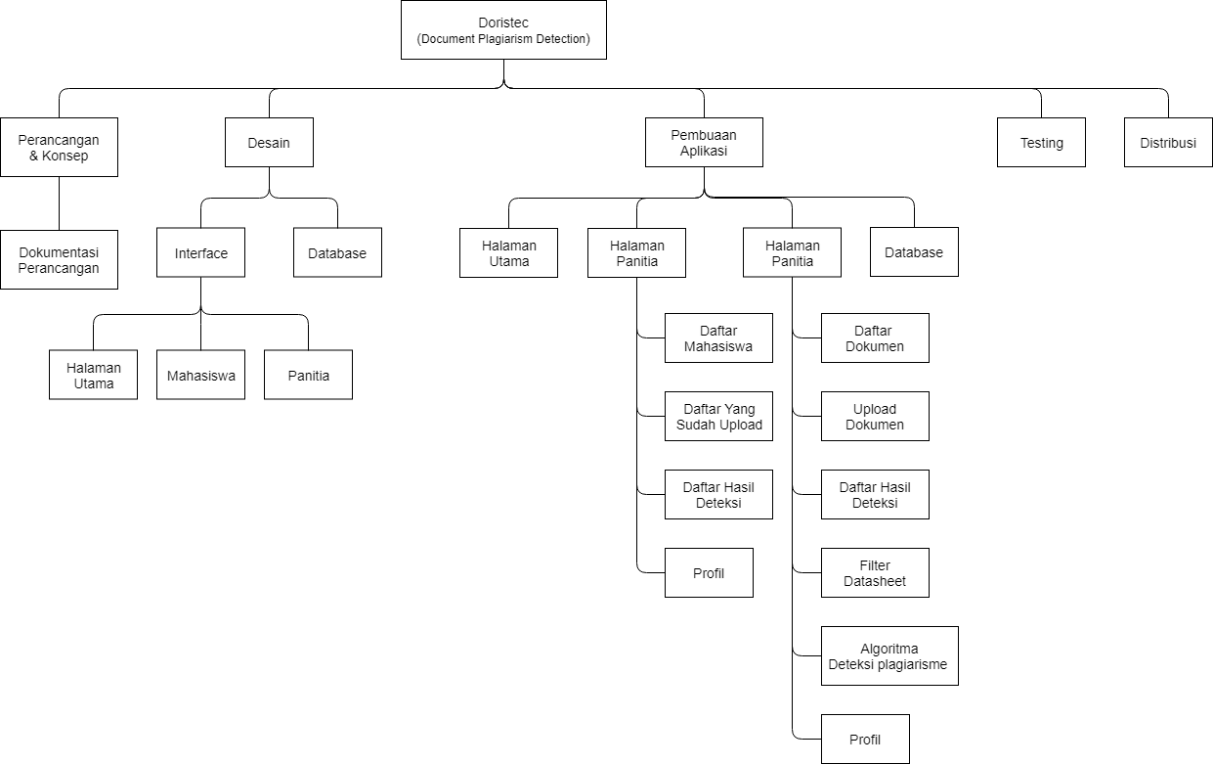
*Database schema* merupakan *blueprint* yang penulis rancang untuk menggambarkan deskripsi dari struktur basis data, tipe data, dan batasan yang dapat digunakan pada basis data. Berikut adalah skema basis data ditunjukkan pada gambar 4.6.



Gambar 4. 6 Database schema dari aplikasi

1. Work Breakdown Structure

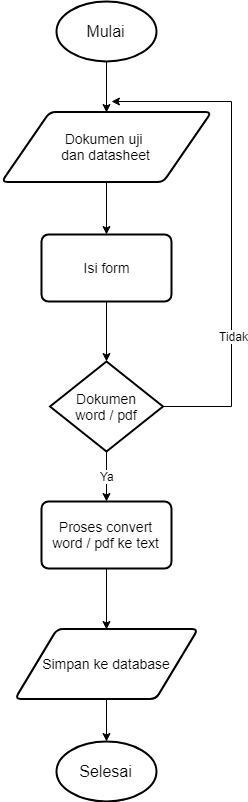
Pembuatan struktur kerja yang digambarkan oleh *Work breakdown structure* dapat mewakili bagaimana alur dari suatu pekerjaan yang harus diselesaikan dengan baik dan benar berdasarkan konsep yang telah dibuat. Pada sistem ini menerapkan *work breakdown structure* untuk mempermudah proses pengerjaan serta pengawasan pada sistem yang penulis bangun. Berikut ini penjabaran dari suatu pekerjaan yang harus dikerjakan saat pembuatan sistem ini ditunjjukan pada gambar 4.7.



Gambar 4. 7 work breakdown structure pada sistem

1. Upload Dokumen

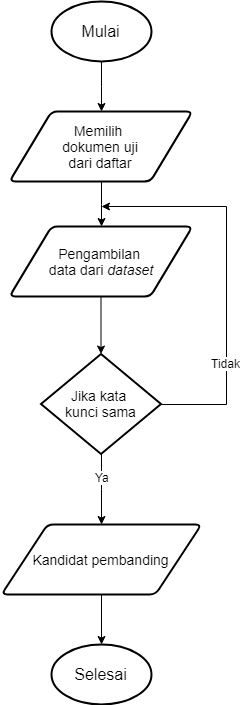
*Upload* dokumen merupakan proses untuk menyimpan kedalam *database*. Hal yang dilakukan sebelum disimpan terlebih dahulu dokumen word atau pdf dilakukan *convert* untuk mengambil keseluruhan text yang ada dalam dokumen, seperti yang ditunjukkan pada gambar 4.8.



Gambar 4. 8 Upload dokumen skripsi dan tugas akhir

1. Filter Kandidat Pembanding

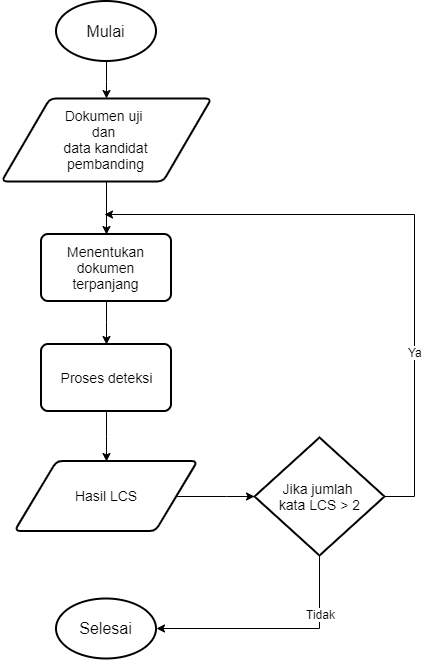
Algoritma *filter* yang diterapkan pada sistem ini akan diterapkan pada sebelum proses deteksi dilakukan, fitur deteksi yang ada pada akses pengguna mahasiswa, cara kerja pada metode ini adalah pengguna (mahasiswa) mengunggah dokumen skripsi atau tugas akhirnya atau jika pernah mengunggah sebelumnya, pengguna dapat memilih langsung dari daftar dokumen yang dimiliki. Selanjutnya melakukan *filter* berdasarkan kata kunci yang sesuai dengan dokumen yang akan diuji. Hasil dari *filter* ini berupa kumpulan data kandidat pembanding yang terpilih, *filter* ini akan bekerja seperti pada gambar 4.9.



Gambar 4. 9 Flowchart filter kandidat pembanding

1. Metode Longest Common Subsequence

Metode yang diterapkan pada sistem ini adalah *Longest Common Subsequence* (LCS)*,* metode LCS ini akan diterapkan pada proses ketika deteksi berlansung ketika tahap pada gambar 4.9 Telah dilalui. Metode ini diterapkan pada akses pengguna (mahasiswa). Berikut ini akan digambarkan seperti gambar 4.10.



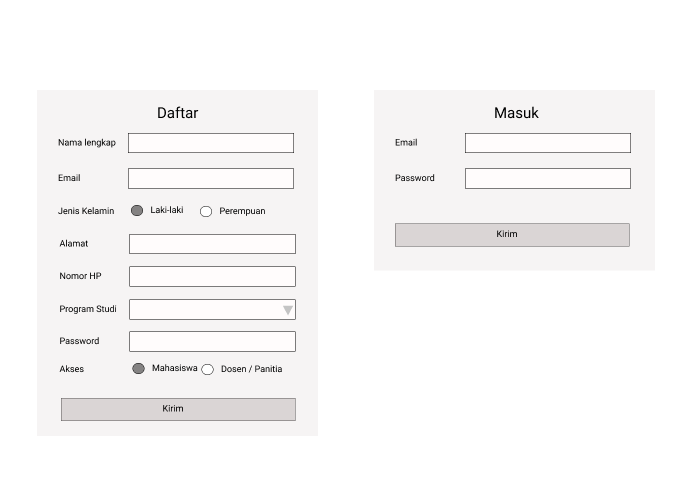
Gambar 4. 10 Flowchart proses LCS

1. Desain Antar Muka Pengguna

Perancangan desain *interface* dari penerapan Deteksi Plagiarisme Pada Dokumen Skripsi Berdasarkan Tingkat Kesamaan Dengan Menggunakan Metode *Longest Common Subsequence* terbagi berdasarkan level pengguna yang dapat diakses, adapun tampilan sebagai berikut :

1. Halaman Register dan Login

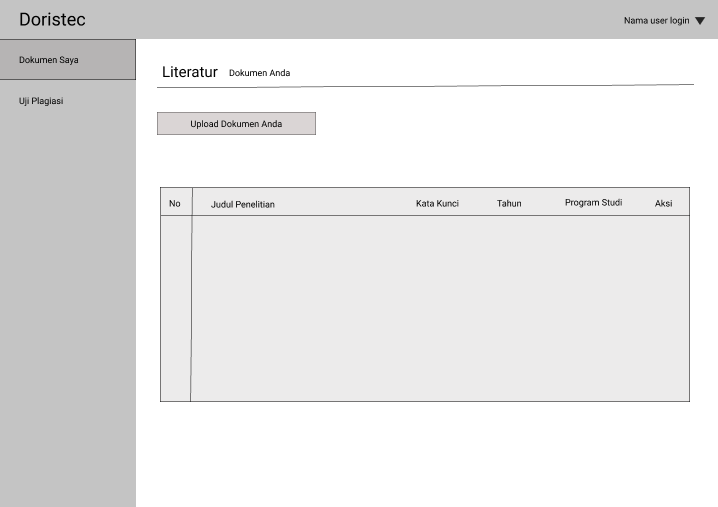
Halaman ini merupakan halaman pertama yang ditampilkan ketika pengguna masuk ke domain utama *website.* Isi dari halaman ini meliputi *form* registrasi dan *form* login seperti yang dicontohkan pada gambar 4.11.



Gambar 4. 11 Halaman Registrasi dan Login

1. Tampilan Halaman Mahasiswa
2. Halaman Dokumen

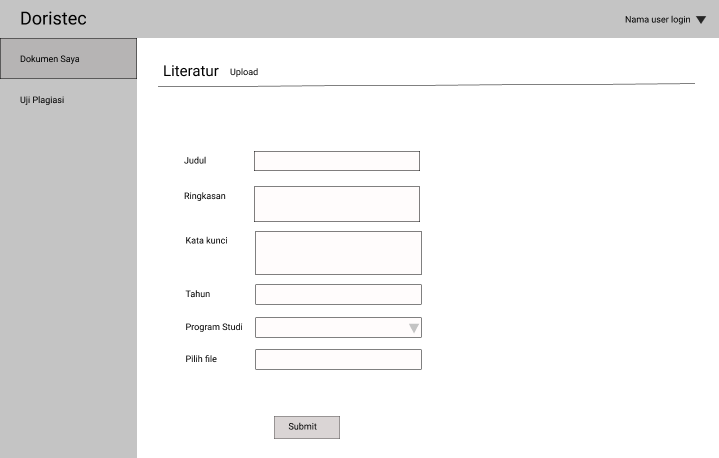
Halaman ini berisi tabel yang menampilkan daftar dokumen yang pernah di *upload* oleh mahasiswa, data yang ditampilkan berdasarkan id pengguna mahasiswa seperti yang ditunjukkan pada gambar 4.12, data-data yang ditampilkan pada tabel berisikan judul penelitian, kata kunci, tahun, program studi dan aksi yang dapat dilakukan oleh pengguna untuk setiap baris data. Selain tabel ada tombol untuk melakukan *upload* dokumen tugas akhir atau skripsi, tujuan diberikan tombol *upload* ini agar pengguna dapat dengan mudah menjangkau fitur *upload* dokumen baru.



Gambar 4. 12 Halaman Dokumen

1. Halaman *Upload* Dokumen

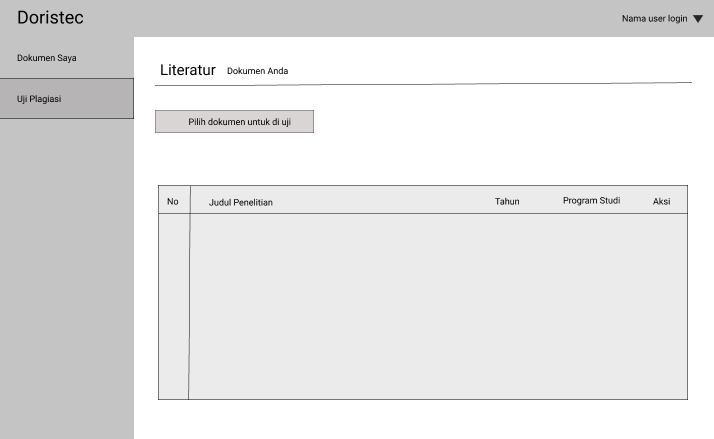
Halaman ini berisi *form* yang digunakan untuk *upload* dokumen tugas akhir atau skripsi, *form* ini harus di isi oleh mahasiswa sesuai isi dari dokumen tugas akhir atau skripsi yang akan di *upload,* seperti yang ditunjukkan pada gambar 4.13*.*



Gambar 4. 13 Halaman Upload Dokumen

1. Halaman Uji Plagiasi

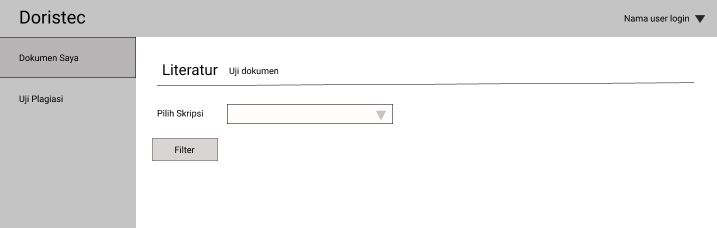
Halaman ini berisi tabel yang menampilkan daftar dokumen yang telah dilakukan proses uji plagiarisme seperti yang ditunjukkan pada gambar 4.14, selain tabel ada tombol untuk memulai uji plagiarisme.



Gambar 4. 14 Halaman Uji Plagiasi

1. Halaman Filter

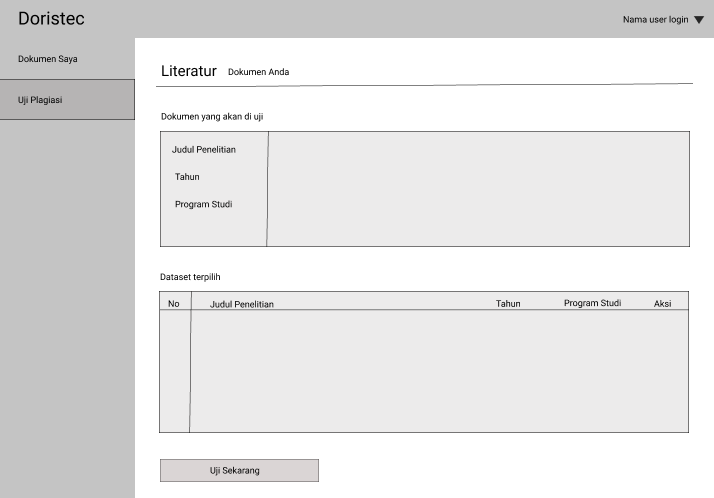
Halaman ini menampilkan dokumen apa saja yang telah diupload seperti yang ditunjukkan pada gamar 4.15, mahasiswa harus memilih dokumen mana yang akan di uji.



Gambar 4. 15 Halaman Filter

1. Halaman Proses Deteksi

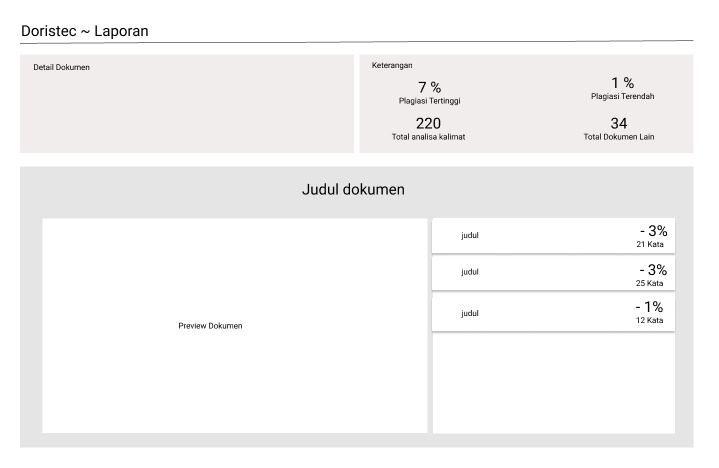
Halaman ini menampilkan hasil dari proses filter yang ditunjukkan pada gambar 4.16 berupa daftar *dataset* yang terpilih sebagai kandidat uji plagiasi.



Gambar 4. 16 Halaman Proses Deteksi

1. Halaman Laporan Hasil Uji Plagiasi

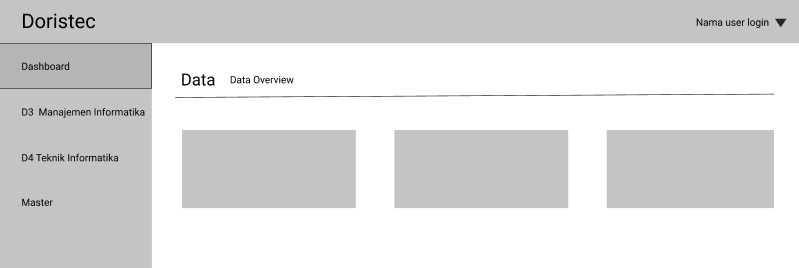
Halaman ini menampilkan hasil dari proses deteksi plagiarisme yang menghasilkan nilai-nilai besarnya plagiasi, yang telah ditunjukkan pada gambar 4.17.



Gambar 4. 17 Halaman Laporan Hasil Uji Plagiasi

1. Tampilan Halaman Panitia
2. Halaman Dashboard

Halaman ini menampilkan panel-panel yang berisi nilai seperti jumlah dokumen yang telah di *upload* berdasarkan program studi dan jumlah total mahasiswa yang terdaftar didalam aplikasi, seperti yang ditunjukkan pada gambar 4.18.



Gambar 4. 18 Halaman Dashboard

1. Halaman Data Mahasiswa D3 Manajemen Informatika

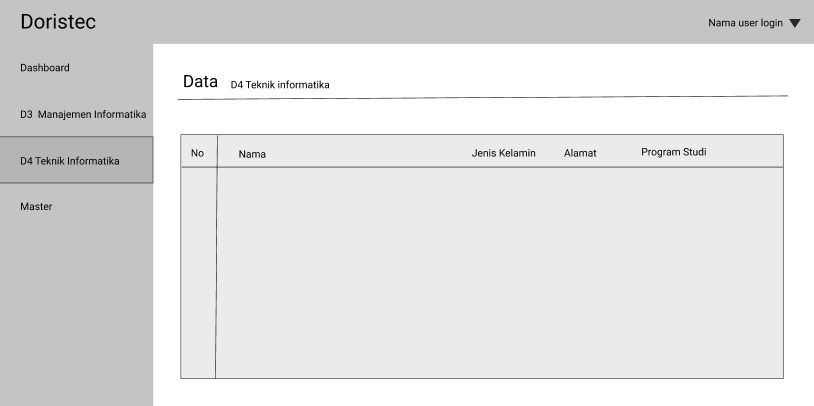
Halaman ini menampilkan semua data pengguna dari program studi D3 Manajemen Informatika yang terdaftar didalam aplikasi, seperti yang ditunjukkan pada gambar 4.19.



Gambar 4. 19 Halaman Data Mahasiswa D3 Manajemen Informatika

1. Halaman Data Mahasiswa D4 Teknik Informatika

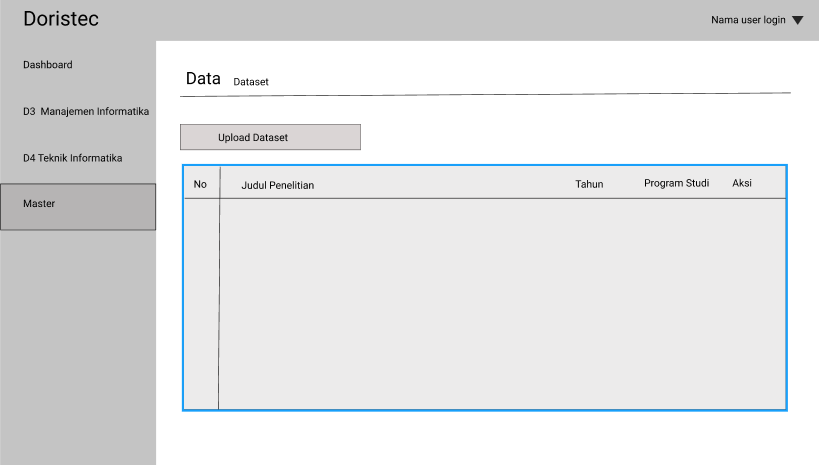
Halaman ini menampilkan semua data pengguna dari program studi D4 Teknik Informatika yang terdaftar didalam aplikasi, seperti yang ditunjukkan pada gambar 4.20.



Gambar 4. 20 Halaman Data Mahasiswa D4 Teknik Informatika

1. Halaman Dokumen *Dataset*

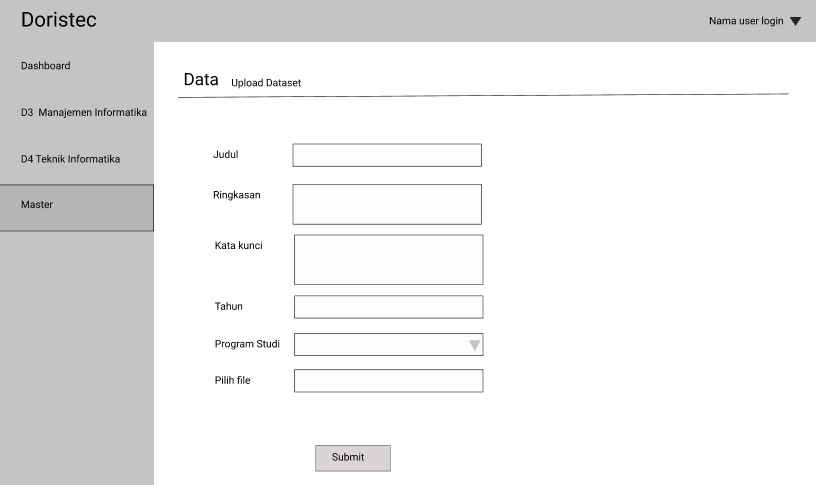
Halaman ini menampilkan daftar dokumen yang telah tersimpan didalam aplikasi, seperti yang ditunjukkan pada gambar 4.21.



Gambar 4. 21 Halaman Dokumen Dataset

1. Halaman Upload Dokumen Pembanding

Halaman ini menampilkan form yang harus diisi ketika menambah data dalam aplikasi, form tersebut berisi judul, ringkasan, kata kunci, tahun, program studi, dan pilih file dari PC, seperti yang ditunjukkan pada gambar 4.22.



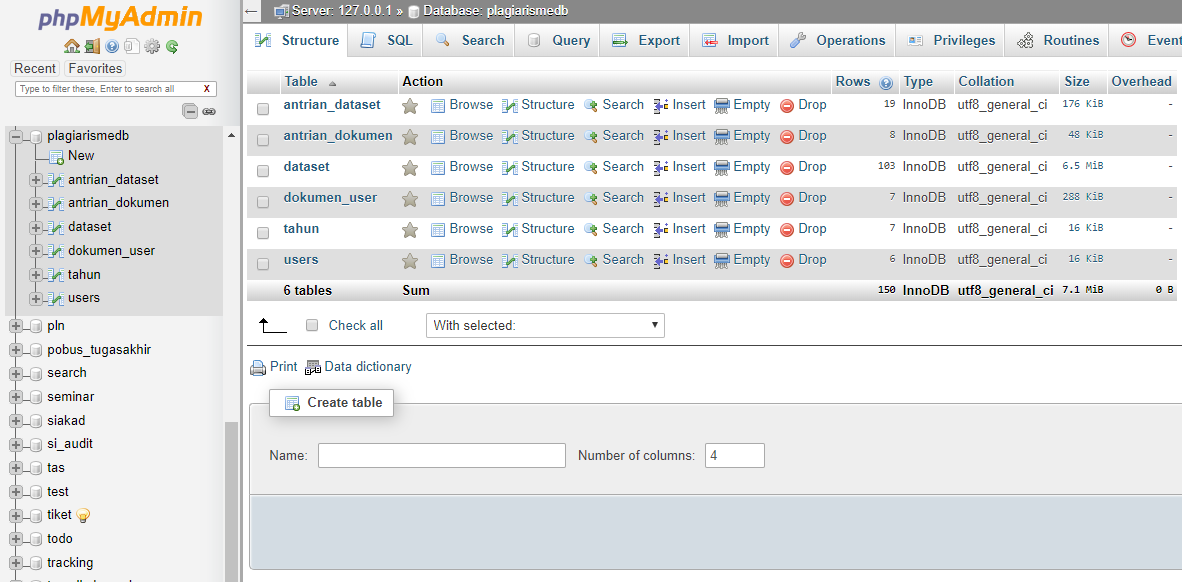
Gambar 4. 22 Halaman Upload Dokumen Pembanding

BAB V. IMPLEMENTASI DAN PENGUJIAN

Setelah dilakukan tahapan perancangan aplikasi, maka pada bab ini akan membahas mengenai proses implementasi basis data, implementasi sistem dan implementasi antar muka untuk merealisasikan perancangan menjadi nyata. Implementasi basis data dibangun menggunakan MySQL. Implementasi sistem terbagi menjadi dua, yaitu bahasa pemrograman Python digunakan sebagai fitur deteksi menggunakan LCS, dan bahasa pemrograman PHP dengan menggunakan *framework* Cogeigniter digunakan sebagai aplikasi berbasis *website* untuk pengguna mahasiswa dan panitia. Implementasi antar muka dibangun menggunakan bahasa pemrograman css dan jquery dengan menggunakan *framework* Bootstrap.

