RANCANG BANGUN SISTEM *E-VOTING* DENGAN MENERAPKAN *HASH* DAN *DIGITAL SIGNATURE* UNTUK VERIFIKASI DATA HASIL *VOTING*

SKRIPSI

KONSENTRASI REKAYASA PERANGKAT LUNAK

Untuk memenuhi sebagian persyaratan memperoleh gelar Sarjana Komputer

Disusun Oleh :

DYAH AYU MARHAENINGTYAS GALUH WISNU

NIM. 0910680052

KEMENTERIAN PENDIDIKAN DAN KEBUDAYAAN

PROGRAM STUDI TEKNIK INFORMATIKA

PROGRAM TEKNOLOGI INFORMASI DAN ILMU KOMPUTER

UNIVERSITAS BRAWIJAYA

MALANG

2014

LEMBAR PERSETUJUAN

RANCANG BANGUN SISTEM *E-VOTING* DENGAN MENERAPKAN *HASH* DAN *DIGITAL SIGNATURE* UNTUK VERIFIKASI DATA HASIL *VOTING*

SKRIPSI

KONSENTRASI REKAYASA PERANGKAT LUNAK

Untuk memenuhi persyaratan memperoleh gelar Sarjana Komputer

Disusun oleh :

DYAH AYU MARHAENINGTYAS GALUH WISNU

NIM. 0910680052

Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I	Dosen Pembimbing II
Aswin Suharsono, S.T., M.T.	**Denny Sagita Rusdianto, S.Kom., M.Kom.**
NIK. 840919 06 1 1 0251	**NIK. 851124 06 1 1 0250**

LEMBAR PENGESAHAN

RANCANG BANGUN SISTEM *E-VOTING* DENGAN MENERAPKAN *HASH* DAN *DIGITAL SIGNATURE* UNTUK VERIFIKASI DATA HASIL *VOTING*

SKRIPSI

KONSENTRASI REKAYASA PERANGKAT LUNAK

Untuk memenuhi persyaratan memperoleh gelar Sarjana Komputer

Disusun oleh :

DYAH AYU MARHAENINGTYAS GALUH WISNU

NIM. 0910680052

Skripsi ini telah diuji dan dinyatakan lulus pada

Tanggal 7 Januari 2014

Penguji I	Penguji II
Dr. Eng Herman Tolle, ST., MT.	**Aryo Pinandito, ST., M.MT**
NIP. 197408232000121001	**NIK. 83051916110374**

Penguji III

Ismiarta Aknuranda, ST., M. Sc., Ph.D

NIK. 74071906110079

Mengetahui

Ketua Program Studi Teknik Informatika

Drs. Marji, M.T.

NIP. 19670801 199203 1 001

PERNYATAAN
ORISINALITAS SKRIPSI

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

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

Malang, Januari 2014
Mahasiswa,

Dyah Ayu Marhaeningtyas Galuh Wisnu
NIM 0910681004

KATA PENGANTAR

Puji syukur penulis panjatkan kehadirat Tuhan Yang Maha Esa karena hanya dengan rahmat dan karunia-Nya, penulis dapat menyelesaikan skripsi dengan judul "**Rancang Bangun Sistem *E-Voting* Dengan Menerapkan *Hash* Dan *Digital Signature* Untuk Verifikasi Data Hasil *Voting***".

Melalui kesempatan ini, penulis ingin menyampaikan rasa hormat dan terima kasih yang sebesar-besarnya kepada semua pihak yang telah memberikan bantuan dan dukungan selama penulisan skripsi, diantaranya:

1. Allah SWT atas segala nikmat dan rahmat-Nya.
2. Orang tua penulis, Bapak Priyo Sunanto Sidhy dan Ibu Yuliana, yang telah memberikan dukungan moral dan material.
3. Bapak Aswin Suharsono, S.T., M.T. selaku dosen pembimbing I yang telah memberikan ilmu dan saran untuk skripsi ini.
4. Bapak Denny Sagita Rusdianto, S.Kom, M.Kom. selaku dosen pembimbing II yang juga memberikan ilmu dan saran untuk skripsi ini.
5. Saudara penulis, Dewanty Ajeng Hastu Kartikaningtyas.
6. Teman-teman penulis, Aldim, Meitika, Wibi, Silvi, Tika, Winny, Ardy, Rangga, Sawung, Alan, Delis, Luthfi, seluruh teman-teman TPL-C, serta teman-teman angkatan 2009, angkatan 2010, dan angkatan 2011 yang telah selalu memberi dukungan, motivasi, kritik, dan saran.

Penulis sadar bahwa skripsi ini masih banyak kekurangan, oleh karena itu kritik dan saran yang bersifat membangun sangat diharapkan untuk menyempurnakan skripsi ini. Penulis berharap skripsi ini dapat bermanfaat khususnya bagi diri sendiri dan bagi semua pihak.

Malang, Januari 2014

Penulis

ABSTRAK

Dyah Ayu Marhaeningtyas Galuh Wisnu. 2014: Rancang Bangun Sistem *E-Voting* Dengan Menerapkan *Hash* Dan *Digital Signature* Untuk Verifikasi Data Hasil *Voting.* Skripsi Program Studi Teknik Informatika, Program Teknologi Informasi dan Ilmu Komputer, Universitas Brawijaya.
Dosen Pembimbing: Bapak Aswin Suharsono, S.T., M.T. dan Bapak Denny Sagita Rusdianto, S.Kom, M.Kom.

Pemungutan suara atau *voting* di Indonesia, khususnya pemilihan umum (pemilu) legislatif dan pimpinan eksekutif, masih menggunakan metode pemungutan suara manual menggunakan kertas suara. Namun metode ini memiliki banyak kekurangan. Banyak perselisihan dalam pemilu yang disebabkan oleh beberapa faktor diantaranya banyak kesalahan memberi tanda pada kertas suara sehingga tidak sah dan proses pengumpulan kartu suara penghitungan suara yang lambat. Untuk mengatasai permasalahan di atas salah satu solusinya dengan menyelenggarakan pemilu secara *online* atau *e-voting*. Namun *e-voting* masih memiliki kepercayaan rendah dalam masyarakat. Untuk itu perlu dibuat sebuah sistem yang dapat menjamin akurasi hasil *e-voting*. Pada kasus *E-Voting*, integritas data dibutuhkan ketika melakukan pengiriman hasil *voting* dari tempat pemungutan suara ke pusat pemungutan suara. Selain itu dalam penerimaan hasil *voting* dilakukan validasi agar dapat diketahui pengirim aslinya. Perancangan sistem ini meliputi dua tahap yaitu perancangan aplikasi dan perancangan proses *hashing* dan *digital signing*. Implementasi sistem ini menggunakan bahasa pemrograman PHP dengan *framework code igniter*. Untuk proses hashing menggunakan *Secure Hash Algorithm-1*, sedangkan untuk *digital signature* menggunakan pasangan kunci RSA. Pengujian menggunakan metode *blackbox-testing* dengan strategi pengujian validasi dan verifikasi. Dari hasil pengujian dapat disimpulkan bahwa sistem dapat bekerja dengan baik dan proses *hashing dan digital signing* pada sistem *e-voting* ini telah berjalan dengan baik sehingga dapat diverifikasi apakah hasil voting mengalami perubahan selama proses pengiriman.

Kata Kunci: *e-voting*; *hashing*; *algoritma RSA*; *digital signature*;

ABSTRACT

Dyah Ayu Marhaeningtyas Galuh Wisnu. 2014: *Design of E-voting System Applying Hash and Digital Signature for Voting Result Data Verification*.
***Supervisors*: Mr. Aswin Suharsono, S.T., M.T. and Mr. Denny Sagita Rusdianto, S.Kom, M.Kom.**

Voting in Indonesia, especially legislative and chief executive election is still using the manual method of voting ballot. However, this method has many shortcomings. Many disputed in the election caused bu several factors, including a lot of errors in marking the paper so the voting is not valid and the gathering and counting voting ballot process is slow. To handle all the problems above is holding elections with electronic voting or e-voting. But e-voting still have a low trust. It needs to made a system that can ensure the accuracy of the voting results. In the case of e-voting, data integrity required when sending the voting results from the polling place to polling centers. In receipt of the voting results needs validation in order to know the original sender. The system design includes two stages: system design and the design process of hashing and digital signing. The implementation uses the programming language PHP with Code Igniter framework. Hashing process uses Secure Hash Algorithm-1, whereas for digital signatures using RSA key pairs. It tests using blackbox-testing with strategy validation and verification. From the test results it can be concluded that the system can work well and the process of hashing and digital signing of e - voting system has been running well so it can be verified whether the results of voting changes during the shipping process.