1. Implementasi Basis Data

Implementasi basis data merupakan hasil penerapan dari perancangan basis data yang telah dilakukan sebelumnya pada bab IV yang ditunjukkan pada gambar 4.8 yaitu skema basis data. Gambar 5.1 merupahan hasil dari penerapan asis data yang dibangung dengan menggunakan MySQL.



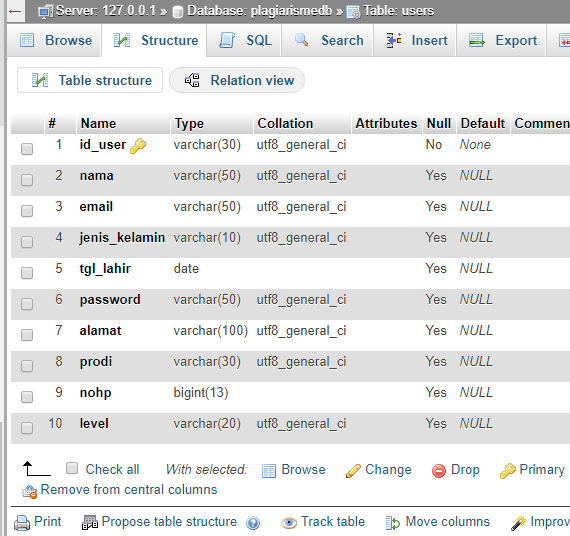
Gambar 5. 1 Basis Data Aplikasi

Basis data penelitian ini memiliki 6 tabel, yaitu tabel users, tabel tahun, tabel *dataset*, tabel dokumen\_user, tabel antrian\_dokumen, dan yang terakhir tabel

antrian\_*dataset*. Implementasi dari tabel-tabel pada gambar 5.1 adalah sebagai berikut :

1. Gambar Tabel Users

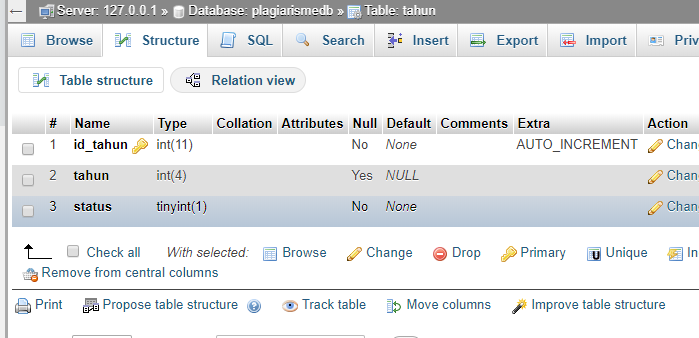
Pada gambar 5.2 menunjukkan implementasi dari rancangan tabel users yang berfungsi untuk menyimpan data mahasiswa dan panitia. Tabel ini memiliki sebanyak 10 *field* dengan 1 *primary key* antara lain : id\_user (*primary key*), nama, email, jenis\_kelamin, tgl\_lahir, *password*, alamat, prodi, nohp, level.



Gambar 5. 2 Tabel Users

1. Gambar Tabel Tahun

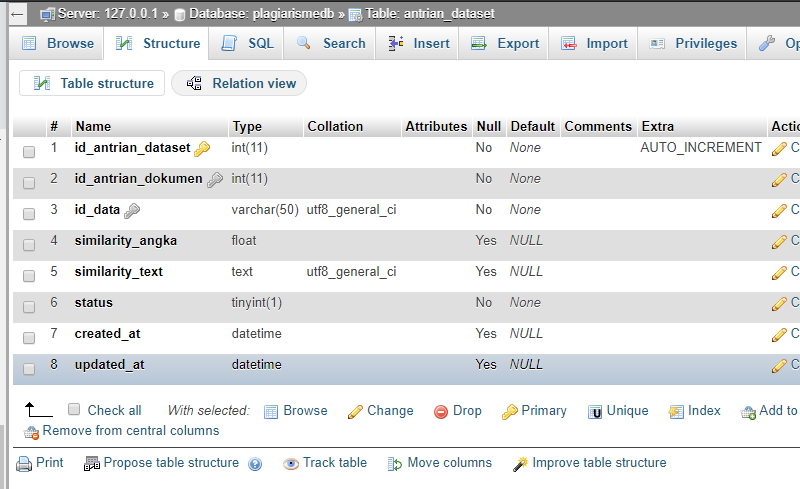
Pada gambar 5.3 menunjukkan implementasi dari rancangan tabel tahun yang berfungsi untuk menyimpan data tahun. Tabel ini memiliki sebanyak 3 *field* dengan memiliki 1 *primary key* antara lain : id\_tahun (*primary key*), tahun, status.



Gambar 5. 3 Tabel Tahun

1. Gambar Tabel Antrian\_*dataset*

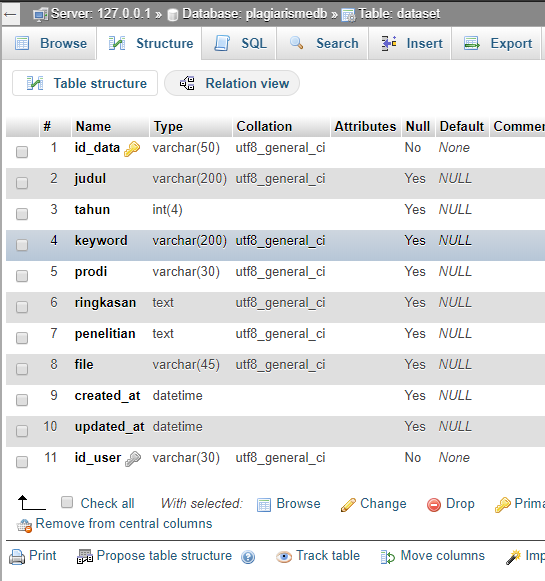
Pada gambar 5.4 menunjukkan implementasi dari rancangan tabel tahun yang berfungsi untuk menyimpan data antrian *dataset,* data antrian yang disimpan merupakan data-data yang diambil dari *dataset* yang dipilih sebagai kandidat pembanding. Tabel ini memiliki sebanyak 8 *field* dengan memiliki 1 *primary key* dan 2 *foreign key* antara lain : id\_antrian\_*dataset* (*primary key*), id\_antrian\_dokumen (*foreign key*), id\_data (*foreign key*), similarity\_angka, similarity\_text, status, created\_at, updated\_at.



Gambar 5. 4 Tabel Antrian Dataset

1. Gambar Tabel *Dataset*

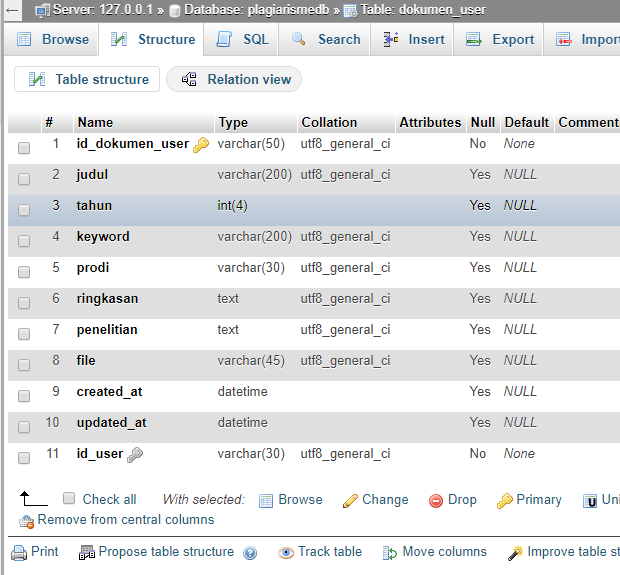
Pada gambar 5.5 menunjukkan implementasi dari rancangan tabel *dataset* yang berfungsi untuk menyimpan *dataset* berupa dokumen skripsi dan tugas akhir terdahulu. Tabel ini memiliki sebanyak 11 *field* dengan memiliki 1 *primary key* dan 1 *foreign key* antara lain : id\_data (*primary key*), judul, tahun, *keyword*, prodi, ringkasan, penelitian, file, created\_at, updated\_at, id\_user (*foreign key*).



Gambar 5. 5 Tabel Dataset

1. Gambar Tabel Dokumen\_user

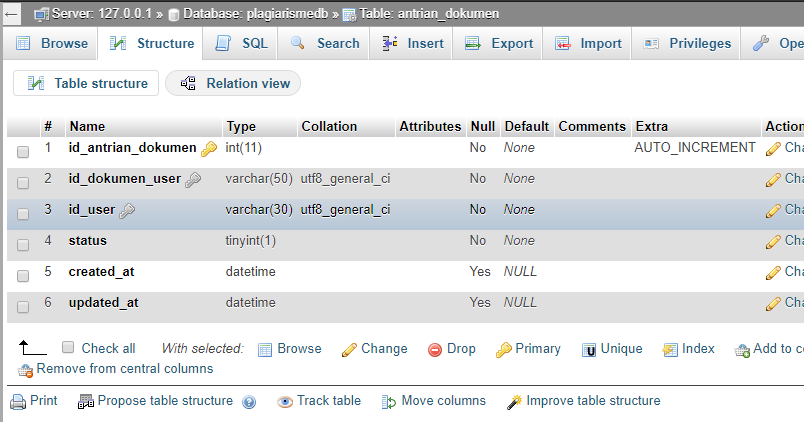
Pada gambar 5.6 menunjukkan implementasi dari rancangan tabel dokumen\_user yang berfungsi untuk menyimpan data dokumen user yang akan diuji yang berupa dokumen skripsi atau tugas akhir. Tabel ini memiliki sebanyak 11 *field* dengan memiliki 1 *primary key* dan 1 *foreign key* antara lain : id\_data (*primary key*), judul, tahun, *keyword*, prodi, ringkasan, penelitian, file, created\_at, updated\_at, id\_user (*foreign key*).



Gambar 5. 6 Tabel Dokumen User

1. Gambar Tabel Antrian\_dokumen

Pada gambar 5.7 menunjukkan implementasi dari rancangan tabel tahun yang berfungsi untuk menyimpan data antrian dokumen, data antrian yang disimpan merupakan data mahasiswa yang dipilih untuk dilakukan pengujian. Tabel ini memiliki sebanyak 6 *field* dengan memiliki 1 *primary key* dan 2 *foreign key* antara lain : id\_antrian\_dokumen (*primary key*), id\_dokumen\_user (*foreign key*), id\_user (*foreign key*), status, created\_at, updated\_at.



Gambar 5. 7 Tabel Antrian Dokumen

1. Implementasi Sistem

Implementasi sistem merupakan proses pengubahan sistem dari perancangan diubah menjadi aplikasi nyata yang diterapkan pada program. Berikut ini akan dijelaskan bagaimana implementasi dari perancangan sistem kedalam aplikasi deteksi plagiarisme pada dokumen skripsi berdasarkan tingkat kesamaan dengan menggunakan metode *Longest Common Subsequence.*

1. Implementasi *Convert* Dokumen ke Text

Berikut ini dijelaskan mengenai implementasi *library* DocxToTextConversion untuk mengkonversi file word ke teks dan pdf2text untuk mengkonversi *file* pdf ke teks pada sistem dalam bentuk potongan-potongan listing program.

Sebagian besar kode program untuk konversi dokumen ke teks dilakukan oleh *library* PHP yang terdapat pada folder *libraries* dengan nama *file* pdf2text.php untuk konversi dari pdf ke teks dan pada folder *controller*/*member* dengan nama *file* Dokumen.php untuk konversi dari file doc/docx ke teks ini juga berlaku pada folder *controller/admin* dengan nama *file* *Dataset*.php. Pertama-tama ketika *user* melakukan *submit* terhadap *form* maka dilakukan pengcekan terhadap *form input file* jika tidak diisi dengan *file* maka proses *input* dibatalkan, jika terisi dengan *file* maka akan dilanjutkan pengambilan nama *file*.

|  |
| --- |
| //uploading File  if(!$this->upload->do\_upload('filedoc'))  {  $this->session->set\_flashdata('error\_message', $this->upload->display\_errors());  redirect('dokumen/upload', 'refresh');  }  else  {  $data['file'] = $this->upload->data()['file\_name']; |

Nama *file* digunakan untuk pengambilan kembali *file* yang telah tersimpan pada folder projek (*assets/uploads*/doc), ketika *file* telah didapatkan dilakukan pembacaan *path info* untuk mengetahui dan mengambil *extension*. Setelah didapatkan *file* *extension* dilakukan pengecekan apakah *file* tersebut berupa pdf atau doc/docx, selanjutnya *file* akan diolah oleh setiap *library* yang sesuai dengan *file extension*, kemudian *library* akan mengembalikan hasil berupa *text* yang kemudian disimpan ke *database.*

|  |
| --- |
| // convert file to text  $file = './assets/uploads/doc/'.$data['file'];  $fileInformation = pathinfo($file);  $extension = $fileInformation['extension'];  //convert pdf to text  if ($extension == 'pdf') {  $converterpdf = new PDF2Text();  $converterpdf->setFilename($file);  $converterpdf->decodePDF();  $data['penelitian'] = $converterpdf->output();  //convert word to text  }elseif ($extension == 'doc' || $extension == 'docx') {  $converter = new DocxToTextConversion($file);  $data['penelitian'] = $converter->convertToText();  } else {  return 'Invalid File Type, please use pdf, doc or docx word document file.';  }  $this->session->set\_flashdata('success\_message', 'Tambah dokumen berhasil');  $this->Dokumen\_user\_model->insert($data);  echo "<script type='text/javascript'>window.location.href='http://localhost/frontendskripsi/dokumen';</script>";  // redirect('dokumen','refresh');  } |

1. Implementasi *Filter* Pada Kandidat Pembanding

Berikut ini dijelaskan mengenai implementasi *filter* yang dilakukan ketika pengambilan *dataset* dari *database* berdasarkan kata kunci pada dokumen uji. Kode program ini hanya diimplementasikan pada level mahasiswa, yang terletak pada folder *controller/member* dengan nama *file* Tesdokumen.php.

Kode program *filter* kandidat pembanding terdapat pada fungsi prosesFilter() yang terdapat pada *file* Tesdokumen.php. Pertama-tama fungsi ini akan bekerja ketika pada *form input* dilakukan aksi *submit*.

|  |
| --- |
| public function prosesFilter()  {  if ($this->input->post('submit')) {  $data\_uji = array();  $id = $this->input->post('id\_dokumen\_user');  $result = $this->Dokumen\_user\_model->select\_by\_id($id);  $data['dokumenuji'] = $result;  $result\_keyword = explode(",",$result[0]['keyword']); //ambil keywoard |

Pada inisialisasi *array* dengan nama variabel $data\_uji digunakan untuk menyimpan hasil dari proses *filter* sebelum disimpan kedalam *database.* Dilakukan pengambilan id dokumen *user* dari *form input*, id ini digunakan untuk mengambil data lengkap dari *database* berdasarkan id dokumen *user* yang telah didapatkan, ketika mendapatkan data lengkap dimasukkan pada variabel array dengan nama dokumenuji. Selanjutnya pengambilan kata kunci untuk dilakukan pemisahan berdasarkan karakter koma ( , ) dengan menggunakan fungsi *explode* sehingga akan menghasilkan data berupa array, seperti contoh pada tabel 5.1

Tabel 5. 1 Contoh Pemisahan Kata Kunci Berdasarkan Koma

|  |  |
| --- | --- |
| Contoh pemisahan kata kunci berdasarkan koma ( , ) | |
| Bentuk awal | Sistem,Penelitian,Deteksi,Dokumen |
| Bentuk Akhir | array(4) {  [0] => string(6) "Sistem"  [1] => string(10) "Penelitian"  [2] => string(7) "Deteksi"  [3] => string(7) "Dokumen"  } |

Pada proses selanjutnya yaitu perulangan berdasarkan jumlah *array* kata kunci, dari setiap *array* akan diambilkan data berdasarkan isi dari setiap *array*. Karna hasil dari pengambilan *dataset* sangat mungkin untuk memiliki lebih dari satu data, maka dilakukan perulangan untuk mengambil disetiap datanya untuk dimasukkan kedalam *array* baru.

|  |
| --- |
| for ($i=0; $i < count($result\_keyword) ; $i++) {    $dataset = $this->Dataset\_model->dataset\_by\_tags($result\_keyword[$i]); //pilih dataset sesuai keyword  // cek jika $dataset lebih dari 1  foreach ($dataset as $key) {  $data\_uji[] = $key;  }  }  $data['datauji'] = array\_unique($data\_uji, SORT\_REGULAR); //convert ke array of string  $data['content'] = $this->load->view('member/tesdokumen/datauji', $data, TRUE);  $data['desc'] = "Data Terpilih";  $this->load->view('member/template', $data); |

Pada *array* baru dengan nama $data\_uji memungkinkan untuk memiliki isi yang sama, karna ketika pengambilan berdasarkan tiap kata kunci bisa terjadi pengambilan *dataset* yang sama dengan kata kunci yang sama pula. Sehingga diperlukan pembuangan data *array* yang memiliki *value* sama.

1. Implementasi Metode LCS Pada Python

Berikut ini dijelaskan mengenai implementasi metode *Longest Common Subsequence* yang dilakukan ketika menguji dokumen untuk dibandingkan antara dokumen uji dengan kandidat pembanding dari *database,* untuk data yang diambil berdasarkan kata kunci pada dokumen uji. Kode program ini terletak pada folder *assets/python* dengan nama *file* *setup*.*py* dan lcs.py.

Dalam penerapannya, kode program dari metode LCS akan bekerja ketika mendapatkan *request* dari *website* untuk melakukan pengujian. Karna pada metode LCS dibangun menggunakan bahasa pemrograman *python* sehingga diperlukan pemanggilan dan kirim data secara khusus, seperti yang ditunjjukan pada kode program berikut :

|  |
| --- |
| public function executeCheker()  {  if ($this->input->post('submit')) {  $data1['id\_dokumen\_user'] = $this->input->post('id\_dokumen\_user');  $data1['id\_user'] = $this->session->userdata('logged\_in')['id\_user'];  $data1['created\_at'] = date('Y-m-d H:i:s');  $data1['status'] = 1;  $id\_data = $this->input->post('id\_data');  $id = $this->Antrian\_dokumen\_model->insert($data1); |

Ketika tombol uji ditekan, maka akan dijalankan fungsi yang bernama executeCheker(), fungsi ini akan menerima data dari *form input* yang berisikan id dokumen dan id user yang akan disimpan ke *database* dengan nama tabel antrian\_dokumen melalui model Antrian\_dokumen\_model, untuk menerima *input* berisi kandidat pembanding dari *form input* yang berisikan kumpulan id kandidat pembanding. Kumpulan dari id kandidat pembanding ini akan disimpan pada tabel antrian\_*dataset*. Untuk dapat menyimpan kumpulan id *dataset* pada tabel antrian\_*dataset* diperlukan penjabaran nama pada setiap id agar sesuai. Berikut ini kode program untuk pemberian nama sesuai dengan nama kolom pada tabel antrian\_*dataset*.

|  |
| --- |
| $data = array();  $path = './assets/python/setup.py';  foreach ($id\_data as $value) {  array\_push($data, array(  'id\_data' => $value,  'id\_antrian\_dokumen' => $id,  'created\_at' => date('Y-m-d H:i:s')  ));  }  $this->Antrian\_dataset\_model->insert\_batch($data); |

Ketika pemberian nama selesai, data akan disimpan ke *database* melalui model Antrian\_*dataset*\_model. Sehingga dapat melanjutkan pada proses berikutnya yaitu pemanggilan kode program LCS.

|  |
| --- |
| $output = passthru("python ".$path." ".$data1['id\_dokumen\_user']." ".$id); //array [0] = $path, array [1] = id\_dokumen\_user, array [2] = $value (data yg di uji)  redirect('tes-plagiasi');  }else {  redirect('tes-plagiasi/uji');  }  } |

Pemanggilan dilakukan secara khusus karna bahasa pemrograman yang digunakan menggunakan bahasa pemrograman *python*, untuk memanggil file *python* dilakukan menggunakan fungsi passthru(). Pada fungsi ini diwajibkan untuk menyebutkan letak dari *file* *python,* dan untuk dua parameter berikutnya diisi jika memang diperlukan untuk mengirim data pada *file python.* Disini penulis mengisi dua parameter yang berisi id dokumen uji dan id antrian dokumen.

*File* yang pertama kali menerima *request* dari *website* adalah *setup.py*, dimana pada *file* ini berisi kode program untuk mengambil data dari *database* berdasarkan id dokumen uji dan dokumen kandidat pembanding berdasarkan id antrian dokumen yang dikirim dari *website* ketika *request* dilakukan.

|  |
| --- |
| import sys  import MySQLdb  import datetime  from lcs import LCS  class Database:  host = "localhost"  user = "root"  password = ""  db = "plagiarismedb"  def \_\_init\_\_(self):  self.connection = MySQLdb.connect( host = self.host,  user = self.user,  password = self.password,  db = self.db ,  autocommit=True)  def query(self, q):  cursor = self.connection.cursor()  cursor.execute(q)  self.connection.commit()  cursor.close()  #self.connection.close()  def queryall(self, q, va):  cursor = self.connection.cursor( MySQLdb.cursors.DictCursor )  cursor.execute(q, va)  return cursor.fetchall()  def queryv(self, q, va):  cursor = self.connection.cursor( MySQLdb.cursors.DictCursor )  cursor.execute(q, va)  return cursor.fetchone()  def \_\_del\_\_(self):  self.connection.close() |

Kode program diatas merupakan fungsi-fungsi yang digunakan untuk mengakses *database.* Untuk mengoprasikan pengambilan data serta menjalankan fungsi metode LCS diterapkan pada fungsi dibawah ini.

|  |
| --- |
| if \_\_name\_\_== "\_\_main\_\_":  db = Database()  id\_doc\_uji = sys.argv[1] #dokumen yg di uji  id\_antrian = sys.argv[2] #ambil id antrian  doc\_uji = (id\_doc\_uji,) #harus ada koma agar tidak eror  antrian = (id\_antrian,)  q = "SELECT \* FROM dokumen\_user WHERE id\_dokumen\_user = %s"  data = db.queryv(q, doc\_uji)  q = "SELECT \* FROM antrian\_dataset as ad join dataset as ds on ad.id\_data = ds.id\_data WHERE id\_antrian\_dokumen = %s"  data1 = db.queryall(q, antrian) |

Pertama-tama yang dilakukan adalah inisialisasi *database*, kemudian membuat variabel yang digunakan untuk menampung data yang dikirim dari website. Kemudian melakukan *query* ke *database* pada tabel dokumen\_user dan antrian\_*dataset*. Data yang didapatkan dari *query* diproses dalam perulangan berikut ini.

|  |
| --- |
| for td in data1:  list\_sentence = ""  angka\_sentence = 0  penelitian = data['penelitian']  while True:  common\_sentence, persent = LCS.longest\_common\_sentence(penelitian, td['penelitian'])  common\_words = common\_sentence.split(' ')  #berhenti ketika hasil plagiat kurang dari 3 kata  if(len(common\_words) < 3):  break  else:  list\_sentence += '%s###' % common\_sentence #mengumpulkan hasil lcs ke 1 variable  angka\_sentence += float(persent)  buang\_plagiarisme = penelitian.split(common\_sentence) #membuang kalimat plagiat pada text uji  penelitian = ' '.join(buang\_plagiarisme) #menggabungkan kembali setelah meembuang kalimat plagiat  q = "UPDATE antrian\_dataset SET similarity\_angka='%s', similarity\_text='%s', status=%d, updated\_at='%s' WHERE id\_antrian\_dataset='%s' "% (angka\_sentence, list\_sentence, 1, datetime.datetime.now(), td['id\_antrian\_dataset'])  db.query(q) |

Perulangan pertama berisi dari banyaknya kandidat pembanding yang diuji, pada perulangan kedua dilakukan untuk menguji pada setiap kandidat pembanding hingga tidak ditemukan lagi kalimat yang sama dengan batas minimal tiga kata dalam satu kalimat. Selanjutnya memanggil fungsi longest\_common\_sentence() dari kelas LCS, fungsi tersebut membutuhkan dua parameter yang berisi teks dari dokumen uji dan teks dari kandidat pembanding*,* proses ini akan menghasilkan dua output data sekaligus yaitu kalimat yang mengandung plagiarisme dan nilai plagiarisme (dalam persen) disetiap pengujian padamasing-masing kandidat pembanding.

Kondisi berikutnya untuk menentukan jika kalimat yang mengandung plagiarisme berjumlah kurang dari tiga kata, maka perulangan akan dihentikan, namun apabila lebih dari atau sama dengan tiga kata, maka perulangan masih terus dijalankan. Setelah didapatkan hasil secara keseluruhan selanjutnya akan dikirim kembali ke database untuk disimpan kembali.

Setelah pembahasan kode program pada *file* *setup.*py, pembahasan selanjutnya adalah kode program LCS yang terdapat pada file lcs.py dengan nama fungsi longest\_common\_substring(). Disinilah metode lcs diterapkan, fungsi ini akan bekerja dan menghasilkan deretan *array* kata dalam satu kalimat panjang.

|  |
| --- |
| def longest\_common\_substring(s1, s2):  m = [[0] \* (1 + len(s2)) for i in range(1 + len(s1))]  longest, x\_longest = 0, 0    for x in range(1, 1 + len(s1)): #ambil per kata dari doc 1, dan x sebagai doc terpanjang  for y in range(1, 1 + len(s2)): #ambil per kata dari doc 2  if s1[x - 1] == s2[y - 1]: #cek juka kata dari doc 1 == doc 2  m[x][y] = m[x - 1][y - 1] + 1  if m[x][y] > longest:  longest = m[x][y]  x\_longest = x  else:  m[x][y] = 0  return s1[x\_longest - longest: x\_longest] #memilih lcs terpanjang |

Untuk memaksimalkan fungsi longest\_common\_substring() dalam mencari kalimat, sebelumnya diperlukan penentuan untuk ngengetahui antara dokumen uji dan kandidat pembanding manakah yang memiliki kata paling Panjang. Pada fungsi dibawah ini akan dilakukan pengecekan panjang kata, kata yang paling panjang akan disimpan pada variabel s1\_arr dan kata yang lebih pendek akan disimpan pada variabel s2\_arr. Fungsi ini menghasilkan deretan kalimat panjang serta nilai plagiarisme disetiap satu kali fungsi ini dijalankan.

|  |
| --- |
| longest\_common\_sentence(s1, s2):  s1\_arr = []  s2\_arr = []  s1\_words = s1.split(' ') #memisah tiap kata menjadi array  s2\_words = s2.split(' ')    lg\_s1 = len(s1\_words) #menghitung panjang  lg\_s2 = len(s2\_words)    if lg\_s1 < lg\_s2: #menentukan dokumen terpanjang  s1\_arr = s2\_words  s2\_arr = s1\_words  else:  s1\_arr = s1\_words  s2\_arr = s2\_words    lcs\_arr = LCS.longest\_common\_substring(s1\_arr, s2\_arr)  s1\_pesent = (len(lcs\_arr)\*100)/len(s1\_arr)    lcs = ' '.join(lcs\_arr) #menggabungkan array hasil lcs  return lcs, s1\_pesent |

1. Implementasi *Output* dan *Ranking* Pada Hasil Uji Plagiarisme

Berikut ini dijelaskan mengenai implementasi *Output* hasil pengujian yang diambil dari *database* berdasarkan id dokumen yang diuji. Kode program ini terletak pada folder *controller/member* dengan nama *file* Tesdokumen.php dan pada folder *view/member/tesdokumen* dengan nama *file* detailhasil.php.

Pada *controller* Tesdokumen.php pada fungsi detail() dijelaskan pada kode program berikut ini.

|  |
| --- |
| public function detail($id)  {  $data['detail'] = $this->Antrian\_dokumen\_model->select\_detail($id);  $data['dataset'] = $this->Antrian\_dataset\_model->select\_detail($id);  $data['content'] = $this->load->view('member/tesdokumen/detailhasil', $data, TRUE);  $data['desc'] = "Laporan Hasil Deteksi";  $this->load->view('member/template', $data);  } |

Data yang akan ditampilkan diambil dari model Antrian\_dokumen\_model dan Antrian\_*dataset*\_model kemudian dikirimkan ke *view*. Pada *view* detailhasil.php terdapat kode program seperti berikut ini.

|  |
| --- |
| <?php  // menghitung hasil dari dataset  $totalkata = 0; //untuk menghitung total kata  $totalangka = 0; //untuk menghitung total angka  $angka = array();  foreach ($dataset as $value) {  array\_push($angka, $value['similarity\_angka']);  $pecah\_kata = explode("###",$value['similarity\_text']); //memisahkan kalimat berdasarkan karakter ###  $hit\_total\_kata = 0;  foreach ($pecah\_kata as $value\_kata) {  $hitungkata = str\_word\_count($value\_kata, 1); //1 untuk membuat array dan mengatur index array, [Ketika returnformat diatur ke 1, itu akan mengembalikan array], [Ketika returnformat diatur ke 2, itu akan mengembalikan array asosiatif]  $hit\_total\_kata += count($hitungkata);  }  $totalkata += $hit\_total\_kata;  $totalangka += $value['similarity\_angka'];  }  $ordered\_angka = $angka;  rsort($ordered\_angka); //pembuatan ranking pada dataset nilai 'similarity\_angka'  $dataset\_ranking = array();  foreach ($ordered\_angka as $key\_angka => $value\_angka) {  foreach ($dataset as $key => $value) {  if ($value['similarity\_angka'] == $value\_angka) { //penyesuaian dataset yang akan di tampilkan sesuai ranking tertinggi  array\_push($dataset\_ranking, $value);  }  }  }  ?> |

Pada kode program diatas terdapat bagian kode untuk menerapkan *ranking* pada kandidatpembanding*.* *Ranking* digunakan sebagai penentu bahwa dengan kandidat mana yang memiliki nilai plagiarisme tertinggi dan yang terendah. Selain itu ada juga kode program yang digunakan untuk menampilkan *detail* dari setiap kandidatmengunakan bahasa pemrograman javascript*.*

|  |
| --- |
| <script type="text/javascript">  function onDatasetView(dataset) {  document.getElementById("judul").innerHTML = dataset.judul;  document.getElementById("angka").innerHTML = Math.round(dataset.similarity\_angka)+'%';  document.getElementById("tahun").innerHTML = dataset.tahun;  document.getElementById("prodi").innerHTML = dataset.prodi;  document.getElementById("waktu").innerHTML = dataset.created\_at;  var pisah\_hasil\_text = dataset.similarity\_text.split("###");  document.getElementById("text").innerHTML = "";  pisah\_hasil\_text.forEach(function (value){  if (value === "") {  }else {  document.getElementById("text").innerHTML +=`  <p class="hasil\_text">${value}</p>  `  }  })  var keyword = dataset.keyword.split(",");  document.getElementById("keyword").innerHTML = "";  keyword.forEach(function (value) {  document.getElementById("keyword").innerHTML += `  <span class="label label-info">${value}</span>  `  })  $('#ModalDataset').modal('show');  }  </script> |

1. Implementasi Antar Muka

Implementasi antar muka merupakan proses pengubahan dari perancangan desain antar muka diubah menjadi tampilan aplikasi nyata yang diterapkan pada program. Berikut ini akan dijelaskan bagaimana implementasi dari perancangan desain antar muka kedalam aplikasi deteksi plagiarisme pada dokumen skripsi berdasarkan tingkat kesamaan dengan menggunakan *Longest Common Subsequence.*

1. Halaman Registrasi dan *Login*

Halaman Registrasi dan *Login* merupakan halaman utama pada *website* ketika pengguna mengunjungi *website* doristec ditunjukkan pada gambar 5.8, gambar 5.9, gambar 5.10, gambar 5.11, halaman ini memuat fitur-fitur yang berisikan informasi singkat seputar *website* doristec serta *form* Regidtrasi dan *form Login.* Halaman utama ini bersifat umum, sehingga bagi mahasiswa maupun panitia yang akan melakukan registrasi dan *login* melalui halaman utama ini. Untuk desain yang digunakan pada *website* ini memilih desain yang simple agar mudah digunakan. Pemilihan desain menggunakan *framework bootstrap custom*.

Pengguna yang belum memiliki akun pada *website* doristec, dihimbau untuk melakukan registrasi terlebih dahulu. Pengguna harus memasukkan nama lengkap, *email,* jenis kelamin, alamat, nomor handphone, program studi, password, hak akses. Data-data tersebut hasus sesuai dengan identitas pengguna serta hak akses yang dibutuhkan.

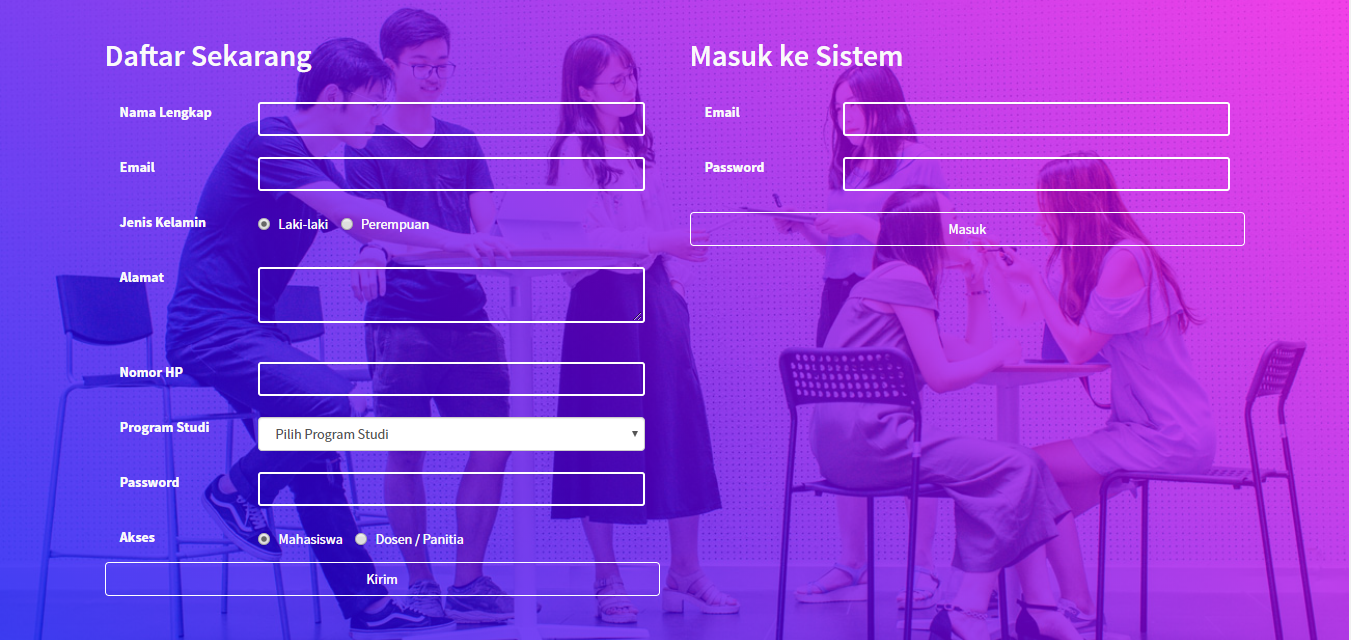
Ketika telah mendapatkan akun, pengguna bisa melanjutkan untuk *login* kedalam sistem doristec. Untuk melakukan *login* Pengguna mahasiswa maupun panitia harus memasukkan *username* dan *password* untuk bisa mendapatkan hak akses.



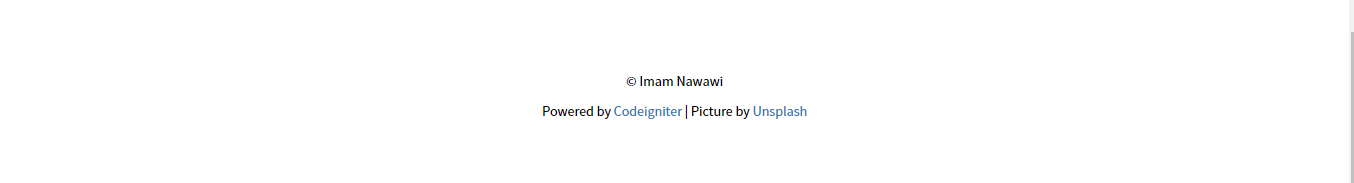
Gambar 5. 8 Halaman Utama Website



Gambar 5. 9 Lanjutan Halaman Utama Website



Gambar 5. 10 Lanjutan Halaman Utama Website



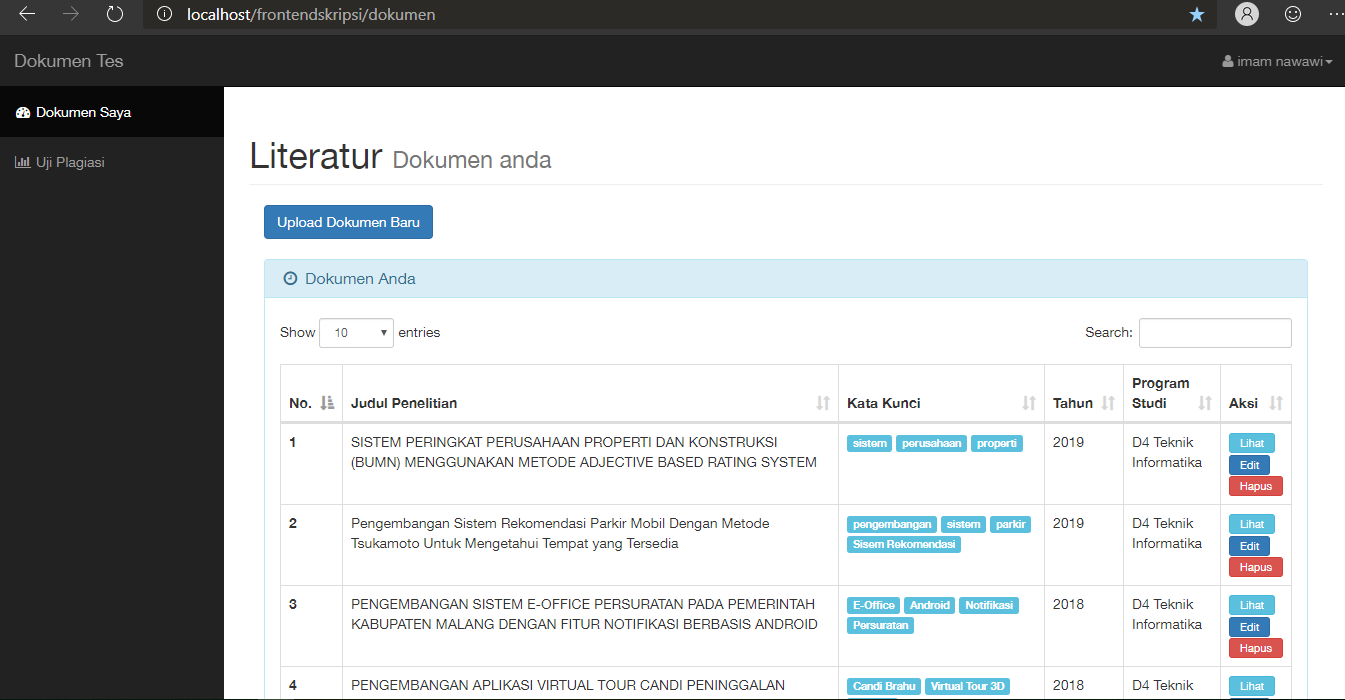
Gambar 5. 11 Lanjutan Halaman Utama Website

1. Halaman Mahasiswa

Halaman mahasiswa merupakan halaman admin yang ditujukan bagi pengguna mahasiswa, halaman ini memuat fitur-fitur yang berisikan daftar dokumen anda, upload dokumen baru, daftar dokumen yang pernah diuji, membuat pengujian baru, melihat hasil yang pernah diuji. Untuk desain yang digunakan pada *website* ini memilih desain yang simple agar mudah digunakan. Pemilihan desain menggunakan *theme startbbootstrap-admin*.

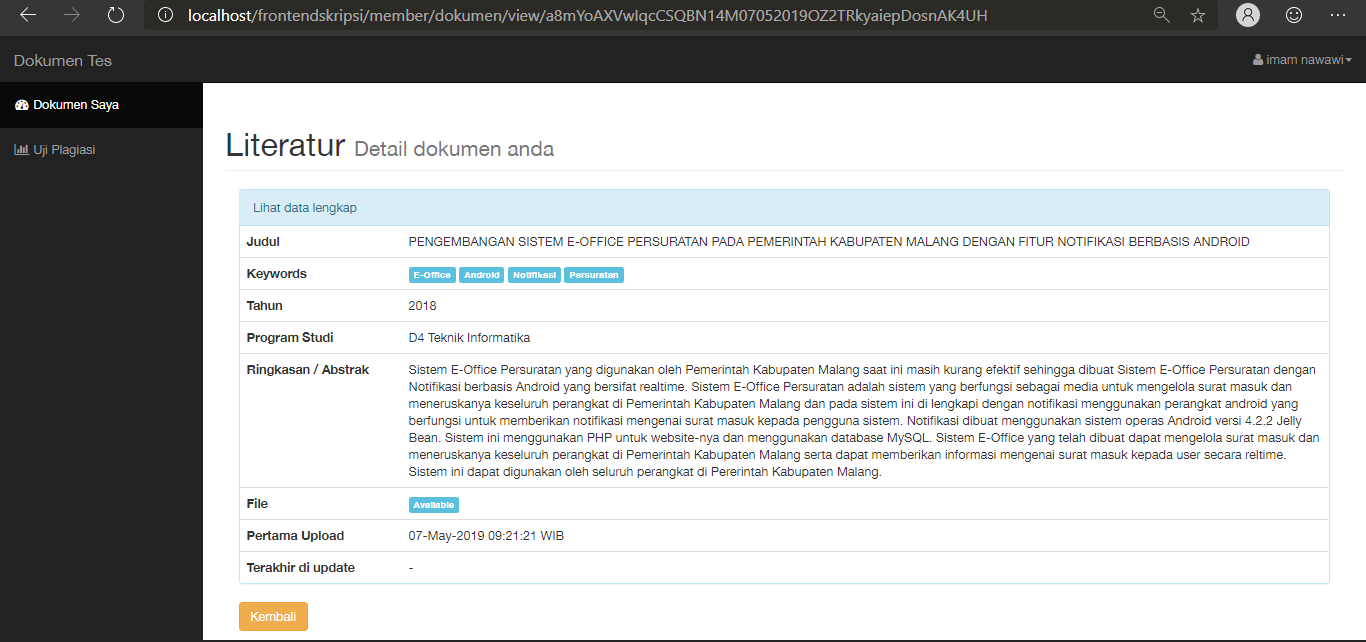
1. Halaman Utama Admin Mahasiswa

Halaman utama admin mahasiswa merupakan halaman awal yang ditampilkan dari ketika *login* berhasil dilakukan. Pada halaman ini terdapat fitur dokumen anda, *upload* dokumen baru. Pada saat halaman utama di buka mahasiswa akan diarahkan langsung menuju dokumen anda dimana fitur ini berfungsi untuk menampilkan semua data dari dokumen yang pernah di *upload* dan sudah tersimpan di *database*. Data tersebut antara lain judul penelitian, kata kunci, tahun dan program studi pada halaman ini mahasiswa mempunyai hak akses untuk mengolah data dokumen anda, pada halaman ini terdapat tiga tombol yaitu tombol lihat, edit dan hapus. Tombol lihat berfungsi untuk melihat secara keseluruhan data yang terpilih, tombol edit berfungsi untuk mengedit data yang terpilih, sedangkan tombol hapus untuk menghapus data yang terpilih, untuk tampilan ditunjukkan pada gambar 5.12.



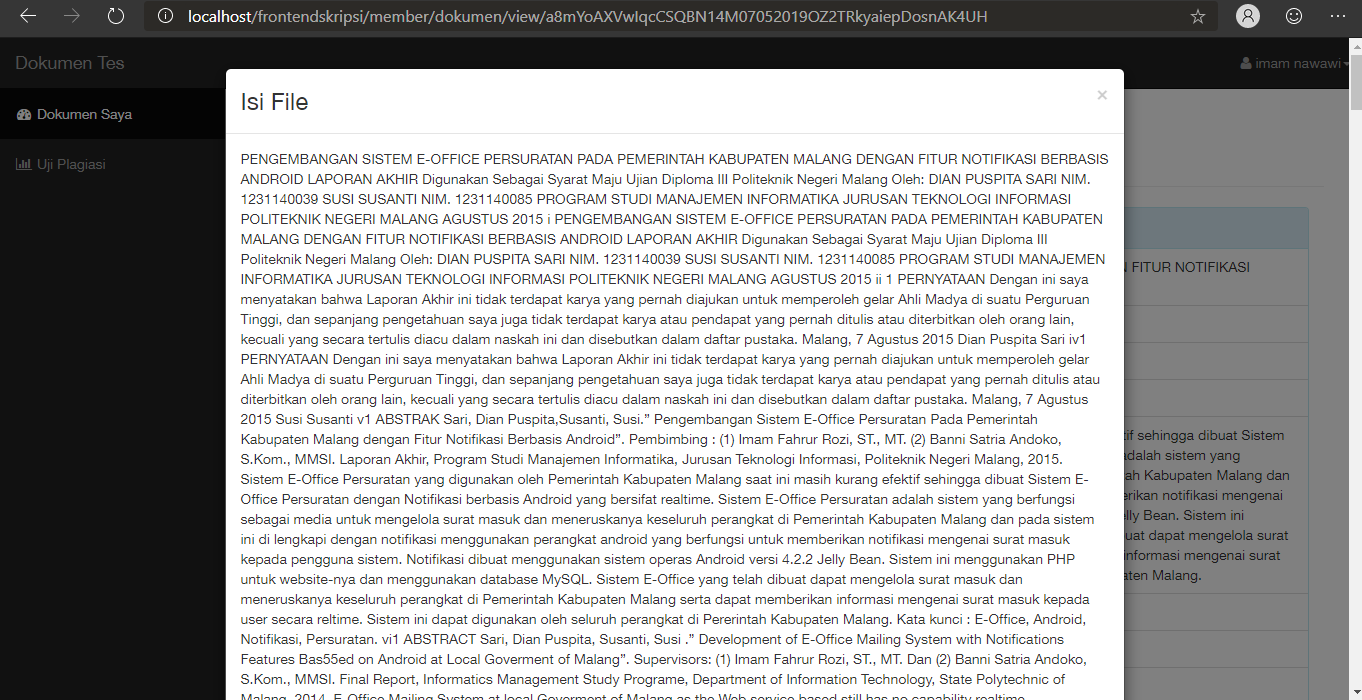
Gambar 5. 12 Halaman Utama Admin Mahasiswa

Apabila ingin melihat data secara lengkap, mahasiswa dapat menekan tombol lihat, maka akan ditampilkan data secara lengkap berdasarkan data yang dipilih seperti ditunjukkan pada gambar 5.13.



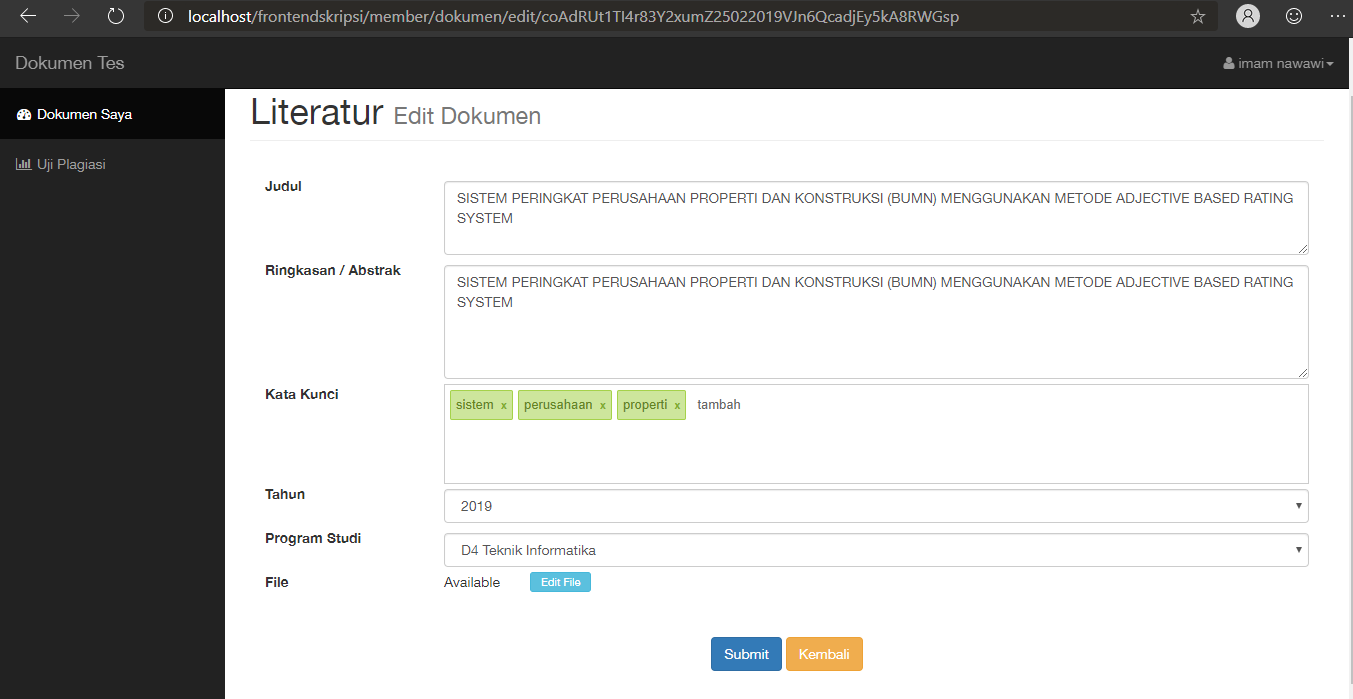
Gambar 5. 13 Halaman Lihat Data

Sedangkan untuk mengetahui isi teks secara keseluruhan, mahasiswa dapat menekan tombol *available* yang artinya teks tersedia dalam *database,* seperti yang ditunjukkan pada gambar 5.14.