KeyWords: e-voting; *hashing*; *RSA algorihm; digital signature*;

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB 1
PENDAHULUAN

1.1 Latar Belakang

Kebanyakan pemungutan suara atau *voting* di Indonesia, khususnya pada pemilihan umum (pemilu) legislatif maupun pimpinan eksekutif, masih menggunakan metode pemungutan suara manual. Metode ini adalah dengan melakukan pemungutan suara menggunakan kertas suara yang diberi tanda untuk memilih calon. Pemilih akan datang ke tempat pemungutan suara kemudian mendaftarkan secara manual berdasarkan KTP dan melakukan pemungutan suara dengan cara melakukan pencoblosan terhadap kertas suara yang diberikan. Namun metode ini memiliki banyak kekurangan.

Banyaknya perselisihan dalam pemilu di antaranya disebabkan oleh beberapa faktor yang meliputi; (1) Ketika pemungutan suara banyak pemilih yang melakukan kesalahan dalam memberi tanda pada kertas suara akhirnya banyak kartu suara yang dinyatakan tidak sah. (2) Proses pengumpulan kartu suara yang berjalan lambat, karena perbedaan kecepatan pelaksanaan pemungutan suara di masing-masing daerah. Hal ini ditambah dengan kondisi geografis negara kita yang heterogen sehingga dapat menghambat distribusi kartu suara. (3) Proses penghitungan suara yang dilakukan di setiap daerah juga berjalan lambat karena proses tersebut harus menunggu semua kartu suara terkumpul terlebih dahulu sehingga memperlambat penghitungan suara. Untuk mengatasai permasalahan di atas salah satu solusi yang dapat diterapkan adalah dengan menyelenggarakan pemilu secara *online* atau yang lebih dikenal dengan istilah *electronic voting* atau *e-voting*.[ROK-11]

E-Voting adalah suatu metode pemungutan suara dan penghitungan suara dalam suatu pemilihan dengan menggunakan perangkat elektronik. Namun e-voting masih menghadapi tingkat kepercayaan yang rendah. Pemilih yang berpartisipasi dalam program percontohan di Jembrana, Bali

kemudian mengatakan kepada kelompok opini bahwa mereka yakin hasil dari sistem ini akan jauh lebih mudah dimanipulasi oleh seseorang. Kurangnya kemampuan berteknologi juga merupakan masalah.[KAT-13] Oleh karena itu, perlu dibuat sebuah sistem yang dapat menjamin akurasi hasil *e-voting* dengan mengamankan hasil *e-voting.*

Garfinkel mengemukakan bahwa keamanan komputer melingkupi empat aspek, yaitu : *privacy(confidentiality), integrity, authentication,* dan *availability.* Selain itu masih ada dua aspek lain yang juga sering dibahas dalam kaitannya dengan *electronic commerce* yaitu *access control* dan *non repudiation*(Budi Raharjo, 2002). Inti utama dari aspek *confidentiality* adalah usaha untuk menjaga informasi dari orang yang tidak berhak mengakses. Aspek *integrity* menekankan bahwa informasi tidak boleh diubah tanpa seijin pemilik informasi. Aspek *authentication* berhubungan dengan metode untuk menyatakan bahwa informasi betul-betul asli, orang yang mengakses atau memberikan informasi adalah betul-betul orang yang dimaksud, atau *server* yang kita tuju adalah benar-benar *server* asli. Aspek *availability* atau ketersediaan berhubungan dengan ketersediaan informasi ketika dibutuhkan. Sistem yang diserang dapat menghambat akses ke informasi. Aspek *access control* berhubungan dengan cara pengaturan akses kepada informasi. Aspek *non repudiation* menjaga agar seseorang tidak dapat menyangkal telah melakukan transaksi.[UPN-13]

Untuk membangun suatu sistem *e-voting* yang aman diperlukan untuk memenuhi semua aspek diatas sehingga dapat memenuhi asas pemilu. Penelitian tentang kemanan suatu sistem *e-voting* merupakan penelitian yang besar sehingga dalam pengerjaannya dibagi dalam beberapa aspek untuk dikerjakan oleh beberapa orang. Untuk aspek *confidentiality* dikerjakan oleh Saudara Tika Rahmadian dan untuk aspek *authentication* dikerjakan oleh Saudara Nurlia. Dan pada skripsi ini yang dikerjakan adalah aspek *integrity*. Sehingga pada pengerjaannya hanya memfokuskan bagaimana mengimplementasikan suatu sistem yang dapat menjaga integritas data.

Integritas data merupakan hal yang sangat penting dalam pengiriman data dari suatu tempat ke tempat lain. Pada kasus *E-Voting*, integritas data dibutuhkan ketika melakukan pengiriman hasil *voting* dari tempat pemungutan suara ke pusat pemungutan suara. Selain itu dalam penerimaan hasil *voting* perlu dilakukan validasi agar dapat diketahui pengirim asli dari hasil *voting* yang telah diterima. Di Indonesia, masih banyak daerah-daerah terpencil yang tidak memiliki akses *internet* sehingga transaksi apapun masih belum dapat dilakukan secara *online*. Hal ini menyebabkan *file* hasil *voting* harus dikirimkan secara *offline*, misalnya melalui *removable disk*. Namun dengan pengiriman secara *offline* akan mempermudah penyerangan dengan cara mengubah isi file maupun mengganti *file* asli dengan *file* yang lain.

Untuk mengatasinya hal tersebut dapat diselesaikan dengan adanya proses *hashing* dan *digital signature*. Proses *hashing* akan membantu untuk mendeteksi ada atau tidaknya perubahan yang dilakukan pada *file* yang terkirim. Sedangkan *digital signature* akan digunakan untuk mengetahui apakah *file* yang terkirim berasal dari pengirim yang bersangkutan atau tidak.

1.2 Rumusan Masalah

Berdasarkan uraian latar belakang di atas, maka dapat dirumuskan permasalahan pada skripsi ini yaitu sebagai berikut:

1. Bagaimana perancangan dan implementasi dari *Hashing* dan *Digital Signature* pada Sistem *E-Voting*?
2. Bagaimana membangun skema sistem *e-voting* dengan *hashing* dan *digital signature*?
3. Apakah sistem ini dapat mendeteksi perubahan pada data hasil *voting* yang terkirim?
4. Apakah sistem ini dapat memverifikasi pengirim dari data hasil *voting* yang terkirim?

1.3 Batasan Masalah

Agar permasalahan yang dirumuskan dapat lebih terfokus, maka penelitian tugas akhir ini dibatasi dalam hal:

1. Pengujian yang dilakukan meliputi validasi dan verifikasi.
2. Keamanan hanya fokus pada aspek *integrity*.
3. Proses *backup*, verifikasi *database*, dan verifikasi *digital signature* hanya dapat dilakukan oleh administrator.
4. User dapat berinteraksi hanya dalam proses *voting*.

1.4 Tujuan

Berdasarkan uraian rumusan masalah di atas, maka dapat diketahui tujuan pada skripsi ini yaitu sebagai berikut:

1. Merancang metode *hashing* menggunakan algoritma SHA-1 dan Algoritma RSA untuk *digital signature*.
2. Mengimplementasikan *algoritma hash* dan *digital signature* pada suatu sistem *E-Voting* berbasis Web.
3. Melakukan pengujian apakah implementasi *hashing* dan *digital signature* dapat mendeteksi perubahan yang ada pada data yang terkirim.
4. Melakukan pegujian apakah implementasi *hashing* dan *digital signature* dapat memvalidasi pengirim data hasil voting.

1.5 Manfaat

Penelitian ini diharapkan dapat bermanfaat untuk berbagai pihak. Manfaat dari penelitian ini adalah sebagai berikut:

- Penulis
    1. Mendapatkan pemahaman tentang implementasi asas pemilu ke dalam aplikasi elektronik *E-Voting*.
    2. Mendapatkan pemahaman tentang implementasi algoritma RSA ke dalam suatu program.

3. Mendapatkan pemahaman tentang implementasi proses *hashing* dan cara kerjanya dalam proses penjagaan integritas untuk mendeteksi perubahan data.
4. Mendapatkan pemahaman tentang implementasi *digital signature* dan cara kerjanya untuk mendeteksi keaslian pengirim data.

- Pengguna
  1. Mendapatkan wawasan akan pengimplementasian sistem pemilu ke dalam suatu aplikasi elektronik.
  2. Dapat diterapkan pada pemilihan-pemilihan pada organisasi mahasiswa di Program Teknologi Informasi dan Ilmu Komputer Universitas Brawijaya.
  3. Memudahkan pengguna untuk melakukan *voting* tanpa khawatir data tidak aman.

1.6 Sistematika Penulisan

Sistematika penulisan laporan tugas akhir ini adalah sebagai berikut:

- **Bab I Pendahuluan**

Berisi tentang latar belakang, rumusan masalah, batasan masalah, tujuan penelitian,manfaat penelitian, sistematika penulisan, dan jadwal penelitian.

- **Bab II Tinjauan Pustaka**

Berisi tentang dasar teori secara luas mengenai *software* maupun *hardware* yang diperlukan untuk pengembangan dan perancangan sistem *e-voting* yang menerapkan *hashing* dan *digital signature* untuk verifikasi data hasil *voting*.

- **Bab III Metode Penelitian dan Perancangan**

Berisi tentang langkah-langkah, penjelasan penelitian dan perancangan sistem dalam pengembangan dan perancangan sistem *e-voting* yang menerapkan *hashing* dan *digital signature* untuk verifikasi data hasil *voting*.

- **Bab IV Implementasi Sistem**

Bab ini berisi tentang implementasi aplikasi yang dibangun, meliputi pembuatan aplikasi berbasis web dengan menggunakan pemrograman PHP dengan *framework Code Igniter*.

- **Bab V Pengujian dan Analisis**

Berisi mengenai analisa hasil perancangan aplikasi, hasil analisa output aplikasi, dan pembahasan terjadinya kegagalan (apabila terjadi kegagalan)

- **Bab VI Kesimpulan dan Saran**

Pada bab ini berisi kesimpulan yang diambil berdasarkan analisa sistem secara keseluruhan, kelebihan dan kekurangannya sistem, serta saran-saran guna menyempurnakan sistem yang dibuat.

1.7 Jadwal Penelitian

Tabel 1. 1 *Timeline* penelitian

No	Kegiatan	Bulan dan Minggu ke-																					
		Bulan 1				Bulan 2				Bulan 3				Bulan 4				Bulan 5				Bulan 6			
		1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
1	Studi literatur																								
2	Analisa kebutuhan																								
3	Perancangan perangkat lunak																								
4	Implementasi perangkat lunak																								
5	Pengujian perangkat lunak																								
6	Penulisan laporan penelitian																								

BAB II
KAJIAN PUSTAKA DAN DASAR TEORI

2.1 Sistem E-Voting

E-Voting adalah suatu sistem pemilihan dimana data dicatat, disimpan, dan diproses dalam bentuk informasi digital . Jadi e-voting pada hakekatnya adalah pelaksanaan pemungutan suara yang dilakukan secara elektronik (digital) mulai dari proses pendaftaran pemilih, pelaksanaan pemilihan, penghitungan suara, dan pengiriman hasil suara.[ROK-11]

Di Indonesia sistem *e-voting* telah diterapkan di Jembrana, Bali pada tahun 2009 untuk pemilihan kepala-kepala desa. Jumlah total pemilih dalam pilkades tersebut tercatat sebanyak 2.507 yang tersebar di empat dusun atau banjar, yakni Banjar Tengah, Banjar Kebebeng, Banjar Dlod Pempatan, Banjar Baler Bale Agung. Empat TPS dengan metode e-voting pun dibangun di tempat terpisah, dan kantor kepala desa akan dijadikan sebagai posko e-voting atau pusat penayangan tabulasi hasil yang dikirimkan dari tiap-tiap TPS.

Pelaksanaan pilkades ini selain mendapat pengawalan ketat dari berbagai unsur masyarakat, juga mendapat pendampingan teknis dari tim BPPT. Adapun pemilihan kepala desa (Pilkades) dengan pemilihan elektronik (e-voting) dilengkapi sistem verifikasi pemilih menggunakan e-KTP.

Kepala Badan Pengkajian dan Penerapan Teknologi (BPPT), Marzan A. Iskandar yang menyaksikan langsung jalannya kegiatan ini menyatakan bahwa peristiwa ini merupakan inovasi dan terobosan baru. Beliau mengatakan bahwa berbeda dengan Pilkades sebelumnya, kali ini e-KTP digunakan sebagai tanda identitas pemilih yang kemudian dibaca dengan card reader, sehingga otentik ketika di cek dengan Daftar Pemilih tetap (DPT) online. Setelah selesai dilakukan pemungutan di seluruh TPS juga

akan direkap secara online. Semua ini berjalan secara rahasia namun lebih terkesan luber dan jurdil. Hal ini jelas merupakan terobosan bagi Indonesia.

Kepala Program Pemilu elektronik BPPT, Andrari Grahitandaru menjelaskan bahwa Pilkades dengan e-voting jelas menciptakan penghematan yang signifikan. Indonesia sendiri memiliki 76665 desa, dengan biaya operasional Pilkades per desanya sebesar Rp 25 juta. Maka total biaya pilkades nasional mencapai hampir 2 triliun. Hal ini dapat dicontohkan pada Pilkades sebelumnya di Boyolali yang dilakukan di 160 desa. Beliau mengatakan bahwa jika ditotal biayanya mencapai Rp 4 miliar. Melalui e-voting menghemat Rp 2 miliar. Dijelaskan jika melaksanakan e-voting pemerintah daerah hanya butuh menginvestasikan lima perangkat e-voting seharga Rp 50 juta dan bisa dipakai berulang-ulang. Karena setiap Kabupaten praktis hanya tinggal membeli minimal 5 perangkat e voting tersebut dan dapat digunakan di tiap penyelenggaraan Pilkades.[SET-13]

Ketua Komisi Pemilihan Umum (KPU) mengatakan bahwa Mahkamah Konstitusi juga menilai pasal 88 UU 32/2004 adalah konstitusional sepanjang penggunaan metode e-voting itu tidak melanggar asas luber jurdil.[BPP-10]

2.2 Enkripsi Asimetris

Enkripsi adalah suatu proses untuk mengubah sebuah pesan, data atau informasi (biasa disebut *plaintext*), sehingga informasi tersebut tidak dapat dibaca oleh orang yang tidak bertanggung-jawab(*ciphertext*). Jadi *plaintext* adalah informasi yang dapat dimengerti dan *ciphertext* adalah informasi yang tidak dapat dimengerti atau dibaca. Enkripsi adalah bagian dari sebuah ilmu yang disebut kriptologi atau kriptografi. Kriptografi adalah sebuah ilmu yang mempelajari teknik untuk membuat sebuah pesan atau informasi tidak dapat dibaca oleh orang yang tidak berhak. Ada dua teknik yang cukup terkenal dalam kriptografi yaitu: *symmetric key encryption* dan *public key encryption* (enkripsi asimetris).