Gambar 5. 14 Halaman Lihat teks keseluruhan

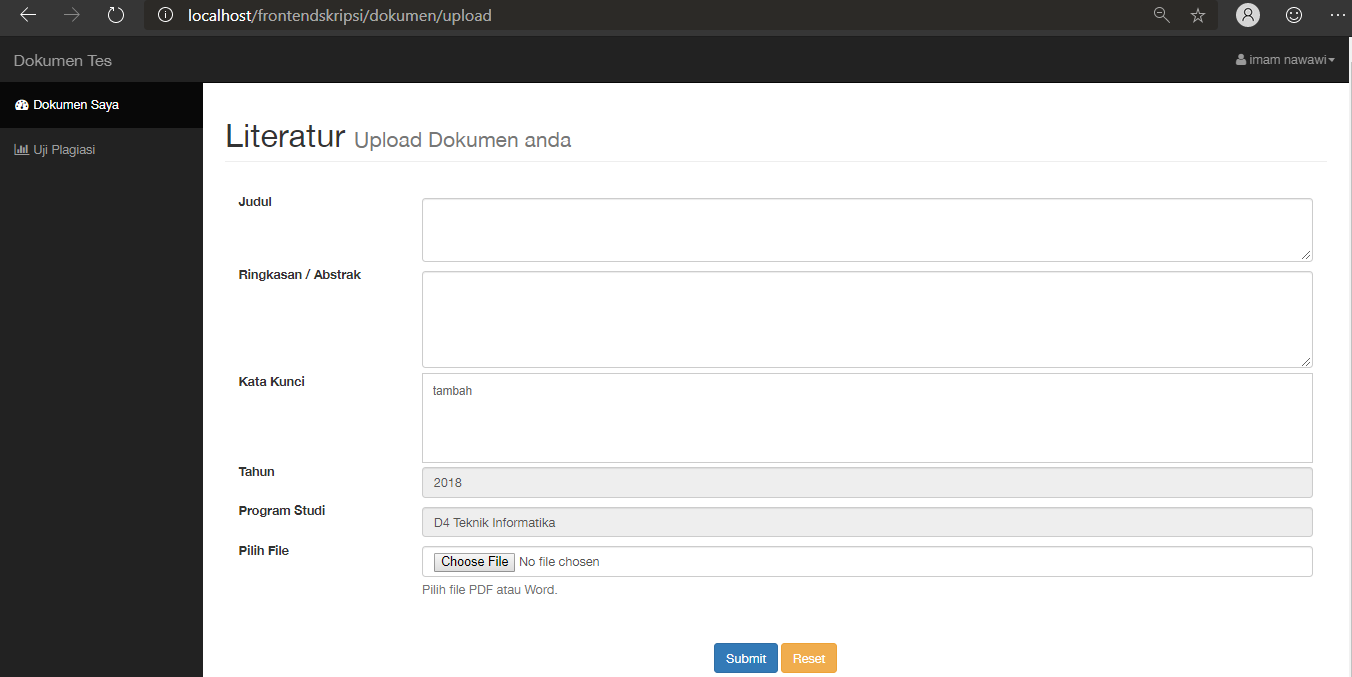
Ketika tombol edit ditekan, maka informasi yang akan dirubah akan tampil dihalaman input data seperti ditunjukkan gambar 5.13 namun dalam keadaan form yang sudah terisi. Mahasiswa cukup merubah data yang diinginkan saja tanpa perlu memasukkan semua informasi dari awal.



Gambar 5. 15 Halaman Edit Data

1. Halaman Upload Dokumen Baru

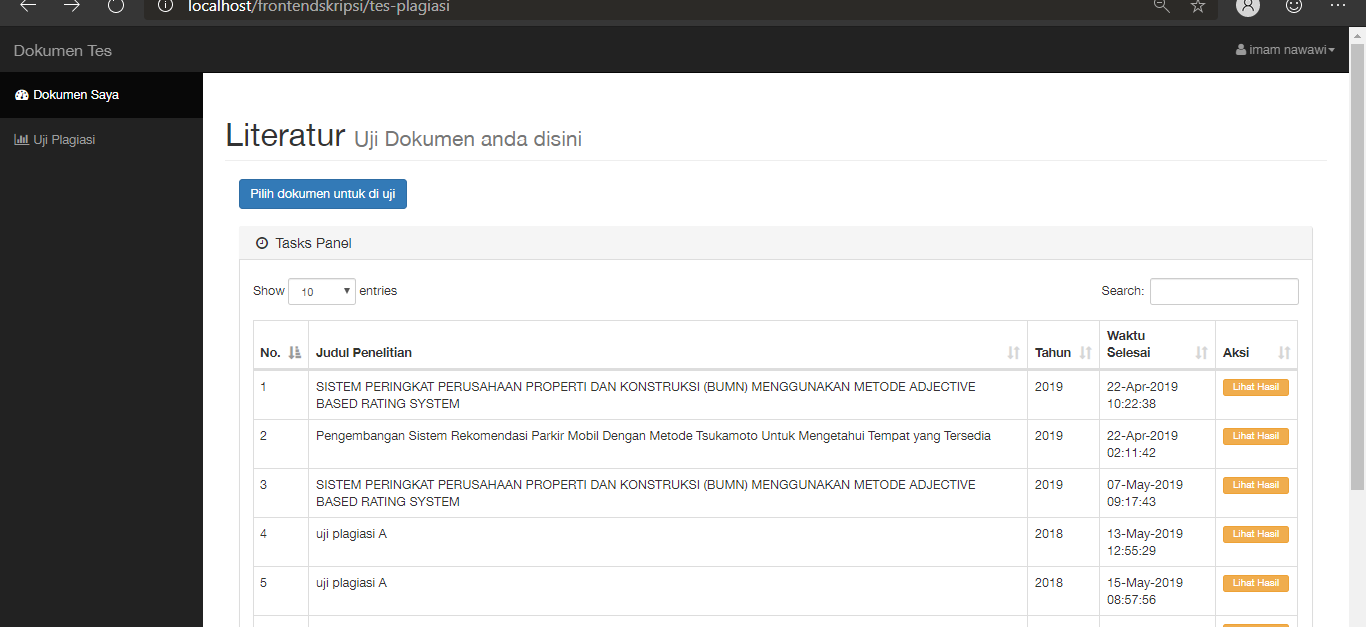
Halaman *upload* dokumen baru merupakan halaman yang berisi *form* untuk menambah dokumen baru yang ditunjukkan pada gambar 5.16. Pada halaman ini *form* upload memiliki komponen antara lain judul, abstrak, kata kunci, file yang ingin di *upload*, sedangkan untuk tahun terisi secara otomatis dari pengaturan panitia dan program studi terisi secara otomatis dari profil mahasiswa. Pada saat halaman *upload* dibuka, mahasiswa dapat mengisi *form* sesuai dengan data dokumen yang akan di upload, apabila *form* telah terisi secara keseluruhan mahasiswa dapat menekan tombol submit yang artinya data akan dikirim ke *database* untuk disimpan.



Gambar 5. 16 Halaman Upload Dokumen Baru

1. Halaman Uji Plagiasi

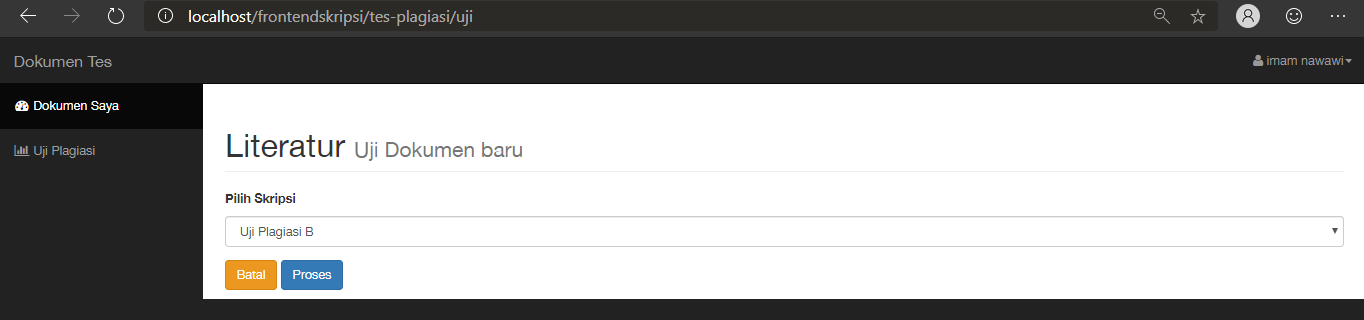
Halaman Uji Plagiasi merupakan halaman yang berisi data-data dokumen yang pernah dilakukan pengujian. Pada halaman ini terdapat fitur pilih dokumen untuk diuji, lihat hasil. Pada saat halaman ini di buka mahasiswa akan diarahkan langsung menuju daftar dokumen yang pernah diuji dimana fitur ini berfungsi untuk menampilkan semua data dari dokumen yang pernah diuji dan sudah tersimpan di *database*. Data tersebut antara lain judul penelitia, tahun dan waktu selesai, pada halaman ini terdapat dua tombol yaitu tombol pilih dokumen yang diuji dan lihat hasil seperti yang ditunjukkan pada gambar 5.17. Tombol pilih dokumen yang diuji berfungsi untuk melakukan tahap *filter* kandidat pembanding sebelum dilakukan pengujian lebih lanjut, sedangkan tombol lihat berfungsi untuk melihat secara keseluruhan data yang terpilih.



Gambar 5. 17 Halaman Uji Plagiasi

1. Halaman Pilih Dokumen Yang Diuji

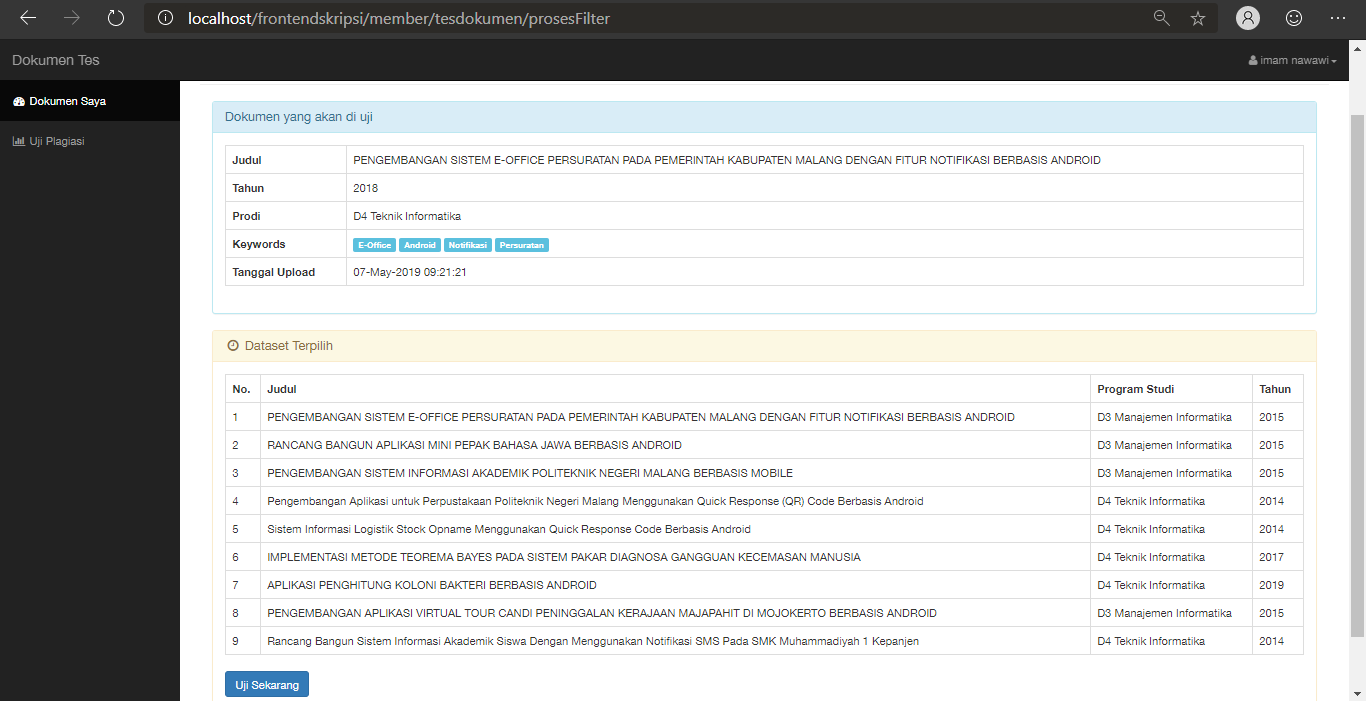
Halaman pilih dokumen yang diuji merupakan halaman yang berisi data-data dokumen mahasiswa yang tersimpan pada *database*. Pada halaman ini terdapat fitur pilihan dokumen untuk diuji, yang dapat dilihat pada gambar 5.18. ketika mahasiswa telah menentukan dokumen pilihan, kemudian menekan tombol proses untuk memulai dilakukan proses *filter* kandidat pembanding berdasarkan kata kunci pada dokumen yang diuji.



Gambar 5. 18 Halaman Pilih Dokumen Uji

1. Halaman Hasil Filter Kandidat Pembanding

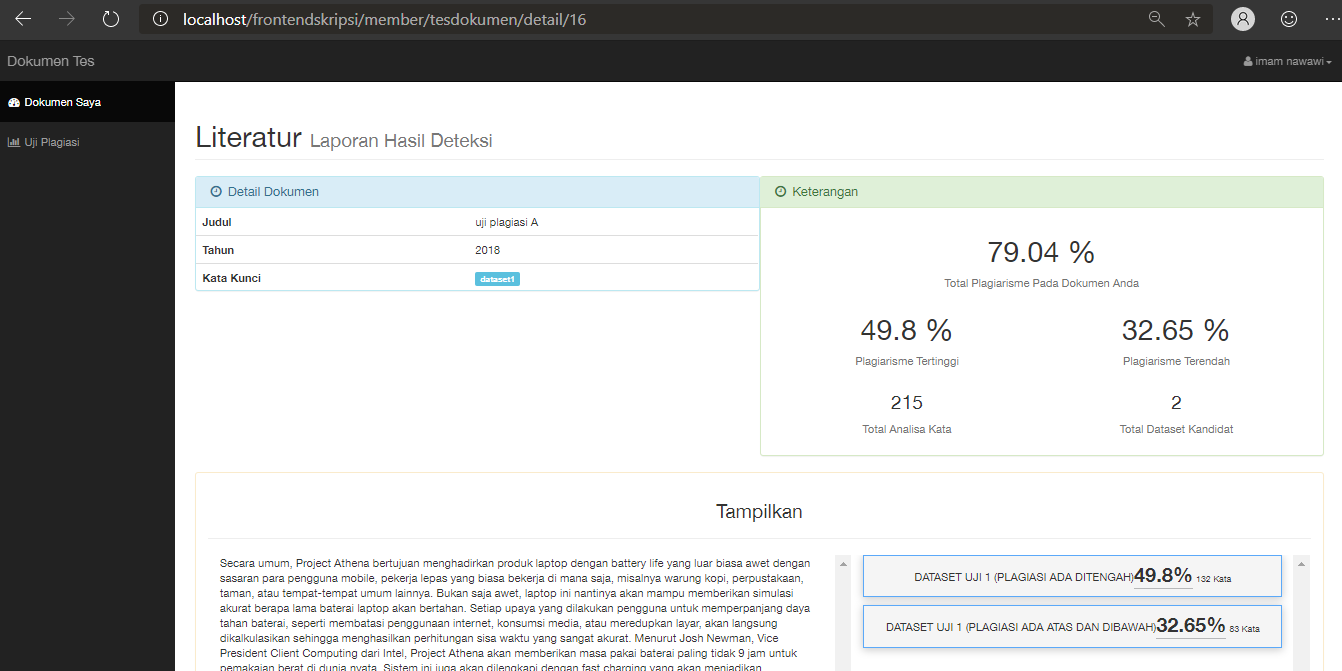
Halaman hasil *filter* kandidat pembanding merupakan halaman yang berisi data-data dari dokumen uji dan kandidat pembanding dari hasil proses *filter kandidat pembanding* yang ditunjukkan pada gambar 5.18. Pada halaman ini mahasiswa tekan tombol uji sekarang untuk memupai proses uji yang ditunjukkan pada gambar 5.19.



Gambar 5. 19 Halaman Hasil Filter Kandidat pembanding

1. Halaman Hasil Uji

Halaman hasil uji merupakan halaman yang berisi data-data dari hasil uji plagiarisme yang telah tersimpan pada *database* seperti yang ditunjukkan pada gambar 5.20. Pada halaman ini mahasiswa dapat mengetahui secara menyeluruh mengenai hasil dari dokumen yang telah diuji.



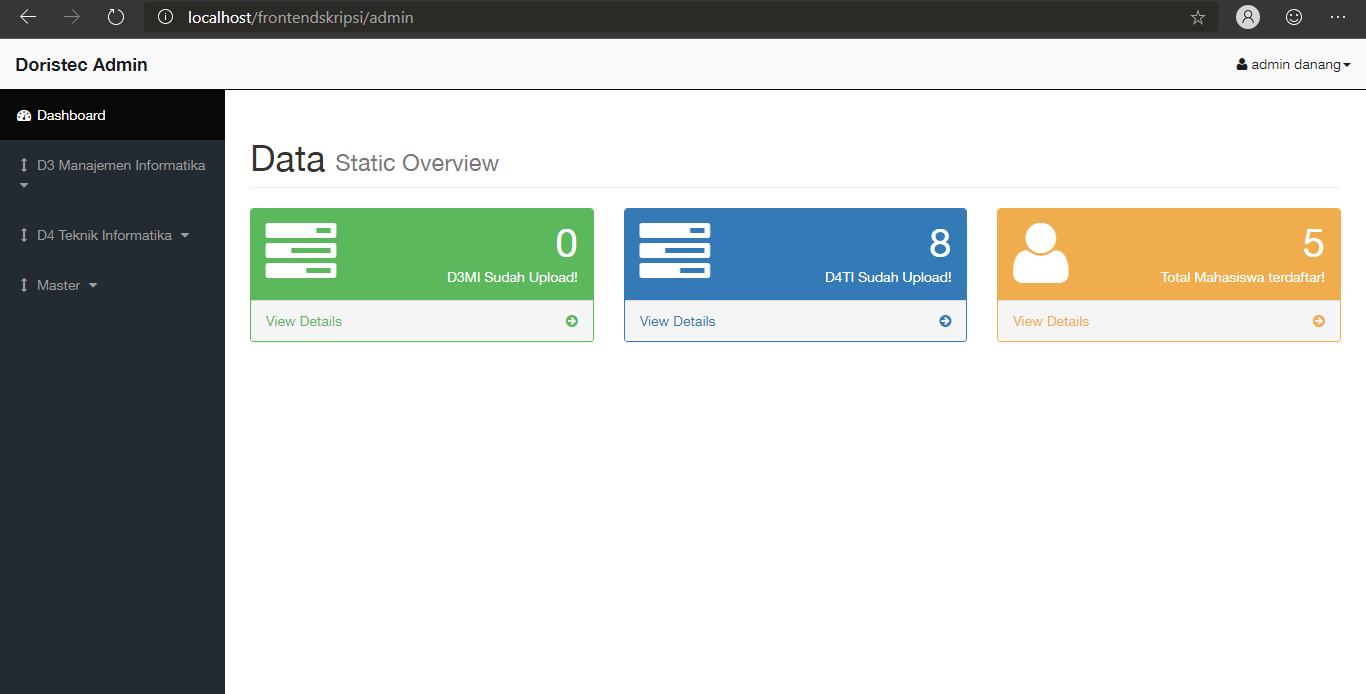
Gambar 5. 20 Halaman Hasil Uji

1. Halaman Panitia

Halaman panitia merupakan halaman admin yang ditujukan bagi pengguna panitia, halaman ini memuat fitur-fitur yang berisikan halaman *dashboard*, daftar mahasiswa D3 manajemen informatika, mahasiswa D3 yang sudah *upload*, daftar mahasiswa D4 teknik informatika, mahasiswa D4 yang sudah *upload*, daftar *dataset*, dan tahun. Untuk desain yang digunakan pada *website* ini memilih desain yang simple agar mudah digunakan. Pemilihan desain menggunakan *theme startbbootstrap-admin custom.*

1. Halaman Dashboard

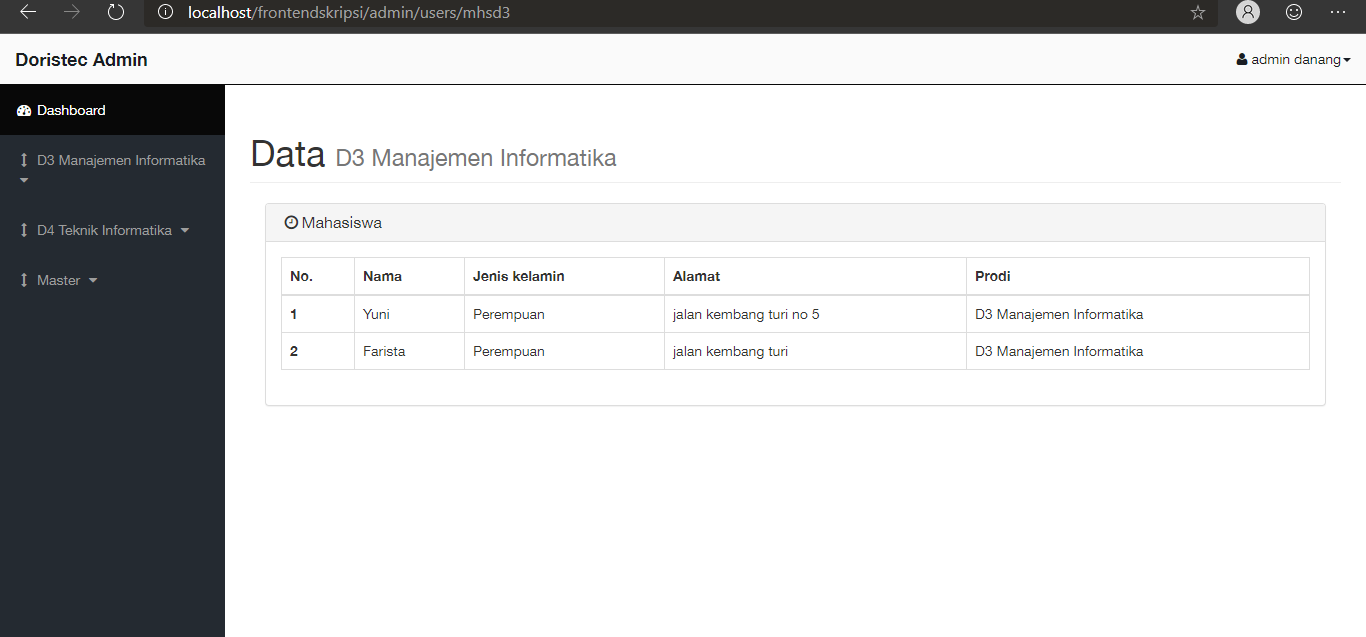
Halaman *dashboard* admin panitia merupakan halaman awal yang ditampilkan dari ketika *login* berhasil dilakukan. Pada halaman ini terdapat fitur mahasiswa D3 Manajemen Informatika, D3 yang sudah *upload* dokumen, mahasiswa D4 Teknik Informatika, D4 yang sudah *upload* dokumen, daftar *dataset*, *upload* dokumen pembandingbaru dan kelola *dataset*, tahun, kelola tahun. Pada saat halaman *dashboard* dibuka panitia akan diarahkan langsung menuju halaman yang dimana fitur ini berfungsi untuk menampilkan semua data berupa grafik yang sudah tersimpan di *database* seperti ditunjukkan pada gambar 5.21.



Gambar 5. 21 Halaman Dashboard Panitia

1. Halaman Mahasiswa D3 Manajemen Informatika

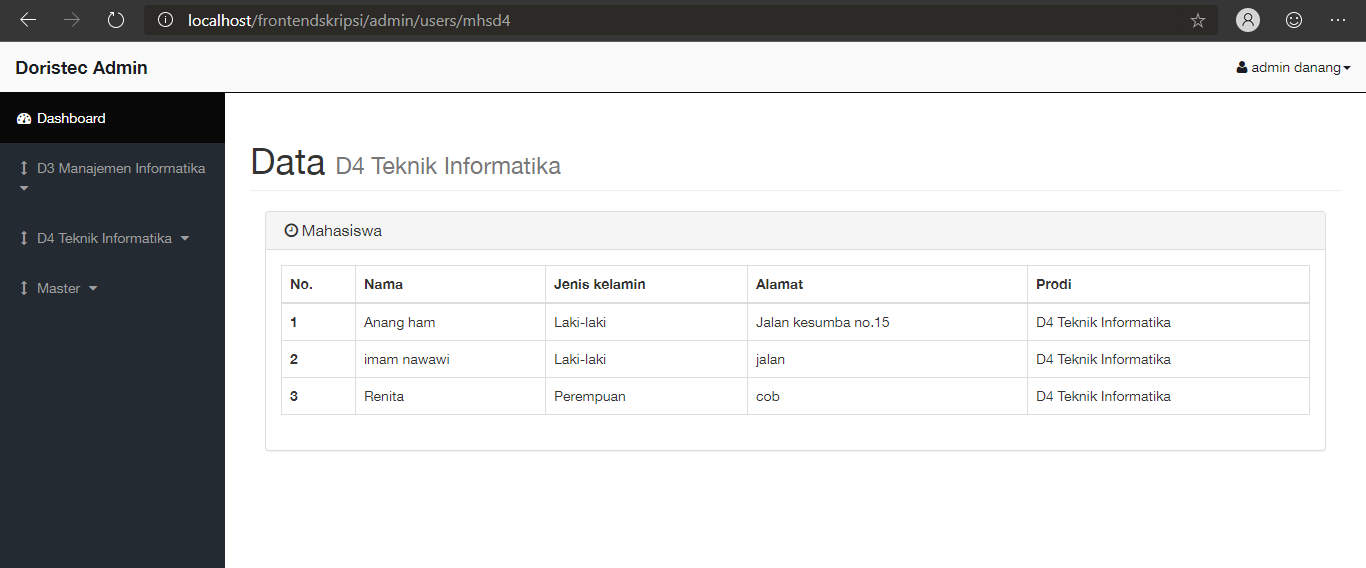
Halaman mahasiswa D3 manajemen informatika merupakan halaman yang berisi data-data mahasiswa D3 yang telah mendaftar pada aplikasi doristec. Pada saat halaman ini di buka panitia akan diarahkan langsung menuju daftar mahasiswa D3 yang pernah mendaftar. Data tersebut antara lain nama, jenis kelamain, alamat, dan program studi seperti yang ditunjukkan pada gambar 5.22.