Pada enkripsi asimetris dibutuhkan dua buah kunci, yaitu: *public key* dan *private key* atau kunci umum dan kunci pribadi. Kunci umum memang kunci yang dibuat untuk disebarkan kepada publik. Kunci umum digunakan oleh siapa saja yang ingin mengirim data atau pesan kepada orang yang mempunyai kunci umum tersebut. Sedangkan kunci pribadi harus dijaga kerahasiaannya dan digunakan untuk mendekrip data atau pesan yang diterima.[SOE-13]

2.2.1 Algoritma RSA

Salah satu algoritma enkripsi dengan kunci asimetris yang terkenal adalah algoritma RSA. Algoritma ini menghasilkan sepasang kunci yang mana salah satu kunci dapat dijadikan kunci umum dan kunci lainnya menjadi kunci pribadi. Faktor yang menyebabkan tingginya tingkat keamanan adalah sulitnya memfaktorkan bilangan yang besar menjadi faktor-faktor prima dimana pemfaktoran ini dilakukan untuk menentukan kunci privat. [SOE-13]

2.2.2 *Generate Key* Algoritma RSA

Besaran- besaran yang digunakan pada algoritma RSA[SOE-13]:

1. **p** dan **q** bilangan prima (rahasia)
2. **r = p . q** (tidak rahasia)
3. **m = (p – 1)(q – 1)**
4. **PK** (kunci enkripsi) (tidak rahasia)
5. **SK** (kunci dekripsi) (rahasia)
6. **X** (*plaintext*) (rahasia)
7. **Y** (*ciphertext*) (tidak rahasia)

Cara pembuatan pasangan kunci RSA[SOE-13]:

1. Tentukan 2 bilangan integer prima besat misal kita sebut dengan **p** dan **q**. Pilih p dan q dengan ukuran besar agar tingkat keamanan semakin besar, misal 1024 bit.

2. Tentukan **n**, dimana **n = p*q**
3. Tentukan **m**, dimana **m = (p-1)*(q-1)**
4. Pilih **e** yang *relatively prime* terhadap **m**. Dimana e *relatively prime* terhadap m, artinya faktor pembagi terbesar keduanya adalah 1, secara matematis disebut **gcd(e,m) = 1**. (dapat digunakan algoritma Euclid)
5. Cari **d**, sehingga **e*d = 1 mod (m)**, atau **d = (1+nm)/e**. Untuk bilangan besar dapat digunakan algoritma extended Euclid.
6. Kunci publik : **e**, **n**, Kunci privat : **d**, **n**

2.3 *Hash*

Fungsi *hash* digunakan untuk membuktikan bahwa data yang dikirimkan tidak mengalami perubahan. Suatu fungsi hash akan mengambil suatu input data dan kemudian mengubahnya untuk menghasilkan suatu *hash value. Hash value* biasa disebut *message digest* atau sidik jari suatu pesan karena sangat kecil kemungkinan bahwa dua dokumen memiliki nilai hash yang sama. Fungsi *hash* sangat sulit untuk dikembalikan ke nilai semula sehingga aman untuk menyimpan *password* atau penanda integritas.[MGL-07]

2.3.1 SHA-1

SHA-1 adalah salah satu dari rangkaian algoritma yang diciptakan *United States National Security Agency.* SHA adalah singkatan dai *Secure Hash Algorithm.* Dari semua jenis SHA yang ada, SHA-1 adalah yang paling umum digunakan. SHA-1 sudah digunakan dalam berbagai macam aplikasi dan protokol.

SHA-1 menghasilkan *digest* sebesar 160-bit. Asal-usul SHA-1 adalah dari prinsip-prinsip yang mirip dengan yang digunakan oleh Ronald L. Rivest untuk algoritma MD4 dan MD5.[AZH-13]

2.4 *Digital Signature*

Digital signature atau tanda tangan digital adalah suatu tanda tangan elektronik yang dapat digunakan untuk melakukan otentikasi identitas pengirim dari sebuah pesan atau penanda tangan suatu dokumen, dan untuk memastikan bahwa isi asli dari pesan atau dokumen yang telah dikirim tersebut tidak mengalami perubahan. Tanda tangan digital mudah dalam transportasinya, tidak dapat ditiru oleh orang lain, dan dapat secara otomatis dilakukan *time-stamp*. Kemampuan untuk memastikan bahwa pesan yang ditandatangani dan diterima adalah asli berarti bahwa pengirim tidak dapat dengan mudah menyangkal nantinya.[MGL-07]

Sebuah digital signature dapat digunakan pada bermacam-macam pesan, tidak peduli apakah terenkripsi atau tidak, sehingga penerima yakin pada identitas pengirim dan pesan yang diterima masih utuh.[MGL-07]

Cara kerja dari digital signature adalah dengan menggunakan kunci privat dan kunci publik penerima pesan. Pengirim pesan akan melakukan *digital signing* dengan menggunakan kunci publik penerima pesan.

Sebuah *digital signature* direpresentasikan dalam komputer sebagai string dari digit biner. Sebuah *digital signature* dihitung dengan menggunakan satu set parameter dan mengotentikasi integritas data ditandatangani dan identitas penandatangan. Sebuah algoritma yang memiliki kemampuan untuk menghasilkan dan memverifikasi tanda tangan. Algoritma ini memanfaatkan sebuah kunci privat untuk menghasilkan suatu *digital signature*. Verifikasi tanda tangan memanfaatkan kunci publik, yang sesuai tetapi tidak sama dengan kunci privat. Setiap user memiliki sepasang kunci privat dan publik. Kunci publik diasumsikan dikenal masyarakat pada umumnya. Kunci privat tidak pernah dibagikan. Siapapun dapat memverifikasi tanda tangan dari pengguna dengan menggunakan kunci publik pengguna. Hanya pemilik dari kunci privat yang dapat melakukan *generate digital signature*.[MGL-07]

2.5 *Openssl*

OpenSSL adalah suatu protokol tambahan yang digunakan untuk Secure Socket Layer. Yang maksudnya adalah mengamankan jaringan kita antara client dan server. Dengan OpenSSL ini, maka jaringan akan sulit di sniffing. Jika dalam keadan HTTP biasa (Plain TEXT), kemungkinan besar bisa terkenad MITM Attack (Man in the Middle Attack). [MAN-10]

Aplikasi OpenSSL ini merupakan command line tool yang menggunakan berbagai fungsi kriptografi OpenSSL's crypto library dari shell. ini dapat digunakan untuk [SAS-09] :

1. Penciptaan RSA, DH dan DSA parameter kunci
2. Penciptaan sertifikat X.509, CSRs dan CRLs
3. Perhitungan Pesan Digests
4. Enkripsi dan Dekripsi dengan Ciphers
5. Pengujian SSL / TLS Client dan Server
6. Penanganan S/MIME signed or encrypted mail

2.6 PHP

Script PHP adalah bahasa program yang berjalan pada sebuah webserver, atau sering disebut server-side. Oleh karena itu,PHP dapat melakukan apa saja yang bisa dilakukan program CGI lain, yaitu mengolah data dengan tipe apapun, menciptakan halaman web yang dinamis, serta menerima dan menciptakan cookies, dan bahkan PHP bisa melakukan lebih dari itu. Arti script server-side adalah, agar dapat menjalankan script ini dibutuhkan tiga program utama, yaitu web-server (dapat berupa IIS dari windows atau apache), modul PHP dan juga web browser.

Sistem kerja dari PHP diawali dengan permintaan yang berasal dari halaman website oleh browser. Berdasarkan URL atau alamat website dalam jaringan internet, browser akan menemukan sebuah alamat dari webserver, mengidentifikasi halaman yang dikehendaki, dan menyampaikan segala informasi yang dibutuhkan oleh webserver. Selanjutnya webserver akan mencarikan berkas yang diminta dan menampilkan isinya di browser.

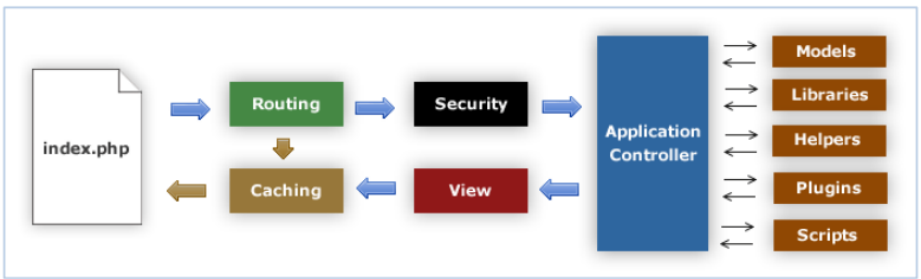
Browser yang mendapatkan isinya segera menerjemahkan kode HTML dan menampilkannya.[AGI-12]

2.7 *Code Igniter*

CodeIgniter (CI) adalah sebuah kerangka (framework) pembangunan aplikasi atau mudahnya disebut toolkit, untuk developer yang akan membuat aplikasi web dengan PHP. Tujuan CI adalah supaya pembangunan aplikasi lebih cepat dibanding menulis source code dari awal, karena CI telah meyediakan banyak library untuk proses-proses yang sering digunakan pada suatu aplikasi, dan juga dengan kemudahan dalam menggunakan library tersebut serta kesederhaan penggunaannya[CAU-11].

2.7.1 Alur Proses Aplikasi

Gambar berikut ini mengilustrasikan alur proses data pada CI.



Gambar 2. 1 Alur Proses Aplikasi pada CI

Sumber : [CAU-11]

a. Index.php berfungsi sebagai pengendali awal, menginisialisasi sumber daya utama yang dibutuhkan CodeIgniter.
b. Router memeriksa paket HTTP request untuk menentukan aksi apa yang harus dilakukan oleh sistem.
c. Jika cache tersedia, maka halaman langsung dikirim ke browser, eksekusi sistem yang normal akan dilewati.
d. Security. Sebelum Application Controller dieksekusi, paket HTTP request dan semua data yang dikirimkan pengguna akan disaring terlebih dahulu oleh Security Class.

e. Application Controller menginisialisasi model, library utama, helpers dan semua sumberdaya yang dibutuhkan untuk setiap request.
f. Antarmuka aplikasi (view) yang sudah disiapkan dikirimkan ke browser. Jika caching diaktifkan, maka view akan disimpan sementara untuk request yang sama berikutnya.

2.7.2 *Model, View, Controller, Libraries, Helper*

Seperti framework PHP pada umumnya, CodeIgniter menggunakan konsep MVC serta menyediakan banyak library dan helper untuk digunakan. Berikut penjelasan mengenai model, view, controller, library dan helper[CAU-11].

a. Model, merepresentasikan struktur data. Biasanya class model akan berisi fungsifungsi untuk mengambil data, insert data, dan update data ke *database*. Pada CI, model tidak harus digunakan, tapi hal ini akan menghilangkan konsep MVC itu sendiri.
b. View, adalah informasi / halaman yang ditampilkan ke pengguna. Sebuah view biasanya adalah sebuah web page, tapi di CodeIgniter view juga dapat berupa bagianbagian halaman web, seperti header dan footer. Bahkan view juga dapat berupa halaman RSS.
c. Controller, berfungsi sebagai penghubung antara Model, View dan dengan sumber daya lain yang digunakan untuk memproses HTTP request. Controller juga biasanya berfungsi sebagai inti pemrosesan lojik aplikasi.
d. Libraries, adalah macam-macam class yang masing-masing mempunyai fungsi khusus yang dapat digunakan untuk mengembangkan aplikasi. Contoh library *database*, email, validasi form, dan lain-lain.
e. Helper, seperti namanya berfungsi menolong untuk melakukan tugas-tugas tertentu. Setiap file helper terdiri dari kumpulan fungsi (function). Contoh URL Helper yang berfungsi untuk membuat link, Form helper untuk membuat elemen-elemen form. Tidak seperti

library, helper tidak menggunakan format Object Oriented, sehingga dapat digunakan dimanapun, baik itu di model, view, controller dan library.

2.7.3 *CodeIgniter URL*

Secara default, URL pada CodeIgniter didesain agar search-engine dan humanfriendly, menggunakan pendekatan segment-based.

example.com/index.php/class/function/parameter1/parameter2

a. Index.php merupakan segment ke-0.
b. Segment pertama merepresentasikan class controller yang diakses.
c. Segment kedua merepresentasikan nama method yang dipanggil pada class tersebut.
d. Segment ketiga dan seterusnya bersifat optional, merepresentasikan parameter masukan untuk fungsi yang dipanggil tersebut.

Index.php dapat dihilingkan dengan .htaccess sederhana. Jika ingin menghilangkan index.php, sebaiknya lakukan diawal sebelum source-code aplikasi dibuat, karena jika aplikasi sudah jadi dan index.php dihilangkan, dapat merubah struktur link seluruh aplikasi, contohnya menu, dan paging.

2.8 MySQL

MySQL adalah Sebuah program database server yang mampu menerima dan mengirimkan datanya sangat cepat, multi user serta menggunakan peintah dasarSQL (Structured Query Language). MySQL merupakan duabentuk lisensi, yaitu FreeSoftware dan Shareware. MySQLyang biasa kita gunakan adalah MySQL FreeSoftware yang berada dibawah Lisensi GNU/GPL (General PublicLicense). MySQL Merupakan sebuah database server yang free, artinya kita bebas menggunakan database ini untuk keperluan pribadi atau usaha tanpa harus membeli atau membayar lisensinya. MySQL pertama kali dirintis oleh seorang programmer database bernama Michael Widenius. Selain database server, MySQL juga merupakan program yang dapat mengakses suatu database MySQL yang

berposisi sebagai Server, yang berarti program kita berposisi sebagai Client. Jadi MySQL adalah sebuah database yang dapat digunakan sebagai Client maupun server. Database MySQL merupakan suatu perangkat lunak database yang berbentuk database relasional atau disebut Relational Database Management System (RDBMS) yang menggunakan suatu bahasa permintaan yang bernamaSQL (Structured QueryLanguage).[SAP-12]

2.9 Rekayasa Pelangkat Lunak

Perangkat Lunak Merupakan program-program komputer dan dokumentasi yang berkaitan. Produk perangkat lunak dibuat untuk pelanggan tertentu ataupun untuk pasar umum terdiri dari[NUG-09]:

1. Generik – dibuat untuk dijual ke suatu kumpulan pengguna yang berbeda
2. Bespoke (custom) – dibuat untuk suatu pengguna tunggal sesuai dengan spesifikasinya.

Rekayasa perangkat lunak berasal dari 2 kata yaitu Software(Perangkat Lunak) dan Engineering (Rekayasa). Perangkat Lunak (Software) adalah source code pada suatu program atau sistem. Perangkat lunak tidak hanya dokumentasi terhadap source code tapi juga dokumentasi terhadap sesuatu yang dibutuhkan selama pengembangan, instalasi, penggunaan dan pemeliharaan sebuah sistem. *Engineering* atau Rekayasa adalah aplikasi terhadap pendekatan sistematis yang berdasar atas ilmu pengetahuan dan matematis serta aplikasi tentang produksi terhadap struktur,mesin, produk, proses atau sistem. Rekayasa Perangkat Lunak adalah suatu disiplin rekayasa yang berkonsentrasi terhadap seluruh aspek. Rekayasa Perangkat Lunak (RPL) juga merupakan pendekatan sistematis dan matematis untuk membangun, memelihara dan mengenyahkan perangkat lunak. Dari cara pandang lain, RPL adalah pendekatan sistematis untuk merekayasa perangkat lunak yang handal/bermutu, tepat waktu dan dengan biaya yang optimal[NUG-09].