Gambar 5. 22 Halaman Mahasiswa D3 Manajemen Informatika

1. Halaman Mahasiswa D4 Teknik Informatika

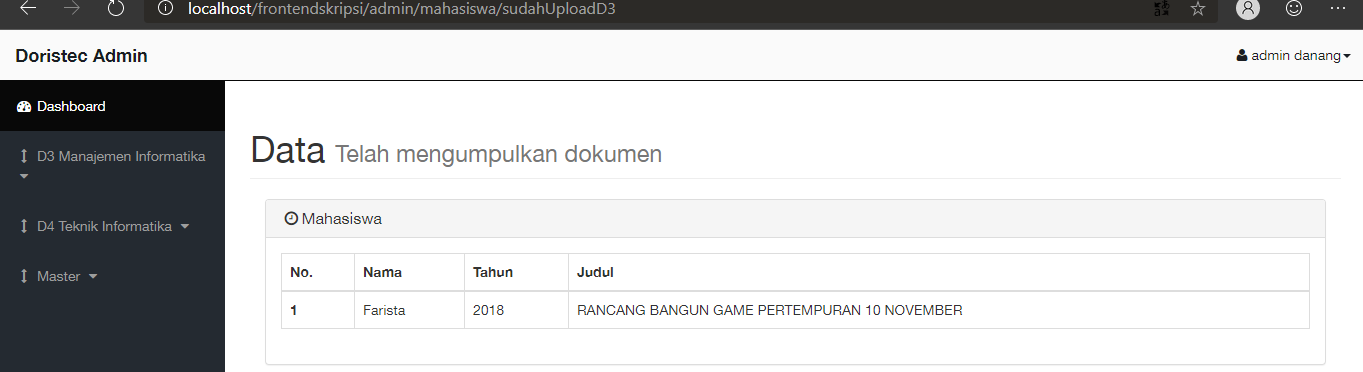
Halaman mahasiswa D4 teknik informatika merupakan halaman yang berisi data-data mahasiswa D4 yang telah mendaftar pada aplikasi doristec. Pada saat halaman ini di buka panitia akan diarahkan langsung menuju daftar mahasiswa D4 yang pernah mendaftar. Data tersebut antara lain nama, jenis kelamain, alamat, dan program studi seperti yang ditunjukkan pada gambar 5.23.



Gambar 5. 23 Halaman mahasiswa D4 teknik informatika

1. Halaman Mahasiswa D3 Yang Sudah Upload

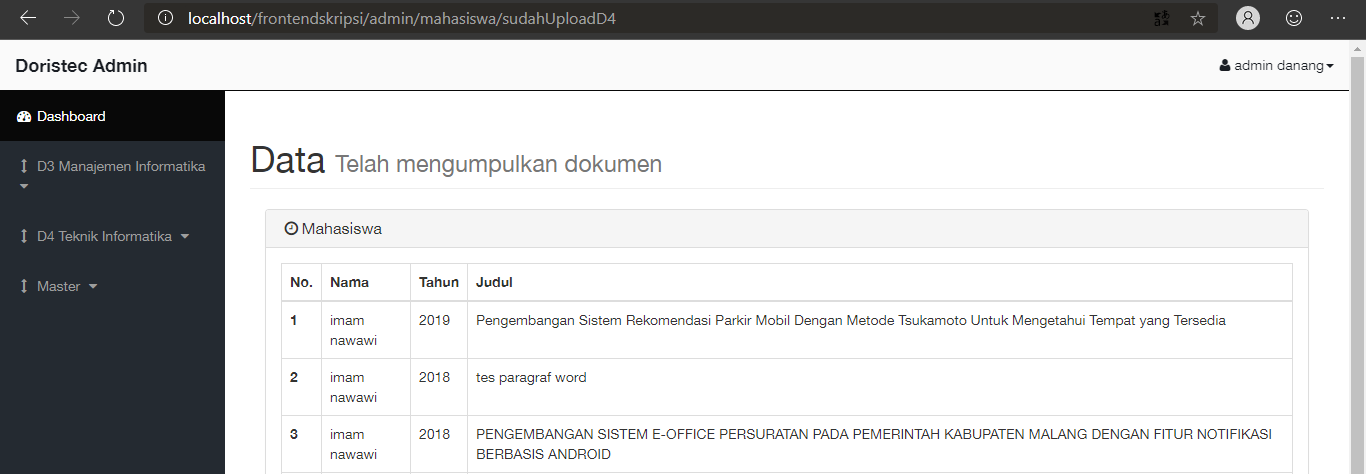
Halaman mahasiswa D3 yang sudah *upload* merupakan halaman yang berisi data-data mahasiswa D3 yang telah *upload* dokumen tugas akhir pada aplikasi doristec. Pada saat halaman ini di buka panitia akan diarahkan langsung menuju daftar mahasiswa D3 beserta dokumen tugas akhir yang telah di *upload*. Data tersebut antara lain nama, tahun, dan judul seperti yang ditunjukkan pada gambar 5.24.



Gambar 5. 24 Halaman mahasiswa D3 yang sudah upload

1. Halaman Mahasiswa D4 Yang Sudah Upload

Halaman mahasiswa D4 yang sudah *upload* merupakan halaman yang berisi data-data mahasiswa D4 yang telah *upload* dokumen skripsi pada aplikasi doristec. Pada saat halaman ini di buka panitia akan diarahkan langsung menuju daftar mahasiswa D4 beserta dokumen skripsi yang telah di *upload*. Data tersebut antara lain nama, tahun, dan judul seperti yang ditunjukkan pada gambar 5.25.

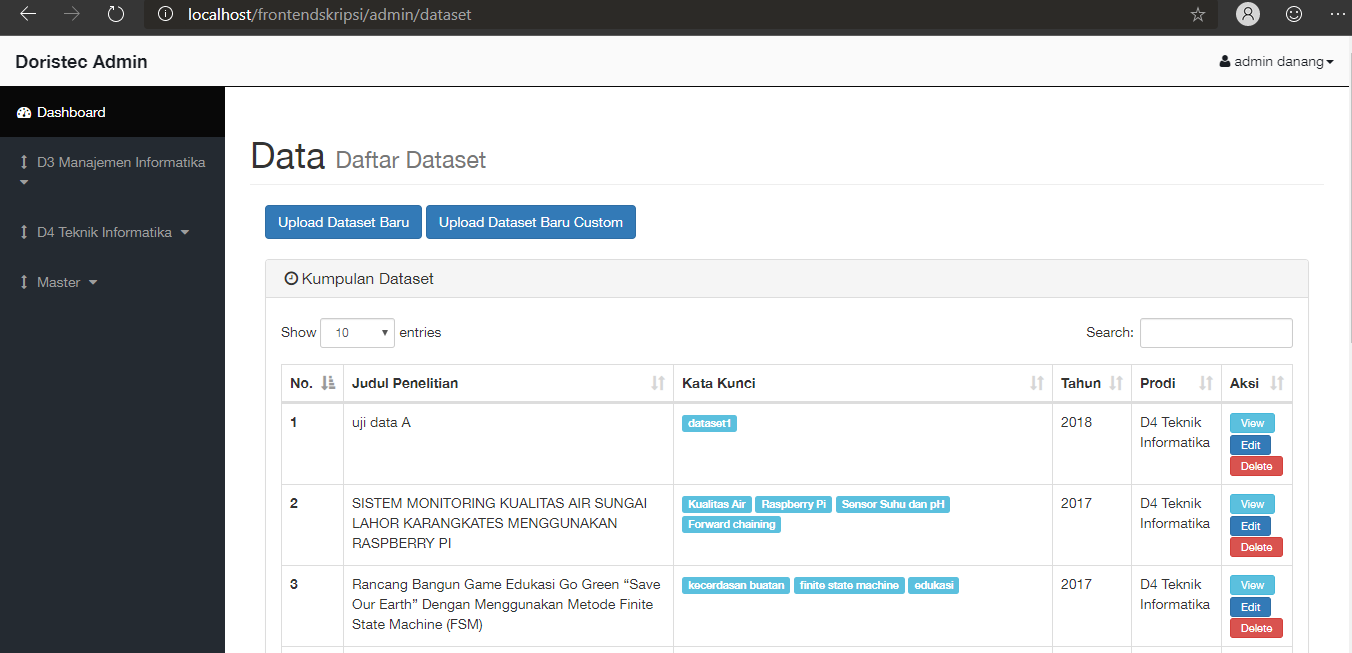


Gambar 5. 25 Halaman mahasiswa D4 yang sudah upload

1. Halaman Dataset

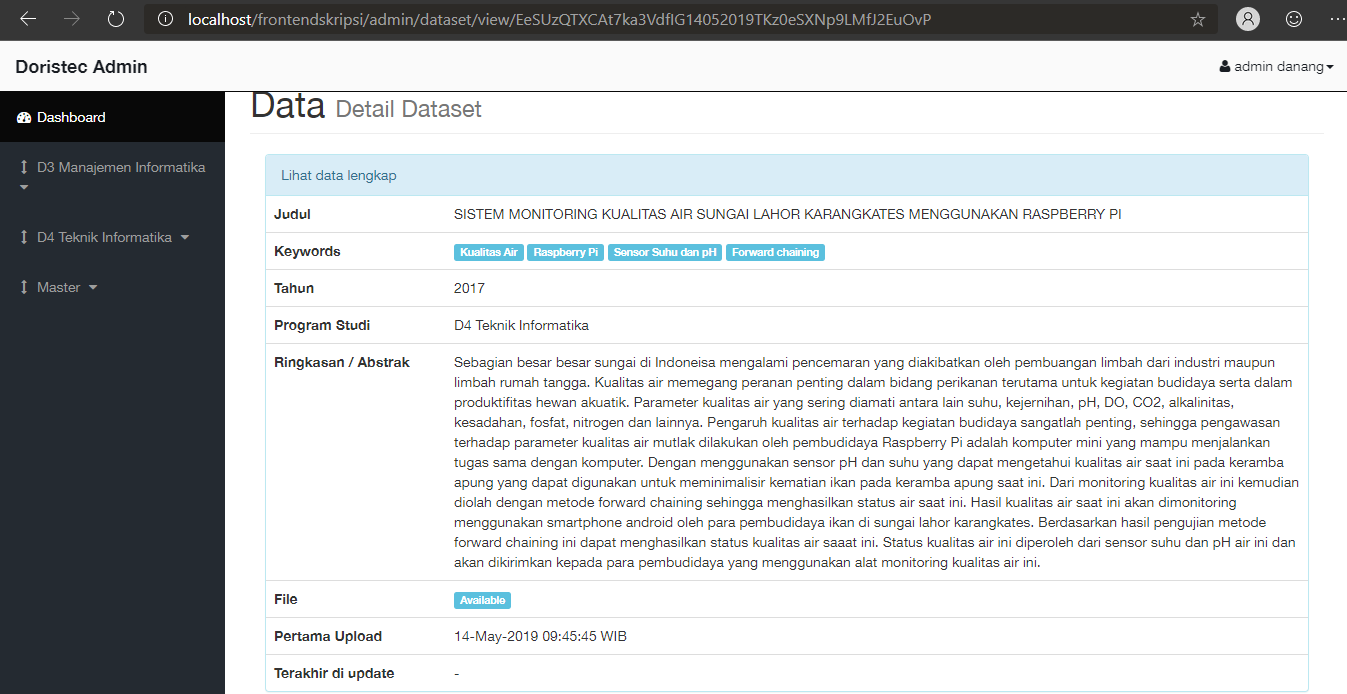
Halaman *dataset* merupakan halaman yang berisi data-data yang telah tersimpan pada *database,* data ini berupa laporan tugas akhir dan laporan skripsi yang digunakan sebagai *dataset* aplikasi ini. Fitur-fitur yang terdapat pada aplikasi ini berfungsi untuk menampilkan dokumen pembanding yang pernah di *upload* dan sudah tersimpan di *database*. Data tersebut antara lain judul penelitian, kata kunci, tahun dan program studi pada halaman ini panitia mempunyai hak akses untuk mengolah data secara keseluruhan seperti yang ditunjukkan pada gambar 5.26, pada halaman ini terdapat tiga tombol yaitu tombol lihat, edit dan hapus. Tombol lihat berfungsi untuk melihat secara keseluruhan data yang terpilih, tombol edit berfungsi untuk mengedit data yang terpilih, sedangkan tombol hapus untuk menghapus data yang terpilih.

Selain itu juga terdapat dua tombol diatas tabel yaitu tombol *upload* dokumen pembanding baru dan *upload* dokumen pembanding baru *custom*. Tombol *upload* dokumen pembanding baru berfungsi untuk menambah koleksi data pembanding baru. Dalam hal ini admin lebih memiliki fitur yang lebih dalam proses upload dokumen baru dengan batasan program studi dan tahun tidak bisa dipilih atau telah otomatis terpilih oleh sistem, sedangkan *upload* dokumen pembanding baru *custom* memiliki *form* yang serupa tetapi program studi dan tahun dapat dipilih sesuai kebutuhan.



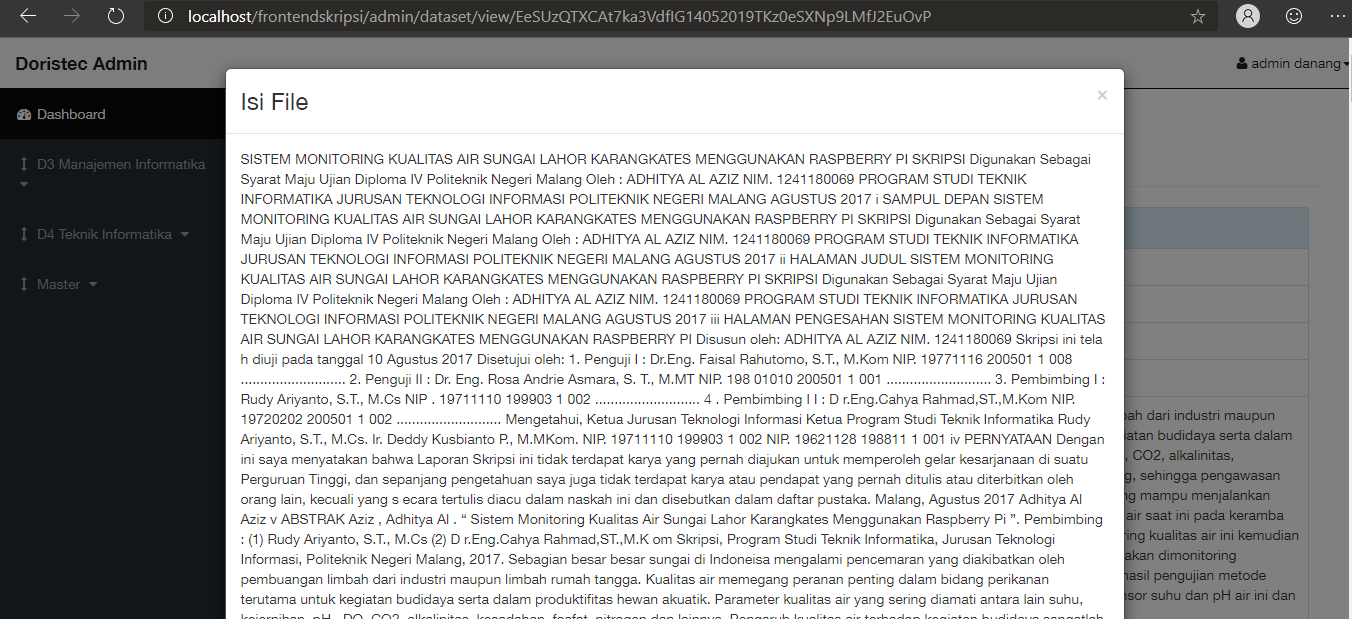
Gambar 5. 26 Halaman dataset

Apabila ingin melihat data secara lengkap, panitia dapat menekan tombol lihat, maka akan ditampilkan data secara lengkap berdasarkan data yang dipilih seperti ditunjukkan pada gambar 5.27.



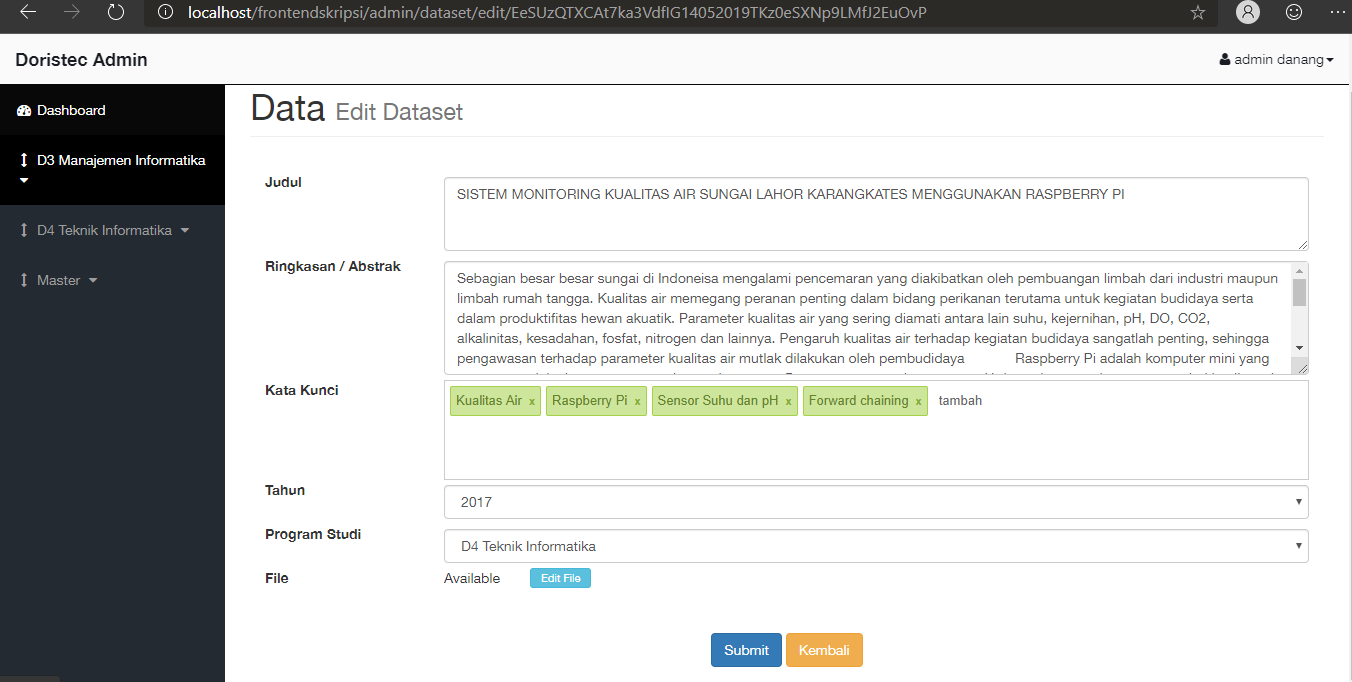
Gambar 5. 27 Halaman Lihat Data Lengkap

Sedangkan untuk mengetahui isi teks secara keseluruhan, panitia dapat menekan tombol available yang artinya teks tersedia dalam *database*, seperti yang ditunjukkan pada gambar 5.28.



Gambar 5. 28 Halaman Lihat Teks Keseluruhan

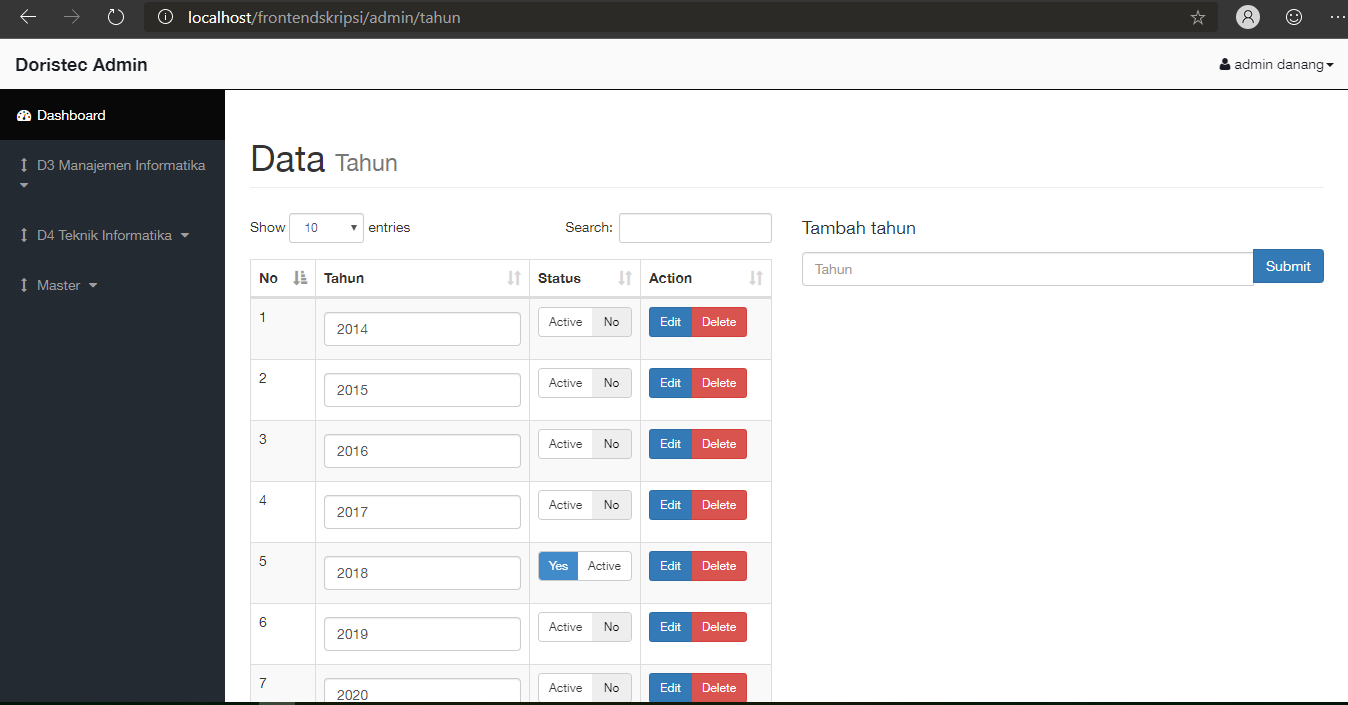
Ketika tombol edit ditekan, maka informasi yang akan dirubah akan tampil dihalaman input data seperti ditunjukkan gambar 5.29 namun dalam keadaan form yang sudah terisi. Panitia cukup merubah data yang diinginkan saja tanpa perlu memasukkan semua informasi dari awal.



Gambar 5. 29 Halaman Edit Data

1. Halaman Tahun

Halaman tahun merupakan halaman yang berisi data-data yang telah tersimpan pada *database,* data ini berupa tahun yang digunakan sebagai penetapan tahun upload dokumen mahasiswa. Fitur-fitur yang terdapat pada aplikasi ini berfungsi untuk menampilkan semua tahun yang pernah di masukkan kedalam sistem dan disimpan di *database*. Data tersebut antara lain nama tahun dan status, pada halaman ini panitia mempunyai hak akses untuk mengolah data tahunsecara keseluruhan seperti yang ditunjukkan pada gambar 5.30, pada halaman ini terdapat dua tombol yaitu tombol edit dan hapus. Tombol edit berfungsi untuk mengedit data yang terpilih namun edit disini tidak perlu berganti halaman lagi melainkan dapat langsung dari tiap baris data tersebut, sedangkan tombol hapus untuk menghapus data yang terpilih. Pada bagian status merupakan opsi *switch* yang berisi kondisi aktif *Yes or No.* Selain itu juga terdapat form tambah tahun pada bagian kanan tabel.



Gambar 5. 30 Halaman Tahun

1. Pengujian

Pengujian merupakan proses pengujian yang dilakukan untuk menguji serta mendokumentasikan hasil pengujian. Pengujian dilakukan menggunakan metode blackbox, metode ini digunakan sebagai demo aplikasi dan mendokumentasi segala aktifitas jalannya aplikasi. Hal ini difungsikan untuk dokumentasi segala kemungkinan yang terjadi ketika aplikasi dijalankan, sehingga dapat diketahui apakah aplikasi telah berjalan dengan baik atau tidak.

1. Pengujian Fungsionalitas Sistem

Pengujian sistem merupakan pengujian yang dilakukan pada aplikasi, pengujian ini dilakukan untuk mengetahui berjalannya aplikasi dapat berfungsi dengan baik. Pengujian pertama dilakukan pada halaman awal aplikasi yang berisi *form* registrasi dan form login, seperti yang ditunjukkan pada tabel 5.2.