2.9.1 Analisis Kebutuhan Perangkat Lunak

Analisis kebutuhan perangkat lunak (*software requirements analysis*) merupakan aktivitas awal dari siklus hidup pengembangan perangkat lunak. Untuk proyek-proyek perangkat lunak yang besar, analisis kebutuhan dilaksanakan setelah tahap rekayasa sistem/informasi dan software project planning. Tujuan pelaksanaan analisis kebutuhan untuk memahami masalah secara menyeluruh (komprehensif) yang ada pada perangkat lunak yang akan dikembangkan dan mendefinisikan apa yang harus dikerjakan oleh perangkat lunak untuk memenuhi keinginan pelanggan[NUG-09].

Secara teknis pelaksanaan pekerjaan analisis kebutuhan perangkat lunak pada dasarnya terdiri dari urutan aktivitas[NUG-09]:

1. Mempelajari dan memahami persoalan
2. Mengidentifikasi kebutuhan pengguna
3. Mendefinisikan kebutuhan perangkat lunak
4. Membuat dokumen spesifikasi kebutuhan perangkat lunak (SKPL)
5. Mengkaji ulang (review) kebutuhan

Sedangkan menurut Pressman, analisis kebutuhan perangkat lunak dapat dibagi menjadi lima area pekerjaan[PRE-01], yaitu:

1. Pengenalan masalah
2. Evaluasi dan sistesis
3. Pemodelan
4. Spesifikasi
5. Tinjau ulang (review)

2.9.2 Perancangan Perangkat Lunak Dengan UML

Unified Modelling Language (UML) adalah notasi yang lengkap untuk membuat visualisasi model suatu sistem. Sistem berisi informasi dan fungsi tapi secara normal digunakan untuk memodelkan sistem komputer. Sebagaimana halnya bahasa pemodelan, UML mengijinkan deskripsi dari sistem dibuat dengan mendetail pada setiap level abstraksi. Notasi tersebut akan mendefinisikan sistem dengan arsitektur berorientasi obyek.

UML tidak hanya merupakan bahasa pemodelan berorientasi obyek, akan tetapi merupakan pemodelan untuk spesifikasi, visualisasi, konstruksi, dokumentasi proses sistem secara intensif. Pada dasarnya, UML berhubungan dengan pengetahuan yang ditangkap, dikomunikasikan dan dikembangkan.

1. **Use Case Diagram**

Use case diagram menggambarkan fungsionalitas yang diharapkan dari sebuah sistem. Yang ditekankan adalah "apa" yang diperbuat sistem, dan bukan "bagaimana". Sebuah *use case*merepresentasikan sebuah interaksi antara aktor dengan sistem. *Use case* merupakan sebuahpekerjaan tertentu, misalnya log in ke sistem, meng-*create* sebuah daftar belanja, dan sebagainya.Seorang/sebuah aktor adalah sebuah entitas manusia atau mesin yang berinteraksi dengan system untuk melakukan pekerjaan-pekerjaan tertentu.

2. **Class Diagram**

Class adalah sebuah spesifikasi yang jika diinstansiasi akan menghasilkan sebuah objek dan merupakan inti dari pengembangan dan desain berorientasi objek. *Class* menggambarkan keadaan (atribut/properti) suatu sistem, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut (metoda/fungsi). *Class diagram* menggambarkan struktur dan deskripsi *class, package* dan objek beserta hubungan satu sama lain seperti *containment*, pewarisan, asosiasi, dan lain-lain.

3. **Activity Diagram**

Activity diagrams menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alir berawal, *decision* yang mungkin terjadi, dan bagaimana mereka berakhir. *Activity diagram* juga dapat menggambarkan proses paralel yang mungkin terjadi pada beberapa eksekusi.

4. **Sequence Diagram**

Sequence Diagram menunjukkan interaksi obyek yang diatur dalam satuan waktu. *Sequence Diagram* menangkap obyek dan class yang terlibat dalam scenario dan urut-urutan message yang ditukar diantara obyek diperlukan untuk melaksanakan fungsionalitas scenario. *Sequence Diagram* berasosiasi dengan use case selama proses pengembangan.

2.9.3 Implementasi

Implementasi perangkat lunak dilakukan untuk merealisasikan desain dari perangkat lunak menggunakan bahasa pemrograman berorientasi objek (*Object Oriented Programming Languages/OOP*).

2.9.4 Pengujian Perangkat Lunak

Pengujian perangkat lunak merupakan sabuah set dari langkah-langkah dimana kita dapat menempatkan desain spesifik *test case* dan metode tes. Berbagai strategi tes perangkat lunak telah diajukan dalam sumber literatur[SOM-10].

1. Pengujian Validasi

Pada kulminasi pengujian terintegrasi, perangkat lunak secara lengkap dirakit sebagai suatu paket; kesalahan *interfacing* telah diungkap dan dikoreksi, dan seri akhir dari pengujian perangkat lunak, yaitu pengujian validasi dapat dimulai. Validasi dapat ditentukan dengan berbagai cara, tetapi definisi yang sederhana adalah bahwa validasi berhasil bila perangkat lunak berfungsi dengan cara yang dapat diharapkan secara bertanggung jawab oleh pelanggan. Validasi perangkat lunak dicapai melalui sederetan pengujian *black-box* yang memperlihatkan konformitas dengan persyaratan. Rencana pengujian menguraikan kelas-kelas pengujian yang akan dilakukan, dan prosedur pengujian menentukan *test case* spesifik yang akan digunakan untuk mengungkap kesalahan dalam konformitas dengan persyaratan. Baik rencana dan prosedur didesain untuk memastikan apakah

semua persyaratan fungsional dipenuhi; semua persyaratan kinerja dicapai; dokumentasi benar dan direkayasa oleh manusia: dan persyaratan lainnya dipenuhi (transportabilitas, kompatibilitas, pembetulan kesalahan, maintanabilitas). [PRE-01]

2. Pengujian Performa

Setelah semua langkah pengujian perangkat lunak secara terstruktur dilakukan, maka perlu dilakukan pengujian sistem di lingkungan dimana dia bekerja untuk mengetahui performa dari perangkat lunak tersebut. Pengujian sistem dirancang untuk menguji kinerja *run-time* dari perangkat lunak dalam konteks sistem terintegrasi. Pengujian performa melibatkan *monitoring* pemanfaatan sumber daya dari perangkat lunak yang diuji seperti perangkat lunak pendukung dan perangkat keras. Pengujian performa dilakukan secara spesifik sesuai dengan tipe perangkat lunak yang diuji. Pengujian performa bertujuan untuk mengungkap situasi yang menyebabkan degradasi dan kemungkinan kegagalan sistem. [PRE-01]

BAB III
METODOLOGI PENELITIAN DAN PERANCANGAN

3.1 Metode Penelitian

Langkah-lagkah yang dilakukan dalan penelitian ini yaitu : studi litelatur, analisis kebutuhan, perancangan perangkat lunak, implementasi perangkat lunak, pengujian perangkat lunak serta pengambilan kesimpulan dan saran.

3.1.1 Studi Literatur

Studi literatur merupakan penelusuran literatur yang bertujuan untuk mempelajari tentang penjelasan dasar teori yang digunakan untuk menunjang penulisan skripsi. Penulisan literatur dapat bersumber dari buku, media, pakar maupun hasil penelitian orang lain. Penyusunan dasar teori dilakukan setelah mendapatkan referensi yang tepat untuk mendukung penulisan penelitian ini.

3.1.2 Analisis Kebutuhan

Analisis kebutuhan bertujuan untuk menganalisis semua kebutuhan yang diperlukan dalam pembuatan perangkat lunak. Langkah-langkah dalam analisis kebutuhan ini antara lain (1) pembuatan alur kinerja aplikasi, (2) identifikasi aktor, (3) Identifikasi kebutuhan, (4) pembuatan *use case diagram.*

1. Pembuatan Alur Kerja Aplikasi

Tujuan alur kerja aplikasi ini adalah untuk mengetahui bagaimana alur jalannya aplikasi tersebut. Dalam hal ini *hash* dan *digital signature* yang diterapkan dalam sistem *e-voting* digunakan untuk membantu memaksimalkan fungsi *e-voting* sehingga dalam penggunaannya didapatkan hasil yang terpercaya. Selain itu, aplikasi ini juga dapat mengidentifikasi hasil *voting* yang telah mengalami perubahan pada proses pengiriman.

2. Identifikasi Aktor

Aktor adalah pengguna sistem, bisa berupa orang ataupun sistem terotomatisasi lain. Tahap ini dilakukan untuk melakukan identifikasi aktor-aktor yang berinteraksi dengan sistem. Dalam hal ini, aktor yang terlibat adalah *voter* dan administrator.

3. Identifikasi Kebutuhan

Identifikasi kebutuhan terdiri dari 2 hal, yaitu :

- Kebutuhan fungsional

  Kebutuhan fungsional merupakan kebutuhan utama yang dibutuhkan dalam menjalankan sistem, seperti pengambilan data dari database, menampilkan data yang diproses, dan lain-lain.

- Kebutuhan non fungsional

  Analisis kebutuhan non fungsional ini merupakan pendukung sistem yang akan dijalankan. Kebutuhannya meliputi kebutuhan perangkat keras dan kebutuhan perangkat lunak.

4. Pembuatan Use Case Diagram

Fungsi-fungsi yang disediakan oleh sistem dan interaksi aktor dengan sistem akan dimodelkan dalam *use case diagram.* Dalam *use case diagram* ini akan ditampilkan fitur-fitur yang terdapat di dalam aplikasi ini. Selain pembuatan diagram, fitur-fitur juga ditampilkan dalam tabel-tabel yang berisi penjelasan-penjelasan tentang kondisi dari setiap *use case* yang dibuat.

5. Pembuatan Activity Diagram

Activity diagram menunjukkan aliran dari aktivitas ke aktivitas dalam suatu sistem. *Activity diagram* mengatasi tampilan dinamis dari sebuah sistem. *Activity diagram* sangat penting dalam pemodelan fungsi sistem dan menekankan aliran kontrol antara objek-objek.

3.1.3 Perancangan Perangkat Lunak

Perancangan perangkat lunak dilakukan setelah semua kebutuhan sudah didapatkan dari proses analisis kebutuhan. Perancangan perangkat lunak terdiri dari :

1. Perancangan diagram kelas untuk memodelkan kelas dan *interface* yang dibutuhkan dalam pembuatan perangkat lunak.
2. Pembuatan diagram sekuen untuk menggambarkan interaksi antar objek yang disusun berdasarkan urutan waktu.
3. Pembuatan diagram *activity* untuk menggambarkan alur kinerja dari aktivitas sistem yang akan dibuat.
4. Perancangan tampilan antar muka aplikasi yang terdiri dari :
   a. Menu Aplikasi
      - Menu Aplikasi User *Voter*
      - Menu Aplikasi User Administrator

3.1.4 Implementasi Perangkat Lunak

Setelah melakukan perancangan, tahap selanjutnya adalah implementasi atau pembuatan aplikasi. Dalam implementasinya, aplikasi ini menggunakan bahasa pemrograman PHP dan database MySQL.

3.1.5 Pengujian Perangkat Lunak

Untuk memastikan bahwa aplikasi yang dibuat berjalan sesuai dengan yang diinginkan, maka perlu dilakukan suatu pengujian. Hal ini bertujuan tidak akan terjadi kesalahan atau *error* saat aplikasi digunakan oleh pengguna. Pengujian yang dilakukan terhadap perangkat lunak ini meliputi pengujian validasi, pengujian verifikasi, dan pengujian performa.

3.1.6 Pengambilan Kesimpulan

Pengambilan kesimpulan dilakukan setelah semua tahapan analisis kebutuhan, perancangan, implementasi dan pengujian sistem aplikasi telah selesai dilakukan. Kesimpulan diambil dari hasil pengujian dan analisis terhadap sistem yang dibangun. Tahap terakhir dari penulisan adalah saran yang dimaksudkan

untuk memperbaiki kesalahan-kesalahan yang terjadi, menyempurnakan penulisan dan untuk memberikan pertimbangan atas pengembangan aplikasi selanjutnya.

3.2 Perancangan

Perancangan perangkat lunak dilakukan setelah semua kebutuhan telah didapatkan dari proses analisis kebutuhan. Proses perancangan terdiri dari analisa kebutuhan perangkat lunak/keras dan perancangan perangkat lunak/keras.

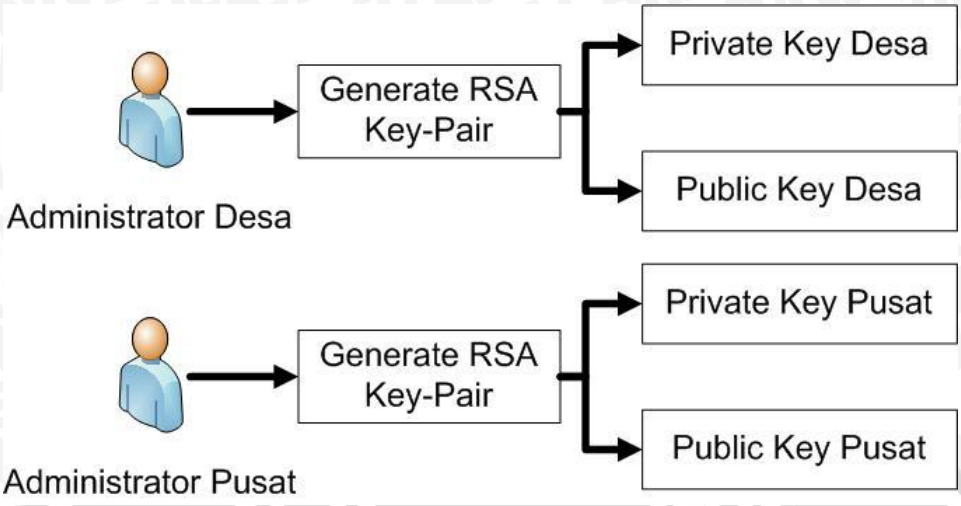
3.2.1 Analisa Kebutuhan Perangkat Lunak/Keras

Proses analisis kebutuhan mengacu pada gambaran umum sistem perangkat lunak *e-voting* yang menerapkan *hash* dan *digital signature* untuk verifikasi data hasil *voting*. Proses analisis kebutuhan ini diawali dengan penjabaran gambaran umum sistem, identifikasi aktor – aktor yang terlibat dalam sistem, penjabaran tentang daftar kebutuhan dan kemudian memodelkannya ke dalam diagram *use case*. Analisis kebutuhan ini bertujuan untuk menggambarkan kebutuhan yang harus disediakan oleh sistem agar dapat memenuhi kebutuhan pengguna.

1. Alur Kerja Aplikasi

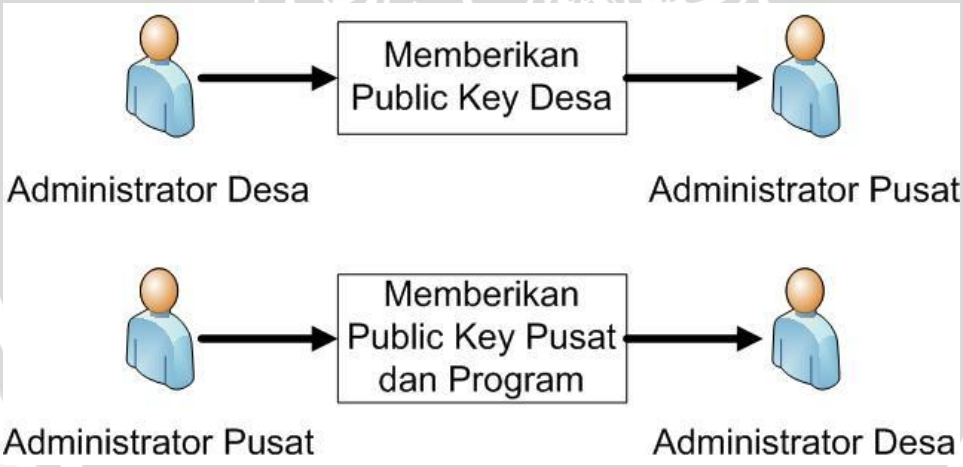
Sebelum membuat aplikasi, diperlukan sebuah alur guna mengetahui bagaimana kerja aplikasi yang akan dibuat dan mempermudah dalam menentukan gambaran umum aplikasi. Hal ini diperlukan agar pada proses implementasi akan lebih mudah terutama pada proses penulisan kode-kode program.