Tabel 5. 2 Pengujian Sistem Pada Halaman Awal Aplikasi

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Skenario** | **Hasil yang diharapkan** | **Hasil yang terjadi** | **Keterangan** |
| 1 | Melakukan *input* pada *form* registrasi | Data pengguna dapat diinputkan | Proses berjalan dengan normal | Berhasil |
| 2 | Melakukan *input* pada *form login* | Dapat masuk pada halaman admin panel | Proses berjalan normal. Ketika tombol *login* ditekan | Berhasil |

Dikarenakan aplikasi memiliki dua level yang berbeda yaitu Mahasiswa dan Panitia, maka akan dibuat tabel blackbox yang berbeda yaitu tabel 5. Digunakan untuk level mahasiswa dan tabel 5. Untuk level panitia. Berikut ini adalah metode blackbox untuk pengujian aplikasi :

Tabel 5. 3 Pengujian Sistem Pada Level Mahasiswa

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Skenario** | **Hasil yang diharapkan** | **Hasil yang terjadi** | **Keterangan** |
| 1 | Melakukan pengambilan data dari *database* | Dapat menampilkan semua data uji pengguna | Proses berjalan dengan normal | Berhasil |
| 2 | Menampilkan detail dokumen pengguna | Dapat menampilkan data keseluruhan dari dokumen yang dipilih | Proses berjalan dengan normal | Berhasil |
| 3 | Melakukan perubahan pada data dokumen | Dapat merubah satu atau lebih data dokumen | Proses berjalan dengan normal, ketika tombol *submit* ditekan | Berhasil |
| 4 | Menghapus dokumen | Melakukan hapus data pada *database* | Proses berjalan dengan normal | Berhasil |
| 5 | *Upload* dokumen baru | Menambah dokumen baru dan disimpan ke *database* | Proses berjalan dengan normal, ketika tombol *submit* ditekan | Berhasil |
| 6 | Menampilkan data-data yang pernah dilakukan pengujian | Dapat menampilkan data keseluruhan yang pernah diuji | Proses berjalan dengan normal | Berhasil |

Lanjutan Tabel 5. 3 Pengujian Sistem Pada Level Mahasiswa

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Skenario** | **Hasil yang diharapkan** | **Hasil yang terjadi** | **Keterangan** |
| 7 | Menampilkan detail hasil pengujian | Dapat menampilkan data keseluruhan dari data yang dipilih | Proses berjalan dengan normal | Berhasil |
| 8 | *Filter* Kandidat Pembanding | Pengambilan kandidat pembanding berdasarkan kata kunci | Proses berjalan dengan normal, ketika tombol proses ditekan | Berhasil |
| 9 | Menampilkan data hasil *filter* | Menampilkan hasil *filter* yang digunakan selanjutnya untuk proses LCS | Proses berjalan dengan normal | Berhasil |
| 10 | LCS | Proses membandingkan dokumen uji dengan kandidat pembanding | Proses berjalan dengan normal | Berhasil |
| 11 | Menampilkan profil pengguna | Menampilkan data profil pengguna | Proses berjalan dengan normal | Berhasil |

Tabel 5. 4 Pengujian Sistem Pada Level Panitia

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Skenario** | **Hasil yang diharapkan** | **Hasil yang terjadi** | **Keterangan** |
| 1 | Menampilkan statistik | Menampilkan ststistik pada halaman *dashboard,* yaitu jumlah D3 yang upload, jumlah D4 yang upload, mahasiswa terdaftar | Proses berjalan dengan normal | Berhasil |
| 2 | Menampilkan daftar mahasiswa D3 | Menampilkan data mahasiswa D3 | Proses berjalan dengan normal | Berhasil |
| 3 | Menampilkan daftar mahasiswa D3 | Menampilkan data mahasiswa D3 | Proses berjalan dengan normal | Berhasil |
| 4 | Menampilkan daftar mahasiswa D4 | Menampilkan data mahasiswa D4 | Proses berjalan dengan normal | Berhasil |

Lanjutan Tabel 5. 4 Pengujian Sistem Pada Level Panitia

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Skenario** | **Hasil yang diharapkan** | **Hasil yang terjadi** | **Keterangan** |
| 5 | Menampilkan dokumen mahasiswa D3 | Menampilkan data mahasiswa D3 | Proses berjalan dengan normal | Berhasil |
| 6 | Menampilkan dokumen mahasiswa D4 | Menampilkan data mahasiswa D4 | Proses berjalan dengan normal | Berhasil |
| 7 | Menampilkan data tahun | Menampilkan data tahun dari *database* | Proses berjalan dengan normal | Berhasil |
| 8 | Melakukan *input* pada *form* tambah tahun | Data dapat diinputkan | Proses berjalan dengan normal | Berhasil |
| 9 | Melakukan perubahan pada data tahun | Dapat merubah satu atau lebih data tahun | Proses berjalan dengan normal, ketika tombol *edit* ditekan | Berhasil |
| 10 | Menghapus tahun | Melakukan hapus data pada *database* | Proses berjalan dengan normal | Berhasil |
| 11 | Menampilkan data semua data pada *dataset* | Menampilkan semua data pada *dataset* dari *database* | Proses berjalan dengan normal | Berhasil |
| 12 | Menampilkan detail data | Dapat menampilkan data keseluruhan dari kandidat | Proses berjalan dengan normal | Berhasil |
| 13 | Melakukan perubahan pada data | Dapat merubah satu atau lebih data | Proses berjalan dengan normal, ketika tombol *submit* ditekan | Berhasil |
| 14 | Menghapus data pada *dataset* | Melakukan hapus data pada *database* | Proses berjalan dengan normal | Berhasil |
| 15 | *Upload* dokumen pembanding baru | Menambah dokumen pembanding baru dan disimpan ke *database,* namun program studi, tahun tidak dapat dipilih | Proses berjalan dengan normal, ketika tombol *submit* ditekan | Berhasil |
| 16 | *Upload* dokumen pembanding baru *custom* | Menambah dokumen pembanding baru dengan *input* program studi, tahun dapat dipilih | Proses berjalan dengan normal, ketika tombol *submit* ditekan | Berhasil |

Lanjutan Tabel 5. 4 Pengujian Sistem Pada Level Panitia

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Skenario** | **Hasil yang diharapkan** | **Hasil yang terjadi** | **Keterangan** |
|  |  | dan disimpan ke *database* |  |  |
| 17 | Menampilkan hasil dokumen yang telah diuji | Menampilkan hasil dokumen mahasiswa yang telah diuji | Proses berjalan dengan normal | Berhasil |
| 18 | Membuat *ranking* pada kandidat pembanding | Dapat meranking nilai plagiarisme pada kandidat pembanding*,* hingga ditemukan nilai tertinggi dan nilai terendah plagiarismenya | Proses berjalan dengan normal | Berhasil |
| 19 | Menampilkan detail pada setiap kandidat pembanding yang ditemukan | Dapat menampilkan nilai plagiarisme, kalimat-kalimat yang ditemukan | Proses berjalan dengan normal | Berhasil |
| 20 | Menampilkan profil pengguna | Menampilkan data profil pengguna | Proses berjalan dengan normal | Berhasil |

1. Pengujian Metode

Pengujian Metode merupakan pengujian yang dilakukan pada aplikasi dengan terfokus pada metode yang diterapkan pada aplikasi, pengujian ini dilakukan untuk mengetahui berjalannya metode LCS dapat berfungsi dengan baik. Pengujian manual dengan metode LCS akan menggunakan tiga studi kasus yang diberikan. Pada setiap studi kasus memiliki beberapa bentuk adanya plagiarisme, bentuk proses perhitungan yang dilakukan dapat digambarkan pada tahapan sebagai berikut :

1. Telah ditentukan jumlah dokumen yang diuji yaitu dua dokumen yang disebut satu dokumen uji dan satu kandidat pembanding:
   1. S1 : Dokumen terpanjang
   2. S2 : Dokumen yang lebih pendek
2. Menentukan dokumen terpanjang dari data uji dan kandidat pembanding
   1. Dokumen terpanjang disimpan pada variabel S1
   2. Dokumen yang lebih pendek disimpan pada variabel S2
3. Melakukan pencarian kata dengan urutan kata terpanjang mengunakan rumus yang ditunjukkan pada gambar 5.31 berikut ini.



Gambar 5. 31 Rumus LCS

1. Simpan hasil pencarian kata terpanjang pada variabel
2. Menghitung prosentase plagiarisme pada setiap kandidat pembanding
3. Jika hasil pencarian kata terpanjang memiliki lebih dari tiga kata, maka ulangi langkah C. jika kurang dari tiga kata dan kandidat pembanding lebih dari nol, maka lanjutkan pencarian pada kandidat pembanding selanjutnya dan mengulangi langkah B. Apabila kandidat pembanding tidak lebih dari nol maka proses menghitung menggunakan metode LCS selesai.
4. Hasil disimpan pada *database*.

Berikut ini pengujian menggunakan tiga studi kasus yang telah disediakan berupa tahapan sebagai berikut :

1. Kasus Pertama

Pengujian pada kasus pertama merupakan pengujian yang dilakukan dengan memberikan dokumen yang sama antara kandidat pembanding dan data uji, berikut ditunjukkan pada tabel 5.5.

Tabel 5. 5 Kasus Pertama Pengujian Metode

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Isi Dokumen** | **Jumlah Kata** | **Keterangan** |
| 1 | Secara umum, Project Athena bertujuan menghadirkan produk laptop dengan *battery life*yang luar biasa awet dengan sasaran para pengguna *mobile*, pekerja lepas yang biasa | 84 Kata | Dokumen Uji |

Lanjutan Tabel 5. 5 Kasus Pertama Pengujian Metode

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Isi Dokumen** | **Jumlah Kata** | **Keterangan** |
|  | bekerja di mana saja, misalnya warung kopi, perpustakaan, taman, atau tempat-tempatumum lainnya. Bukan saja awet, laptop ini nantinya akan mampu memberikan simulasi akurat berapa lama baterai laptop akan bertahan. Setiap upaya yang dilakukan pengguna untuk memperpanjang daya tahan baterai, seperti membatasi penggunaan internet, konsumsi media, atau meredupkan layar, akan langsung dikalkulasikan sehingga menghasilkan perhitungan sisa waktu yang sangat akurat. |  |  |
| 2 | Secara umum, Project Athena bertujuan menghadirkan produk laptop dengan *battery life*yang luar biasa awet dengan sasaran para pengguna *mobile*, pekerja lepas yang biasa bekerja di mana saja, misalnya warung kopi, perpustakaan, taman, atau tempat-tempat umum lainnya. Bukan saja awet, laptop ini nantinya akan mampu memberikan simulasi akurat berapa lama baterai laptop akan bertahan. Setiap upaya yang dilakukan pengguna untuk memperpanjang daya tahan baterai, seperti membatassi penggunaan internet, konsumsi media, atau meredupkan layar, akan langsung dikalkulasikan sehingga menghasilkan perhitungan sisa waktu yang sangat akurat. | 84 Kata | Kandidat pembanding |

Tabel 5. 6 Perhitungan Manual Pada Studi Kasus Pertama

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Kategori Dokumen** | **Jumlah Kata** | **Jumlah ditemukan kata** | **Jumlah kalimat ditemukan** |
| 1 | Dokumen Uji | 84 Kata |  |  |
| 2 | Kandidat pembanding | 84 Kata | 84 kata | 1 |

1. Kasus Kedua

Pengujian pada kasus kedua merupakan pengujian yang dilakukan dengan memberikan dokumen data uji dengan memiliki beberapa kemiripan kalimat pada tiga kandidat pembandingyang diujikan, berikut ditunjukkan pada tabel 5.7.

Tabel 5. 7 Kasus Kedua Pengujian Metode

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Isi Dokumen** | **Jumlah Kata** | **Keterangan** |
| 1 | Secara umum, Project Athena bertujuan menghadirkan produk laptop dengan battery life yang luar biasa awet dengan sasaran para pengguna mobile, pekerja lepas yang biasa bekerja di mana saja, misalnya warung kopi, perpustakaan, taman, atau tempat-tempat umum lainnya. Bukan saja awet, laptop ini nantinya akan mampu memberikan simulasi akurat berapa lama baterai laptop akan bertahan. Setiap upaya yang dilakukan pengguna untuk memperpanjang daya tahan baterai, seperti membatasi penggunaan internet, konsumsi media, atau meredupkan layar, akan langsung dikalkulasikan sehingga menghasilkan perhitungan sisa waktu yang sangat akurat. Menurut Josh Newman, Vice President Client Computing dari Intel, Project Athena akan memberikan masa pakai baterai paling tidak 9 jam untuk pemakaian berat di dunia nyata. Sistem ini juga akan dilengkapi dengan fast charging yang akan menjadikan pengguna mobile tidak akan lagi kesulitan untuk menambah daya baterainya di mana pun. Newman juga menambahkan bahwa solusi untuk mencapai hal ini bukanlah dengan memperbesar kapasitas baterai, karena ini nanti akan merusak pengalaman pengguna yang menginginkan laptop tipis premium. Intel akan mencapai ini dengan efisiensi daya yang optimal dan melakukan tuning yang tepat terhadap keseluruhan komponen laptop. Penyesuaian ini juga akan dibantu oleh sistem A.I, misalnya, dengan kamera yang didukung A.I, perangkat akan dapat mengenali apakah pengguna berada di depan layar monitor atau tidak, dan mengambil keputusan untuk meredupkan layar atau bahkan membawa laptop ke posisi sleep.  Intel juga sudah mengambil langkah lebih jauh dengan bekerja bersama para pembuat komponen komputer untuk menghadirkan perangkat Project Athena generasi pertama. | 262 Kata | Dokumen Uji |

Lanjutan Tabel 5. 7 Kasus Kedua Pengujian Metode

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Isi Dokumen** | **Jumlah Kata** | **Keterangan** |
|  | Jika perangkat prototype tersedia, Intel nantinya juga akan membuka diri terhadap manufaktur PC, sehingga lebih banyak manufaktur yang dapat memanfaatkan teknologi Intel ini. |  |  |
| 2 | Ini berarti bahwa pengguna Windows PowerShell dan PowerShell Core akan dapat menggunakan versi yang sama dari PowerShell untuk mengotomatisasi seluruh Windows, Linux, dan macOS dan pada Windows, dan pengguna PowerShell 7 akan memiliki tingkat kompatibilitas yang sangat tinggi dengan modul Windows PowerShell yang mereka andalkan. Setiap upaya yang dilakukan pengguna untuk memperpanjang daya tahan baterai, seperti membatasi penggunaan internet, konsumsi media, atau meredupkan layar, akan langsung dikalkulasikan sehingga menghasilkan perhitungan sisa waktu yang sangat akurat.  Dengan kata lain, aplikasi UWP dan Win32 akan terus hidup berdampingan dengan paket fitur yang sama dan dengan Microsoft Store masih berfungsi sebagai saluran distribusi yang penting. Meskipun ini pada akhirnya bisa mendorong pengembang dan pengguna kembali ke aplikasi Win32, tentu akan menarik untuk melihat bagaimana Microsoft mengelola agar aplikasi UWP tetap relevan sebagai bagian dari pendekatan yang diperbarui. | 134 Kata | Kandidat pembanding 1 |
| 3 | Dari perspektif teknis, ini berarti semua fitur akan tersedia untuk semua aplikasi, terlepas dari apakah itu aplikasi UWP atau Win32. Dengan hal ini, pada gilirannya, akan memberi pengguna pengalaman yang lebih konsisten di seluruh perangkat, yang merupakan tujuan jangka panjang Microsoft karena berencana untuk memperluas ekosistem Windows 10 dengan SKU baru dan kategori perangkat.  PowerShell Core akan dapat menggunakan versi yang sama dari PowerShell untuk mengotomatisasi seluruh Windows, Linux, | 197 Kata | Kandidat pembanding 2 |

Lanjutan Tabel 5. 7 Kasus Kedua Pengujian Metode

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Isi Dokumen** | **Jumlah Kata** | **Keterangan** |
|  | dan macOS dan pada Windows, dan pengguna PowerShell 7 akan memiliki tingkat kompatibilitas yang sangat tinggi dengan modul Windows PowerShell yang mereka andalkan. Newman juga menambahkan bahwa solusi untuk mencapai hal ini bukanlah dengan memperbesar kapasitas baterai, karena ini nanti akan merusak pengalaman pengguna yang menginginkan laptop tipis premium. Intel akan mencapai ini dengan efisiensi daya yang optimal dan melakukan tuning yang tepat terhadap keseluruhan komponen laptop. Penggunaan PowerShell Core telah tumbuh secara signifikan dalam dua tahun terakhir. Secara khusus, sebagian besar pertumbuhan kami berasal dari penggunaan Linux, statistik yang menggembirakan mengingat investasi kami dalam membuat PowerShell layak untuk lintas platform.  Dengan kata lain, aplikasi UWP dan Win32 akan terus hidup berdampingan dengan paket fitur yang sama dan dengan Microsoft Store masih berfungsi sebagai saluran distribusi yang penting. |  |  |
| 4 | Dengan kata lain, aplikasi UWP dan Win32 akan terus hidup berdampingan dengan paket fitur yang sama dan dengan Microsoft Store masih berfungsi sebagai saluran distribusi yang penting. Meskipun ini pada akhirnya bisa mendorong pengembang dan pengguna kembali ke aplikasi Win32, tentu akan menarik untuk melihat bagaimana Microsoft mengelola agar aplikasi UWP tetap relevan sebagai bagian dari pendekatan yang diperbarui.  Intel juga sudah mengambil langkah lebih jauh dengan bekerja bersama para pembuat komponen komputer untuk menghadirkan perangkat Project Athena generasi pertama. dengan efisiensi daya yang optimal dan melakukan *tuning*yang tepat terhadap keseluruhan komponen laptop. Penyesuaian ini juga akan dibantu oleh sistem A.I, | 156 Kata | Kandidat pembanding 3 |

Tabel 5. 8 Perhitungan Manual Pada Studi Kasus Kedua

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Kategori Dokumen** | **Jumlah Kata** | **Jumlah kata ditemukan** | **Jumlah kalimat ditemukan** |
| 1 | Dokumen Uji | 262 Kata |  |  |
| 2 | Kandidat pembanding 1 | 134 Kata | 29 kata | 1 |
| 3 | Kandidat pembanding 2 | 197 kata | 44 kata | 1 |
| 4 | Kandidat pembanding 3 | 156 kata | 41 kata | 1 |

1. Kasus Ketiga

Pengujian pada kasus ketiga merupakan pengujian yang dilakukan dengan memberikan dokumen data uji dengan memiliki beberapa kemiripan kalimat pada dua kandidat pembanding, berikut ditunjukkan pada tabel 5.9.

Tabel 5. 9 Kasus Ketiga Pengujian Metode

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Isi Dokumen** | **Jumlah Kata** | **Keterangan** |
| 1 | Penggunaan PowerShell Core telah tumbuh secara signifikan dalam dua tahun terakhir. Secara khusus, sebagian besar pertumbuhan kami berasal dari penggunaan Linux, statistik yang menggembirakan mengingat investasi kami dalam membuat PowerShell layak untuk lintas platform. Namun, kami juga dapat dengan jelas melihat bahwa penggunaan Windows kami belum tumbuh secara signifikan, mengejutkan mengingat bahwa PowerShell dipopulerkan pada platform Windows. Ini berarti bahwa pengguna Windows PowerShell dan PowerShell Core akan dapat menggunakan versi yang sama dari PowerShell untuk mengotomatisasi seluruh Windows, Linux, dan macOS dan pada Windows, dan pengguna PowerShell 7 akan memiliki tingkat kompatibilitas yang sangat tinggi dengan modul Windows PowerShell yang mereka andalkan.  Dengan kata lain, Microsoft ingin meletakkan semua platform pada versi yang | 183 Kata | Dokumen Uji |

Lanjutan Tabel 5. 9 Kasus Ketiga Pengujian Metode

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Isi Dokumen** | **Jumlah Kata** | **Keterangan** |
|  | sama dari PowerShell untuk kompatibilitas lintas-platform yang lebih baik. Microsoft juga membuat perubahan pada Support Life Cycle PowerShell. Microsoft juga akan merilis LTS (Long Term Servicing) dan versi PowerShell 7 non-LTS untuk menyelaraskannya dengan support life cycle .NET Core.  Sekarang Microsoft telah meluncurkan versi preview PowerShell 7. Microsoft mengharapkan untuk meluncurkan preview baru setiap bulan. Ketersediaan Umum PowerShell akan membutuhkan waktu karena Microsoft harus menyelaraskan perilisan dengan .NET Core 3.0. |  |  |
| 2 | Secara umum, Project Athena bertujuan menghadirkan produk laptop dengan battery life yang luar biasa awet dengan sasaran para pengguna mobile, pekerja lepas yang biasa bekerja di mana saja, misalnya warung kopi, perpustakaan, taman, atau tempat-tempat umum lainnya. Namun, kami juga dapat dengan jelas melihat bahwa penggunaan Windows kami belum tumbuh secara signifikan, mengejutkan mengingat bahwa PowerShell dipopulerkan pada platform Windows. Bukan saja awet, laptop ini nantinya akan mampu memberikan simulasi akurat berapa lama baterai laptop akan bertahan. Setiap upaya yang dilakukan pengguna untuk memperpanjang daya tahan baterai, seperti membatasi penggunaan internet, konsumsi media, atau meredupkan layar, akan langsung dikalkulasikan sehingga menghasilkan perhitungan sisa waktu yang sangat akurat.  Dengan kata lain, aplikasi UWP dan Win32 akan terus hidup berdampingan dengan paket fitur yang sama dan dengan Microsoft Store masih berfungsi sebagai saluran distribusi yang penting.  Intel juga sudah mengambil langkah lebih jauh dengan bekerja bersama para pembuat komponen komputer untuk menghadirkan perangkat Project Athena generasi pertama. | 198 Kata | Kandidat pembanding 1 |

Lanjutan Tabel 5. 9 Kasus Ketiga Pengujian Metode

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Isi Dokumen** | **Jumlah Kata** | **Keterangan** |
|  | Dengan kata lain, Microsoft ingin meletakkan semua platform pada versi yang sama dari PowerShell untuk kompatibilitas lintas-platform yang lebih baik. Jika perangkat prototype tersedia, Intel nantinya juga akan membuka diri terhadap manufaktur PC, sehingga lebih banyak manufaktur yang dapat memanfaatkan teknologi Intel ini. |  |  |
| 3 | Sistem ini juga akan dilengkapi dengan *fast charging*yang akan menjadikan pengguna *mobile*tidak akan lagi kesulitan untuk menambah daya baterainya di mana pun. PowerShell dipopulerkan pada platform Windows Newman juga menambahkan bahwa solusi untuk mencapai hal ini bukanlah dengan memperbesar kapasitas baterai, karena ini nanti akan merusak pengalaman pengguna yang menginginkan laptop tipis premium. Microsoft ingin meletakkan semua platform pada versi yang sama intel akan mencapai ini dengan efisiensi daya yang optimal dan melakukan *tuning*yang tepat terhadap keseluruhan komponen laptop. Penyesuaian ini juga akan dibantu oleh sistem A.I, misalnya, pengguna PowerShell 7 akan memiliki tingkat kompatibilitas dengan kamera yang didukung A.I, perangkat akan dapat mengenali apakah pengguna berada di depan layar monitor atau tidak, dan mengambil keputusan untuk meredupkan layar atau bahkan membawa laptop ke posisi *sleep.*  Intel juga sudah mengambil langkah lebih jauh dengan bekerja bersama para pembuat komponen komputer untuk menghadirkan perangkat Project Athena generasi pertama. Penggunaan PowerShell Core telah tumbuh secara signifikan. Jika perangkat *prototype* tersedia, Intel nantinya juga akan membuka diri terhadap manufaktur PC, sehingga lebih banyak manufaktur yang dapat memanfaatkan teknologi Intel ini. | 180 Kata | Kandidat pembanding 2 |

Tabel 5. 10 Perhitungan Manual Pada Studi Kasus Ketiga

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No** | **Kategori Dokumen** | **Jumlah Kata** | **Jumlah kata disetiap LCS** | **Jumlah kalimat ditemukan** | **Total Kata** |
| 1 | Dokumen Uji | 183 Kata |  |  |  |
| 2 | Kandidat pembanding1 | 198 Kata | 23 kata, 18 kata, 6 kata | 3 | 47 kata |
| 3 | Kandidat pembanding2 | 180 kata | 9 kata, 6 kata, 5 kata, 4 kata | 4 | 24 kata |

BAB VI. HASIL DAN PEMBAHASAN

Setelah dilakukan pengujian pada bab V, pada bab ini digunakan sebagai dokumentasi hasil dilakukan pengujian setelah implementasi sistem. Hasil dan pembahasan dilakukan untuk mengetahui kesesuaian fungsional aplikasi dengan perancangan yang telah dibuat pada bab IV.

1. Hasil Pengujian

Untuk mengetahui validitas dari hasil perhitungan sistem, sehingga diperlukan pengujian sistem dengan dilakukan perhitungan pada metode menggunakan tiga studi kasus dan metode pengujian *recall precision*. *Recall* merupakan tingkat keberhasilan dari sistem dalam menemukan kembali sebuah informasi. Sedangkan *precision* merupakan tingkat ketepatan antara informasi yang diminta oleh pengguna dengan jawaban yang diminta oleh sistem. Berikut ini hasil dari pengujian menggunakan metode LCS pada aplikasi.

Tabel 6. 1 Tabel Matrik Recall Precision

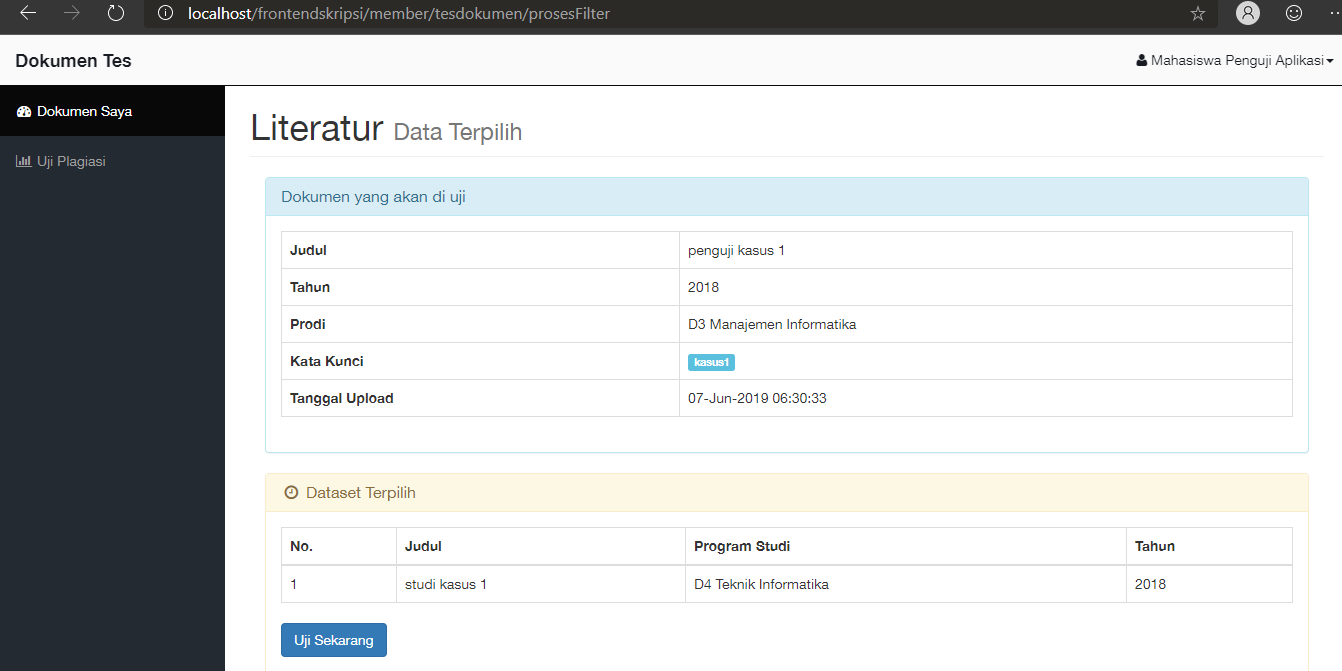
|  |  |  |  |
| --- | --- | --- | --- |
|  | | Nilai Sebenarnya | |
| TRUE | FALSE |
| Nilai Prediksi | TRUE | TP  (True Positive)  *Correct result* | FP  (False Positive)  *Unexpected result* |
| FALSE | FN  (False Negative)  *Missing result* | TN  (True Negative)  *Correct absence of result* |

*Recall =*

*Precision =*

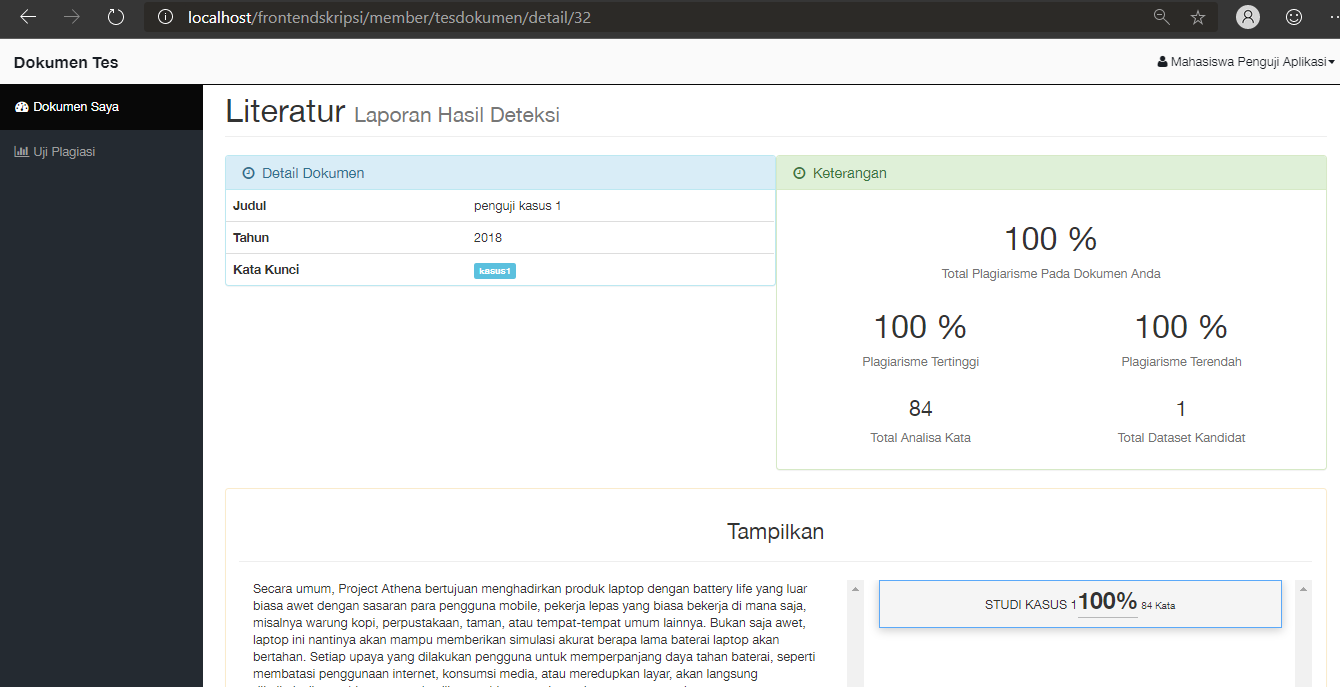
1. Hasil Pengujian Kasus Pertama

Hasil Pengujian pada kasus pertama merupakan pengujian yang dilakukan dengan memberikan dokumen yang sama antara kandidat pembanding dan data uji, berikut ini kandidat pembanding yang ditemukan berdasarkan kata kunci kasus1 sebanyak 1 kandidat pembanding, ditunjukkan pada gambar 6.2.



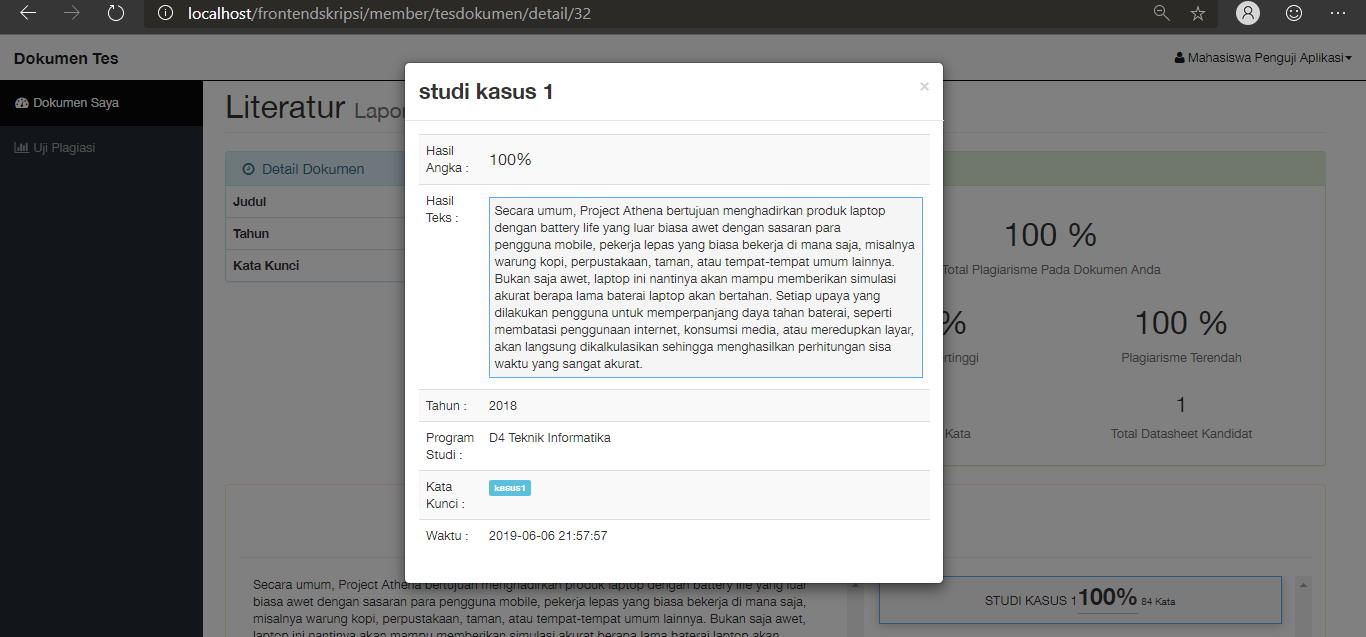
Gambar 6. 1 Hasil Pemilihan Kandidat Pembanding Kasus Pertama

Setelah kandidat pembanding telah ditemukan proses uji akan dilakukan sehingga akan menghasilkan tampilan yang ditunjukkan pada gambar 6.2. Pengujian pada kasus pertama ditemukan bahwa dokumen uji memiliki plagiarisme sebanyak 100% dengan 84 kata.



Gambar 6. 2 Hasil Pengujian Pada Kasus Pertama

Untuk mengetahui kalimat yang ditemukan dapat dilihat pada detail di setiap kandidat pembanding pada bagian panel tampilan dibagian kiri. Berikut ditunjukkan pada gambar 6.3.



Gambar 6. 3 Tampilan Detail Hasil Setiap Kandidat Pembanding

Dari hasil pengujian pada kasus pertama telah disediakan 113 dokumen yang tersimpan pada sistem dan 1 dokumen uji dimana perhitungan menggunakan *recall precision* sebagai berikut.

Tabel 6. 2 Recall Precision sesuai skenario pertama

|  |  |  |  |
| --- | --- | --- | --- |
|  | | Nilai Sebenarnya | |
| TRUE | FALSE |
| Nilai Prediksi | TRUE | 1 | 0 |
| FALSE | 0 | 112 |

*Recall =*

*Recall =*  x 100 = 100%

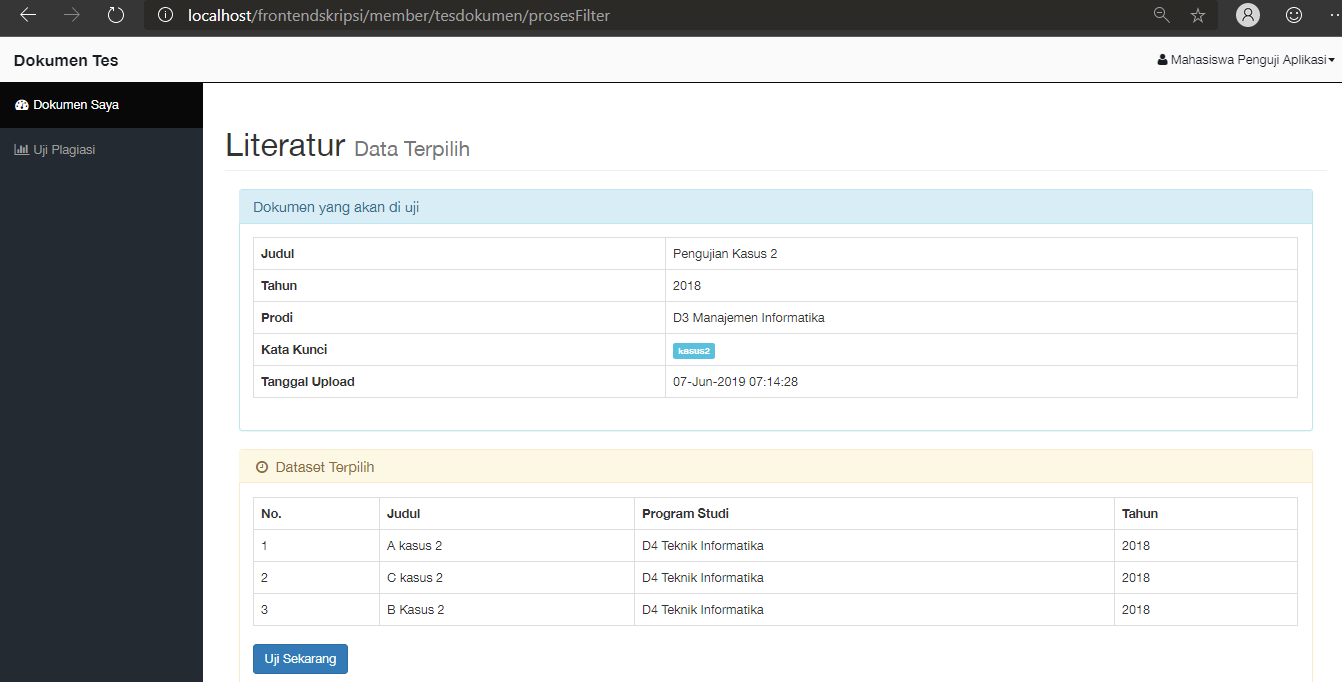
*Precision =*

*Precision =* x 100 = 100%

Dari hasil pengujian pada studi kasus pertama menggunakan *recall precision,* hasil perhitungan *recall* memiliki tingkat keberhasilan sistem dalam menemukan kembali sebuah informasi sebesar 100% ,sedangkan perhitungan *precision* didapatkan tingkat ketepatan antara informasi yang diminta oleh pengguna dengan jawaban yang diberikan oleh sistem sebesar 100%.

1. Hasil Pengujian Kasus Kedua

Hasil Pengujian pada kasus kedua merupakan pengujian yang dilakukan dengan memberikan dokumen data uji dengan memiliki beberapa kemiripan kalimat pada tiga kandidat pembanding, berikut ini kandidat pembanding yang ditemukan berdasarkan kata kunci kasus2 sebanyak 3, ditunjukkan pada gambar 6.4.

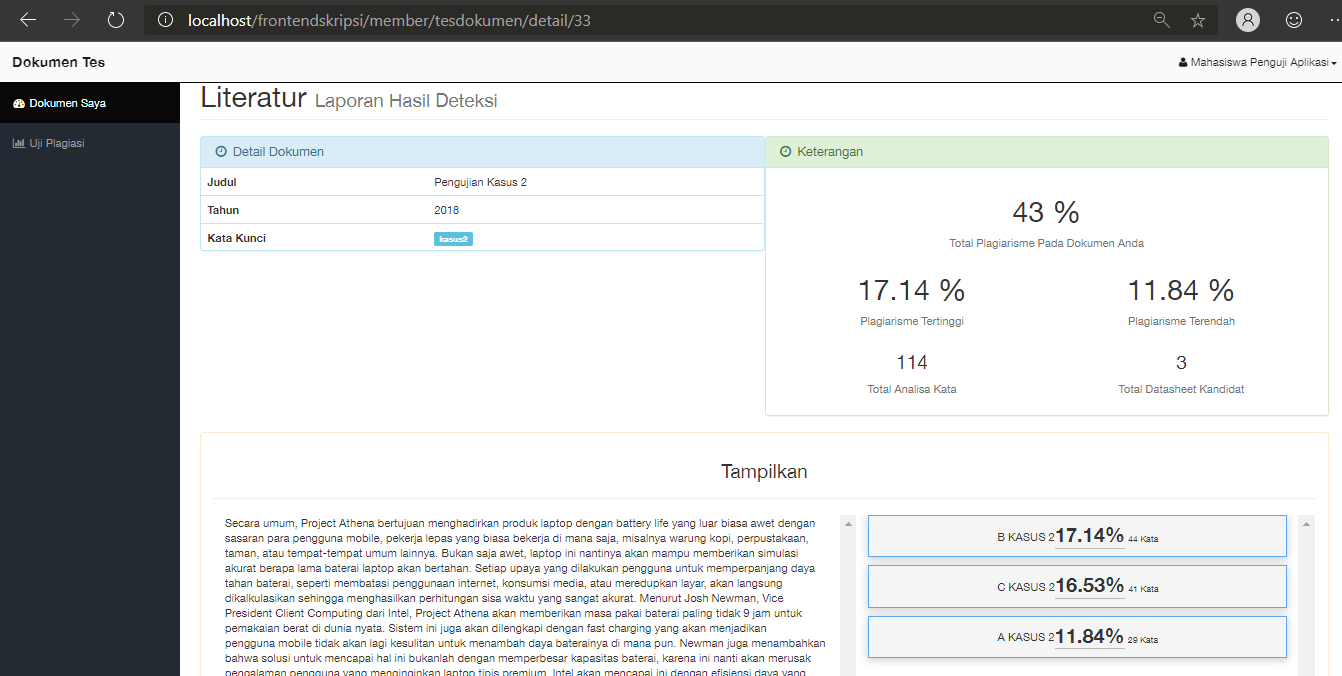


Gambar 6. 4 Hasil Pemilihan Kandidat Pembanding Kasus Kedua

Setelah kandidat pembanding telah ditemukan proses uji akan dilakukan sehingga akan menghasilkan tampilan yang ditunjukkan pada gambar 6.5. Pengujian pada kasus kedua ditemukan bahwa dokumen uji memiliki plagiarisme sebanyak 43.18% dengan 114 total kata dari 3 kandidat pembanding yang diujikan, dengan masing-masing kandidat pembanding terdapat plagiarisme yang ditunjukkan pada tabel 6.3.

Tabel 6. 3 Hasil Pengujian Pada Kasus Kedua

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Kandidat Pembanding** | **Jumlah Kata** | **Tingkat Plagiarisme (%)** |
| 1 | Kandidat pembanding 2 | 44 kata | 17.14 % |
| 2 | Kandidat pembanding 3 | 41 kata | 16.53 % |
| 3 | Kandidat pembanding 1 | 29 kata | 11.84 % |



Gambar 6. 5 Hasil Pengujian Pada Kasus Kedua

Dari hasil pengujian pada kasus kedua telah disediakan 113 dokumen yang tersimpan pada sistem dan 1 dokumen uji dimana perhitungan menggunakan *recall precision* sebagai berikut.

Tabel 6. 4 Recall Precision sesuai skenario kedua

|  |  |  |  |
| --- | --- | --- | --- |
|  | | Nilai Sebenarnya | |
| TRUE | FALSE |
| Nilai Prediksi | TRUE | 3 | 0 |
| FALSE | 0 | 110 |

*Recall =*

*Recall =*  x 100 = 100%

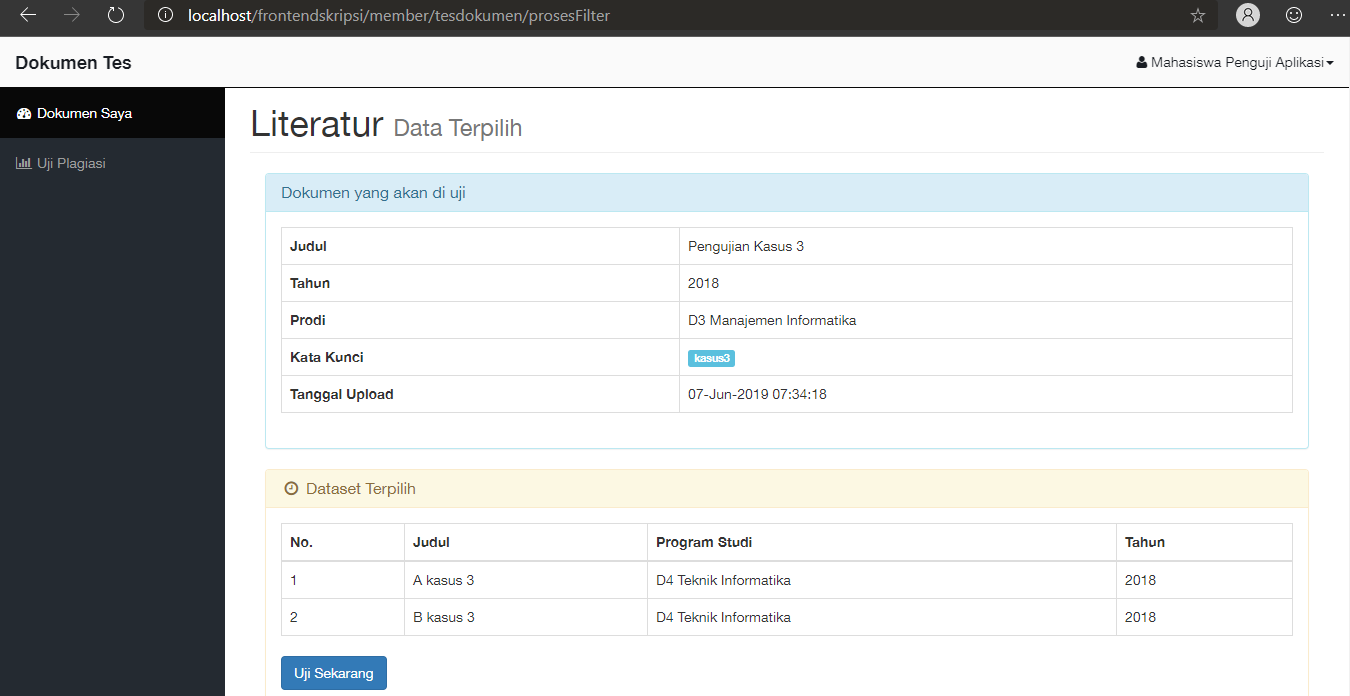
*Precision =*

*Precision =* x 100 = 100%

Dari hasil pengujian pada studi kasus kedua menggunakan *recall precision,* hasil perhitungan *recall* memiliki tingkat keberhasilan sistem dalam menemukan kembali sebuah informasi sebesar 100% ,sedangkan perhitungan *precision* didapatkan tingkat ketepatan antara informasi yang diminta oleh pengguna dengan jawaban yang diberikan oleh sistem sebesar 100%.

1. Hasil Pengujian Kasus Ketiga

Hasil Pengujian pada kasus ketiga merupakan pengujian yang dilakukan dengan memberikan dokumen data uji dengan memiliki beberapa kemiripan kalimat disetiap kandidat pembanding, berikut ini kandidat pembanding yang ditemukan berdasarkan kata kunci kasus3 sebanyak 2 kandidat pembanding, ditunjukkan pada gambar 6.6.

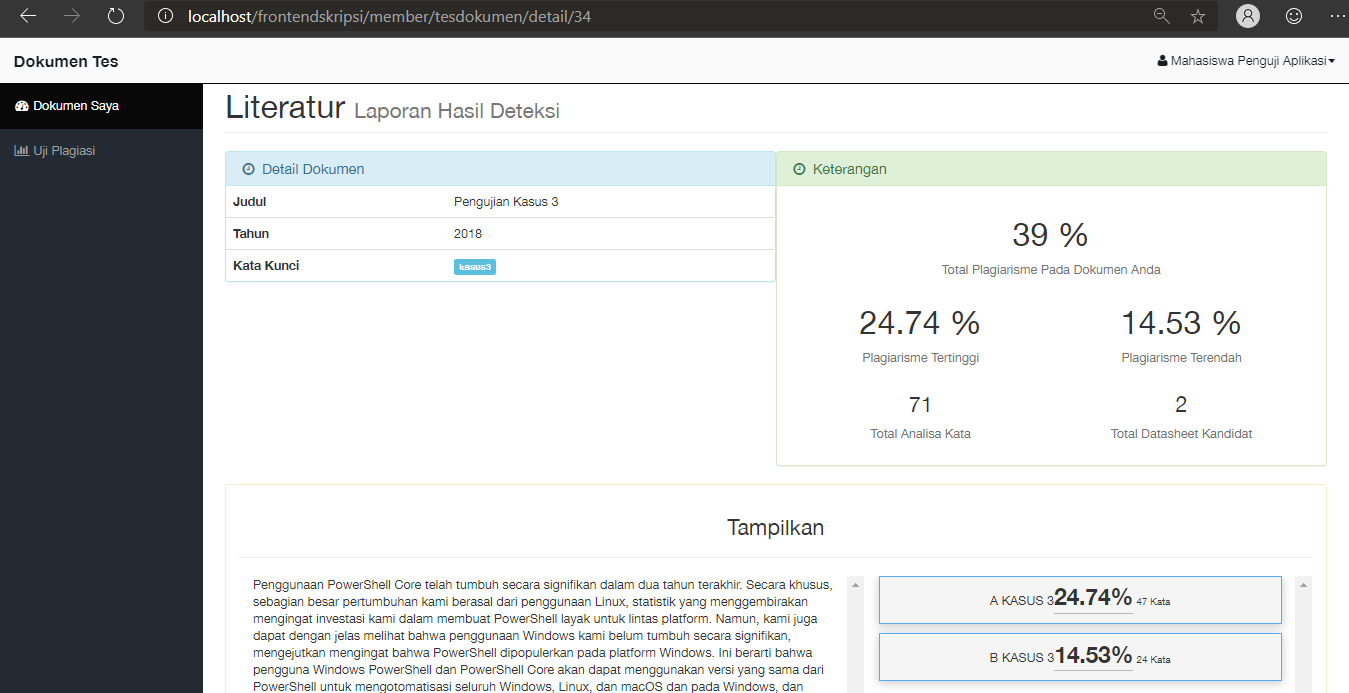


Gambar 6. 6 Hasil Pemilihan Kandidat Pembanding Kasus Ketiga

Setelah kandidat pembanding telah ditemukan proses uji akan dilakukan sehingga akan menghasilkan tampilan yang ditunjukkan pada gambar 6.7. Pengujian pada kasus ketiga ditemukan bahwa dokumen uji memiliki plagiarisme sebanyak 39 % dengan 71 total kata dari 2 kandidat pembanding yang diujikan, dengan masing-masing kandidat pembanding terdapat plagiarisme yang ditunjukkan pada tabel 6.5.

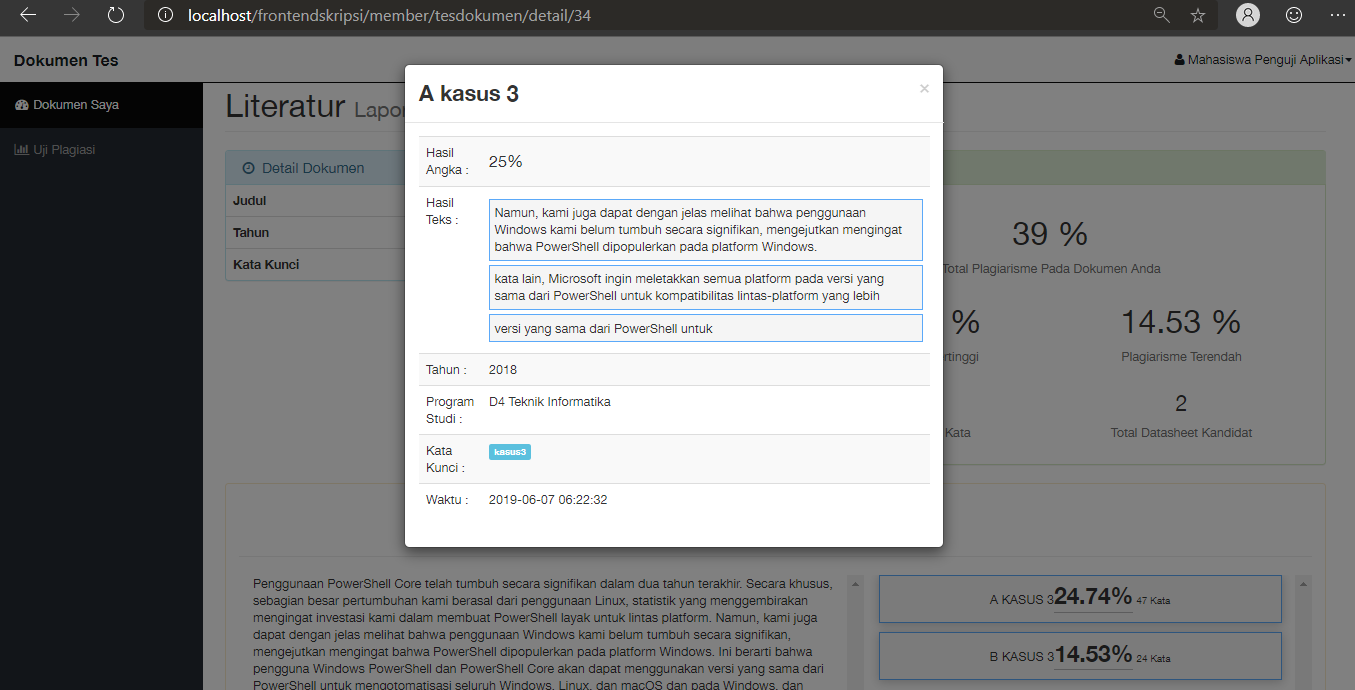
Tabel 6. 5 Hasil Pengujian Pada Kasus Ketiga

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Kandidat Pembanding** | **Jumlah Kata** | **Tingkat Plagiarisme (%)** |
| 1 | Kandidat pembanding 1 | 47 kata | 24.74 % |
| 3 | Kandidat pembanding 2 | 24 kata | 14.53 % |



Gambar 6. 7 Hasil Pengujian Pada Kasus Ketiga

Untuk mengetahui kalimat yang ditemukan dapat dilihat pada detail di setiap kandidat pembanding pada bagian panel tampilan dibagian kiri. Berikut ditunjukkan pada gambar 6.8.



Gambar 6. 8 Tampilan Detail Hasil Setiap Kandidat Pembanding

Dari hasil pengujian pada kasus ketiga telah disediakan 113 dokumen pada *dataset* dan 1 dokumen uji dimana perhitungan menggunakan *recall precision* sebagai berikut.

Tabel 6. 6 Recall Precision sesuai skenario ketiga

|  |  |  |  |
| --- | --- | --- | --- |
|  | | Nilai Sebenarnya | |
| TRUE | FALSE |
| Nilai Prediksi | TRUE | 2 | 0 |
| FALSE | 0 | 111 |

*Recall =*

*Recall =*  x 100 = 100%

*Precision =*

*Precision =* x 100 = 100%

Dari hasil pengujian pada studi kasus ketiga menggunakan *recall precision,* hasil perhitungan *recall* memiliki tingkat keberhasilan sistem dalam menemukan kembali sebuah informasi sebesar 100% ,sedangkan perhitungan *precision* didapatkan tingkat ketepatan antara informasi yang diminta oleh pengguna dengan jawaban yang diberikan oleh sistem sebesar 100%.

1. Pembahasan

Pembahasan memiliki tujuan untuk mendapatkan kesimpulan dari hasil pengujian aplikasi deteksi plagiarisme pada dokumen skripsi berdasarkan tingkat kesamaan dengan menggunakan *Longest Common Subsequence.* Pada proses pemberian studi kasus yang telah dilakukan. Pembahasan dilakukan terhadap hasil pengujian di setiap tahap pengujian. Pembahasan ini meliputi pembahasan pengujian validasi, pembahasan pengujian sistem dan pembahasan pengujian akurasi.

1. Pembahasan Pengujian Validitas

Pembahasan terhadap pengujian validasi yang dilakukan dengan melihat antara hasil kinerja sistem dengan daftar pengujian yang diberikan. Sehingga dapat disimpulkan bahwa implementasi dan fungsionalitas sistem deteksi plagiarisme memiliki kinerja yang bagus dan dapat berjalan sesuai skenario *use case* yang telah dijabarkan.

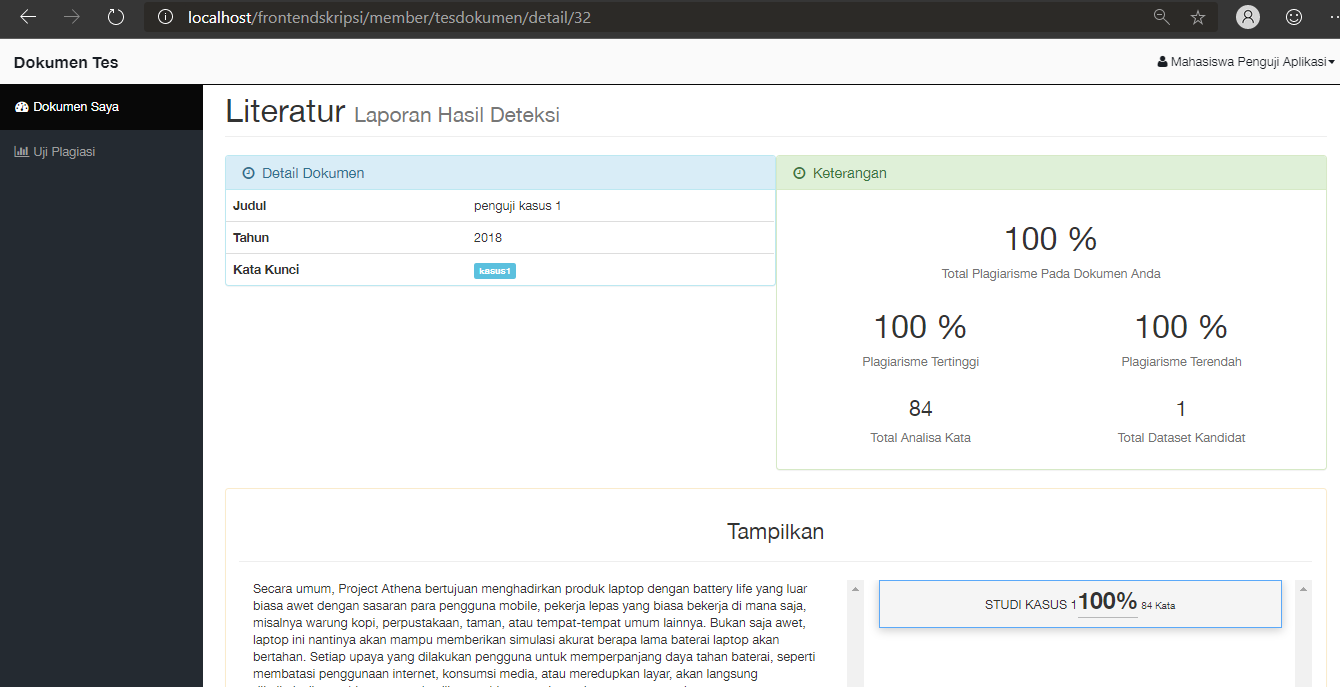
1. Pembahasan Pengujian Sistem

Dari percobaan pengujian sistem didapatkan kesimpulan bahwa tingkat keberhasilan pembuatan sistem deteksi plagiarisme telah berhasil dilakukan, setelah melakukan implementasi dan pengujian yang dilakukan pada aplikasi deteksi plagiarisme pada dokumen skripsi berdasarkan tingkat kesamaan dengan menggunakan *Longest Common Subsequence* menghasilkan hasil yang sama antara perancangan dan implementasi.

1. Pembahasan Pengujian Metode

Berdasarkan hasil pengujian pada metode yang telah diimplementasi kedalam sistem, pembahasan dari pengujian yang telah dilakukan terhadap tiga jenis setudi kasus antara lain :

1. Kasus Pertama



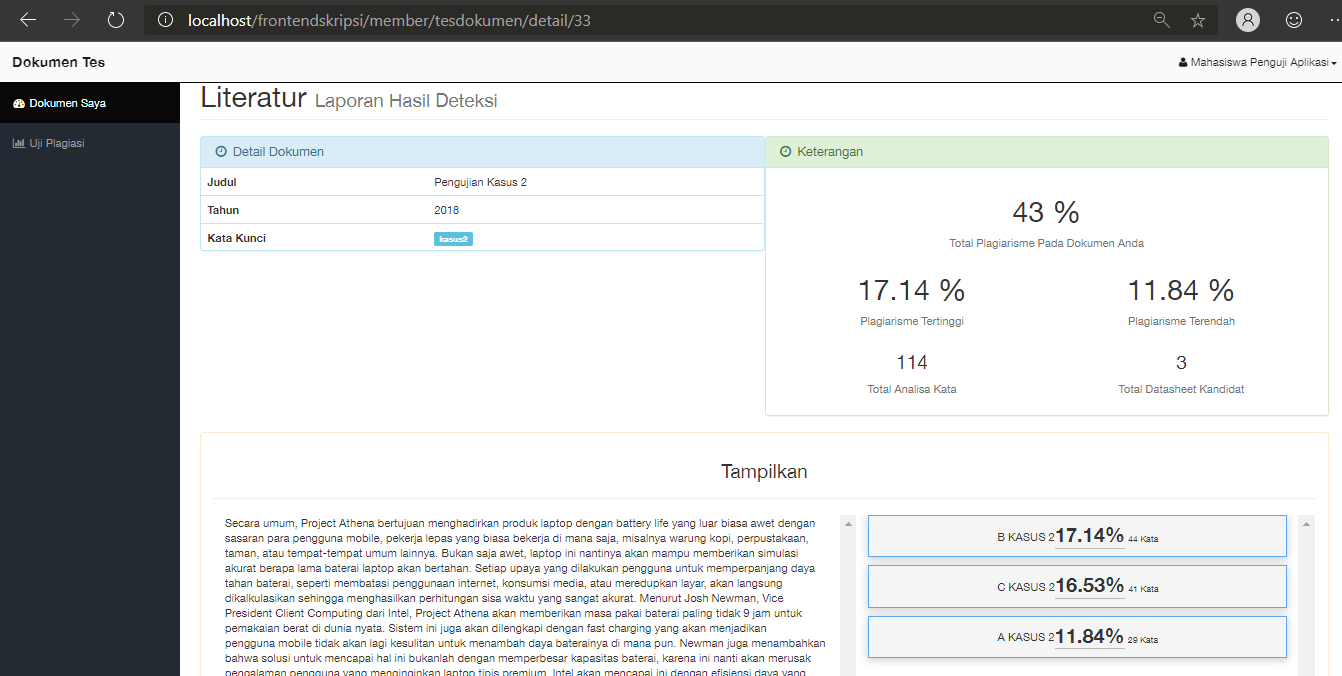
Gambar 6. 9 Perhitungan Kasus Pertama Pada Sistem

Tabel 6. 7 Perhitungan Kasus Pertama Manual

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No** | **Kategori Dokumen** | **Jumlah Kata** | **Jumlah ditemukan kata** | **Jumlah kalimat ditemukan** | **Tingkat Plagiarisme (%)** |
| 1 | Dokumen Uji | 84 Kata |  |  |  |
| 2 | Kandidat pembanding | 84 Kata | 84 kata | 1 | 100% |

Jumlah yang akan didapat dari tabel 6.7 adalah , maka hasil yang didapatkan 100%.

1. Kasus Kedua



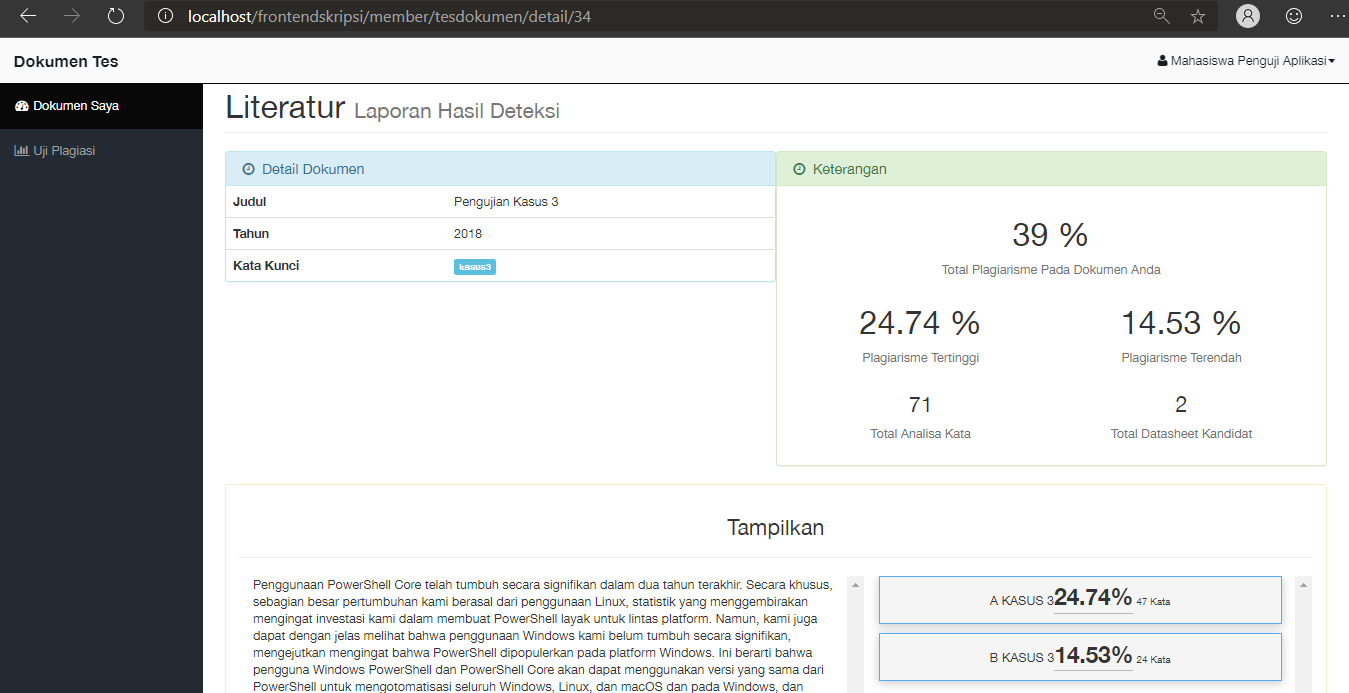
Gambar 6. 10 Perhitungan Kasus Kedua Pada Sistem

Tabel 6. 8 Perhitungan Kasus Kedua Manual

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No** | **Kategori Dokumen** | **Jumlah Kata** | **Jumlah kata ditemukan** | **Jumlah kalimat ditemukan** | **Tingkat Plagiarisme (%)** |
| 1 | Dokumen Uji | 262 Kata |  |  |  |
| 2 | Kandidat pembanding 1 | 134 Kata | 29 kata | 1 | 17.14 % |
| 3 | Kandidat pembanding 2 | 197 kata | 44 kata | 1 | 16.53 % |
| 4 | Kandidat pembanding 3 | 156 kata | 41 kata | 1 | 11.84 % |

Jumlah yang akan didapat dari tabel 9.8 adalah , maka total plagiarisme yang didapat pada dokumen uji yaitu 43 %.

1. Kasus Ketiga



Gambar 6. 11 Perhitungan Kasus Ketiga Pada Sistem

Tabel 6. 9 Perhitungan Kasus Ketiga Manual

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **No** | **Kategori Dokumen** | **Jumlah Kata** | **Jumlah kata disetiap LCS** | **Total kata ditemukan** | **Jumlah kalimat ditemukan** | **Tingkat Plagiarisme (%)** |
| 1 | Dokumen Uji | 183 Kata |  |  |  |  |
| 2 | Kandidat pembanding1 | 198 Kata | 23 kata, 18 kata, 6 kata | 47 kata | 3 | 24.74 % |
| 3 | Kandidat pembanding2 | 180 kata | 9 kata, 6 kata, 5 kata, 4 kata | 24 kata | 4 | 14.53 % |

Jumlah yang akan didapat dari tabel 6.9 adalah , maka total plagiarisme yang didapat pada dokumen uji yaitu 39 %.

Tabel 6.7, tabel 6.8, tabel 6.9 merupakan perhitungan yang dilakukan secara manual mengan menghitung jumlah kata di setiap dokumen uji dan kandidat pembandingnya.

BAB VII. KESIMPULAN DAN SARAN

1. Kesimpulan

Adapun kesimpulan yang dapat diambil dari hasil penelitian yang dilakukan mengenai deteksi plagiarisme pada dokumen skripsi berdasarkan tingkat kesamaan dengan menggunakan *Longest Common Subsequence* adalah sebagai berikut :

1. Metode Longest Common Subsequence dapat digunakan untuk deteksi plagiarisme dengan perbandingan dua atau lebih dokumen.
2. Sistem ini telah berhasil menerapkan metode *Longest Common Subsequence* untuk menguji lebih dari satu kalimat dan lebih dari satu kandidat pembanding*.*
3. Hasil pengujian yang telah dilakukan dengan tiga jenis setudi kasus sesuai dengan yang diharapkan, untuk pengujian *accuracy* ditunjukkan sebagai berikut :
   1. Kasus Pertama

, maka hasil yang didapatkan adalah 100%.

* 1. Kasus Kedua

, maka hasil yang didapat pada kandidat pembanding 2 adalah 17%.

, maka hasil yang didapat pada kandidat pembanding 3 adalah 16%.

, maka hasil yang didapat pada kandidat pembanding 1 adalah 11%.

, maka total plagiarisme yang didapat pada dokumen uji adalah 43 %.

* 1. Kasus Ketiga

, maka hasil yang didapat pada kandidat pembanding 1 adalah 23%.

, maka hasil yang didapat pada kandidat pembanding 2 adalah 13%.

, maka total plagiarisme yang didapat pada dokumen uji adalah 39 %.

sehingga dapat disimpulkan bahwa deteksi plagiarisme menggunakan metode LCS dapat menghasilkan tingkat akurasi yang tinggi.

1. Saran

Saran yang dapat diberikan dari hasil penelitian untuk pengembangan sistem ini ke depan sebagai berikut:

1. Sistem dapat ditambahkan library lain seperti memperluas *dataset* diluar jurusan Teknologi Informasi, crawling dari internet.
2. Sistem dapat dikombinasikan dengan algoritma lain untuk mendapatkan tingkat akurasi yang lebih maksimal.

DAFTAR PUSTAKA

[1] Alamsyah, Nur, “Deteksi Plagiarisme Tingkat Kemiripan Judul Skripsi Dengan Algoritma Winnowing,” Technologia,Vol.8, No.4, pp.205-213, Desember 2017.

[2] Widiastuti, Inta, Aplikasi Pendeteksi Kemiripan Dokumen Menggunakan Algoritma Rabin Karp, Malang : Politeknik Negeri Malang (2014).

[3] Pasma Cadea Mikha, Pengembangan Sistem Pendeteksi Kemiripan Karya Pada Inaicta 2013, Malang : Politeknik Negeri Malang (2014).

[4] Kharismadita Paratisa, Implementasi Tokenizing Plus Pada Sistem Pendeteksi Kemiripan Jurnal Skripsi Di Program Studi Teknik Informatika Politeknik Negeri Malang, Malang : Politeknik Negeri Malang (2015).

[5] Roshinta Trisna Ali, Analisis Skema-Skema Kemiripan Vektor Pada Sistem Penilaian Essay Online, Malang : Politeknik Negeri Malang (2016).

[6] Nurhadi Riza A, Pengembangan Data Uji Sistem Komputasi Kemiripan Teks Secara Semantik Berbahasa Indonesia, Malang : Politeknik Negeri Malang (2016).

[7] Dominikus Damas Putranto “Pengkajian Masalah *Longest Common Subsequences”* (2008). [Online]. http://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2007-2008/Makalah2008/MakalahIF2251-2008-047.pdf.

[8] Kensuke Baba, Tetsuya Nakatoh, Toshiro Minami “*Plagiarism detection using document similarity based on distributed representation*” 8th International Conference on Advances in Information Technology, IAIT2016, 19-22 December 2016, Macau, China, doi: https://doi.org/10.1016/j.procs.2017.06.038.

[9] Asad Abdi, Norisma Idris, Rasim M. Alguliyev, Ramiz M. Aliguliyev “*Plagiarism detection using linguistic knowledge*” Expert Systems With Applications (2015), doi: https://doi.org/10.1016/j.eswa.2015.07.048.

[10] Col Jyotindu Debnath “*Plagiarism: A silent epidemic in scientific writing – Reasons, recognition and remedies” Medical Journal Armed Forces India.* (2016), doi: https://doi.org/10.1016/j.mjafi.2016.03.010.

[11] L.K. Burdine, BA, M.B. de Castro Maymone, MD, DSc, N.A. Vashi, MD “*Text recycling: Self-plagiarism in scientific writing” International Journal of Women's Dermatology* (2018), doi: https://doi.org/10.1016/j.ijwd.2018.10.002.

[12] Alwi, dkk, Tata Bahasa Baku Bahasa Indonesia, cet. 8, Jakarta : Pusat Bahasa dan Balai Pustaka, 2010.

[13] Pratama, Rahmatullah Aditya, “Belajar UML - Use Case Diagram,” 21 02 2019. [Online]. *Available*: https://www.codepolitan.com/mengenal-uml-diagram-use-case.

[14] Aini, Bariroh Isriya Nur, Identifikasi Kemiripan Judul Tugas Akhir PSTI Dan PSMI Di Politeknik Negeri Malang, Malang : Politeknik Negeri Malang (2014).

[15] Basuki, Alwan Pribadi, “Menguasai CodeIgniter Kasus Membangun Aplikasi Perpustakaan”, Yogyakarta, Lokomedia, 2016.

[16] Galuh, Kresna, “Cara Menggunakan Bootstrap 3 untuk Membuat Web” 21 12 2015. [Online]. *Available* : https://www.codepolitan.com/tutorial/cara-menggunakan-bootstrap-3-untuk-membuat-web

[17] Sidik., Betha, Framework Codeigniter, Bandung : Infomatika (2012).

[18] Solikin, I, Perancangan Sistem Infomasi Penjualan Berbasis Framework Model View Controller (MVC) Pada PT Thamrin Brother Cabang Oki, Jurnal Media Informatika dan Komputer, (2014), 4 (1), pp. 1-13.

[19] Riyanto, Membuat Sendiri Aplikasi *E-commerce* dengan PHP dan MySQL Menggunakan *Codeigniter* dan *JQuery*, Yogyakarta : Andi, (2011).

# LAMPIRAN

Script setup.py

|  |
| --- |
| # python  import sys  import MySQLdb  import datetime  from lcs import LCS  class Database:  host = "localhost"  user = "root"  password = ""  db = "plagiarismedb"  def \_\_init\_\_(self):  self.connection = MySQLdb.connect( host = self.host,  user = self.user,  password = self.password,  db = self.db ,  autocommit=True)  def query(self, q):  cursor = self.connection.cursor()  cursor.execute(q)  self.connection.commit()  cursor.close()  #self.connection.close()  def queryall(self, q, va):  cursor = self.connection.cursor( MySQLdb.cursors.DictCursor )  cursor.execute(q, va)  return cursor.fetchall()  def queryv(self, q, va):  cursor = self.connection.cursor( MySQLdb.cursors.DictCursor )  cursor.execute(q, va)  return cursor.fetchone()  def \_\_del\_\_(self):  self.connection.close()  if \_\_name\_\_== "\_\_main\_\_":  db = Database()  id\_doc\_uji = sys.argv[1] #dokumen yg di uji  id\_antrian = sys.argv[2] #ambil id antrian  doc\_uji = (id\_doc\_uji,) #harus ada koma agar tidak eror  antrian = (id\_antrian,)  #doc\_uji = ("uKShycv3EiU4J1xrbRzC13052019tiYka0QA5dRB6MJpNb9j",) #harus ada koma agar tidak eror  #antrian = (24,)  q = "SELECT \* FROM dokumen\_user WHERE id\_dokumen\_user = %s"  data = db.queryv(q, doc\_uji)  q = "SELECT \* FROM antrian\_dataset as ad join dataset as ds on ad.id\_data = ds.id\_data WHERE id\_antrian\_dokumen = %s"  data1 = db.queryall(q, antrian)  #--------------------------------------------------  for td in data1:  list\_sentence = ""  angka\_sentence = 0  penelitian = data['penelitian']  while True:  common\_sentence, persent = LCS.longest\_common\_sentence(penelitian, td['penelitian'])  common\_words = common\_sentence.split(' ')  #berhenti ketika hasil plagiat kurang dari 3 kata  if(len(common\_words) <= 3):  break  else:  list\_sentence += '%s###' % common\_sentence #mengumpulkan hasil lcs ke 1 variable  angka\_sentence += float(persent)  buang\_plagiarisme = penelitian.split(common\_sentence) #membuang kalimat plagiat pada text uji  penelitian = ' '.join(buang\_plagiarisme) #menggabungkan kembali setelah meembuang kalimat plagiat  q = "UPDATE antrian\_dataset SET similarity\_angka='%s', similarity\_text='%s', status=%d, updated\_at='%s' WHERE id\_antrian\_dataset='%s' "% (angka\_sentence, list\_sentence, 1, datetime.datetime.now(), td['id\_antrian\_dataset'])  db.query(q) |

Script lcs.py

|  |
| --- |
| class LCS:  # --------------------------------------------------  def longest\_common\_substring(s1, s2):  m = [[0] \* (1 + len(s2)) for i in range(1 + len(s1))]  longest, x\_longest = 0, 0    for x in range(1, 1 + len(s1)): #ambil per kata dari doc 1, dan x sebagai doc terpanjang  for y in range(1, 1 + len(s2)): #ambil per kata dari doc 2  if s1[x - 1] == s2[y - 1]: #cek juka kata dari doc 1 == doc 2  m[x][y] = m[x - 1][y - 1] + 1  if m[x][y] > longest:  longest = m[x][y]  x\_longest = x  else:  m[x][y] = 0  return s1[x\_longest - longest: x\_longest] #memilih lcs terpanjang  def longest\_common\_sentence(s1, s2):    s1\_arr = []  s2\_arr = []  s1\_words = s1.split(' ') #memisah tiap kata menjadi array  s2\_words = s2.split(' ')    lg\_s1 = len(s1\_words) #menghitung panjang kata pada dokumen  lg\_s2 = len(s2\_words)    if lg\_s1 < lg\_s2: #menentukan dokumen terpanjang  s1\_arr = s2\_words  s2\_arr = s1\_words  else:  s1\_arr = s1\_words  s2\_arr = s2\_words    lcs\_arr = LCS.longest\_common\_substring(s1\_arr, s2\_arr)  s1\_pesent = (len(lcs\_arr)\*100)/len(s1\_arr)    lcs = ' '.join(lcs\_arr) #menggabungkan array hasil lcs  return lcs, s1\_pesent    # -------------------------------------------------- |