Aplikasi yang akan dikembangkan pada proyek skripsi ini adalah sebuah aplikasi *e-voting* berbasis web yang hasil dari *voting*nya memiliki *hash* dan *digital signature* yang digunakan untuk mengamankan data hasil *voting*. Secara fungsional aplikasi *e-voting* ini tidak berbeda jauh dengan aplikasi *e-voting* pada umumnya, hanya saja pada aplikasi ini diterapkan sistem keamanan yang dapat membantu proses *voting* yang dilakukan di desa agar hasil voting yang didapat dapat terkirim ke pusat dengan aman. Alur sistem secara umum di pada gambar berikut.



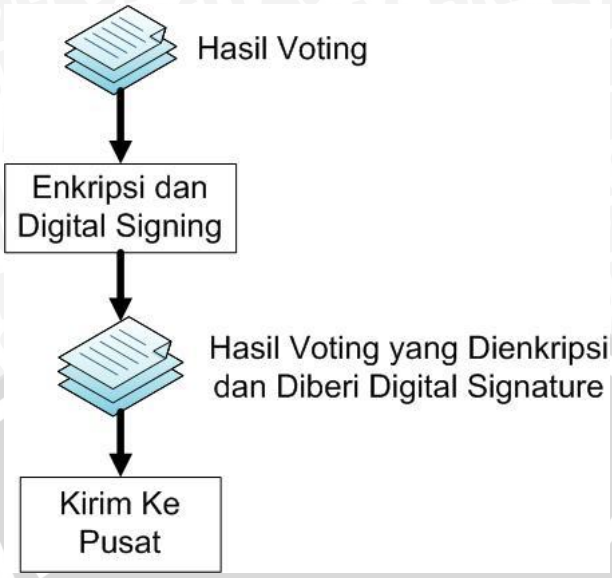
Gambar 3. 1 Diagram proses awal e-voting

Pada gambar 3.1 proses awal administrator pusat dan desa akan melakukan *generate* sepasang kunci RSA yang menghasilkan kunci privat dan kunci publik masing-masing.



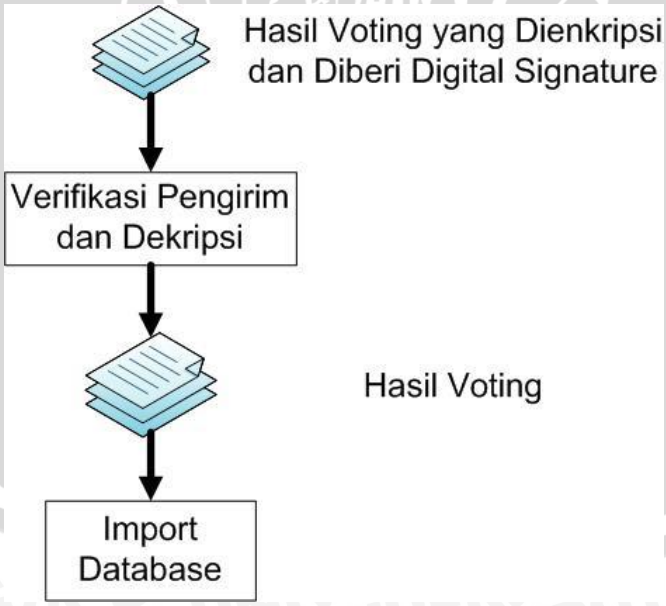
Gambar 3. 2 Diagram pertukaran kunci publik

Pada gambar 3.2 setelah masing-masing memiliki sepasang kunci privat dan publik, maka kunci publik ditukarkan sehingga pada akhirnya administrator desa memiliki kunci privat desa dan kunci publik pusat sedangkan administrator pusat memiliki kunci privat pusat dan kunci publik desa.



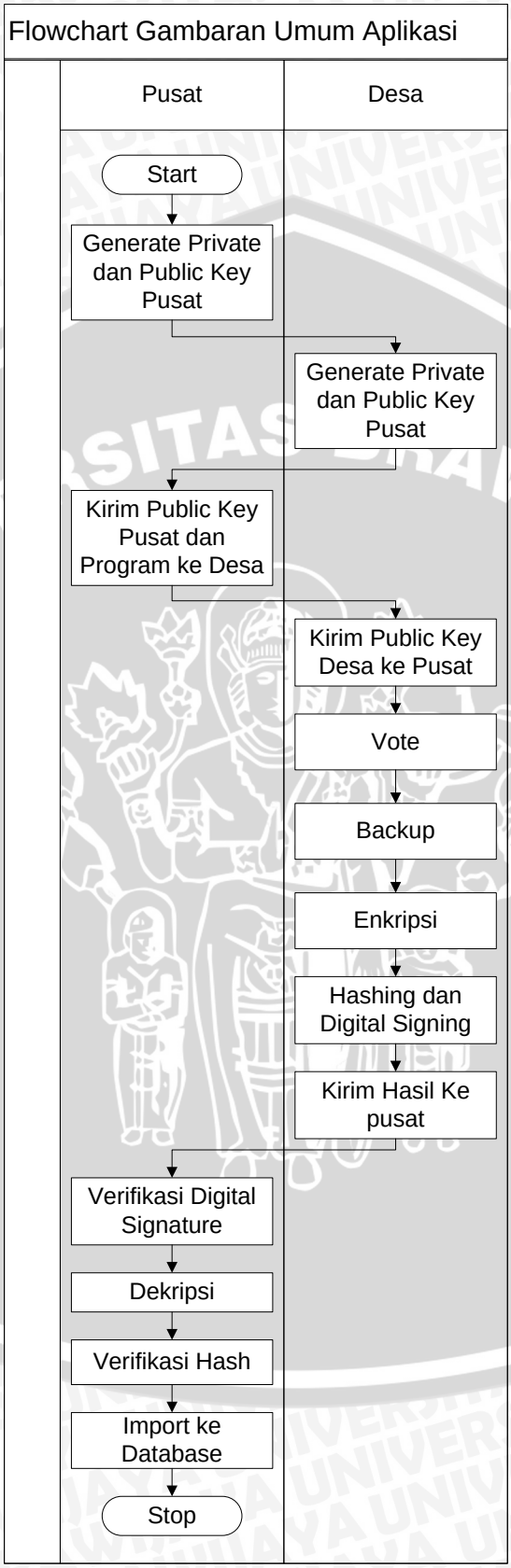
Gambar 3. 3 Diagram alur aplikasi yang dikirim ke desa secara umum

Pada gambar 3.3 hasil voting yang didapat di enkripsi dan diberi *digital signature* kemudian dikirimkan ke pusat.



Gambar 3. 4 Diagram hasil voting setelah sampai di pusat secara umum

Pada gambar 3.4 hasil voting yang telah diterima oleh pusat akan dilakukan verifikasi pengirim. Jika pengirim terverifikasi maka akan dilanjutkan pada proses dekripsi. Sedangkan jika tidak terverifikasi maka file tidak dapat didekripsi. Setelah file di dekripsi kemudian akan menghasilkan data yang dapat diimport ke dalam database oleh administrator pusat. Secara umum proses diatas ditunjukkan pada gambar 3.5 berikut.



Gambar 3. 5 Flowchart gambaran umum aplikasi

2. Identifikasi Aktor

Sistem *e-voting* yang menerapkan *hash* dan *digital signature* untuk verifikasi data hasil voting ini mempunyai dua aktor, yaitu user *voter* dan user administrator. User disini dibedakan menjadi dua karena alur tiap-tiap user berbeda. Identifikasi aktor dijelaskan pada tabel 3.1

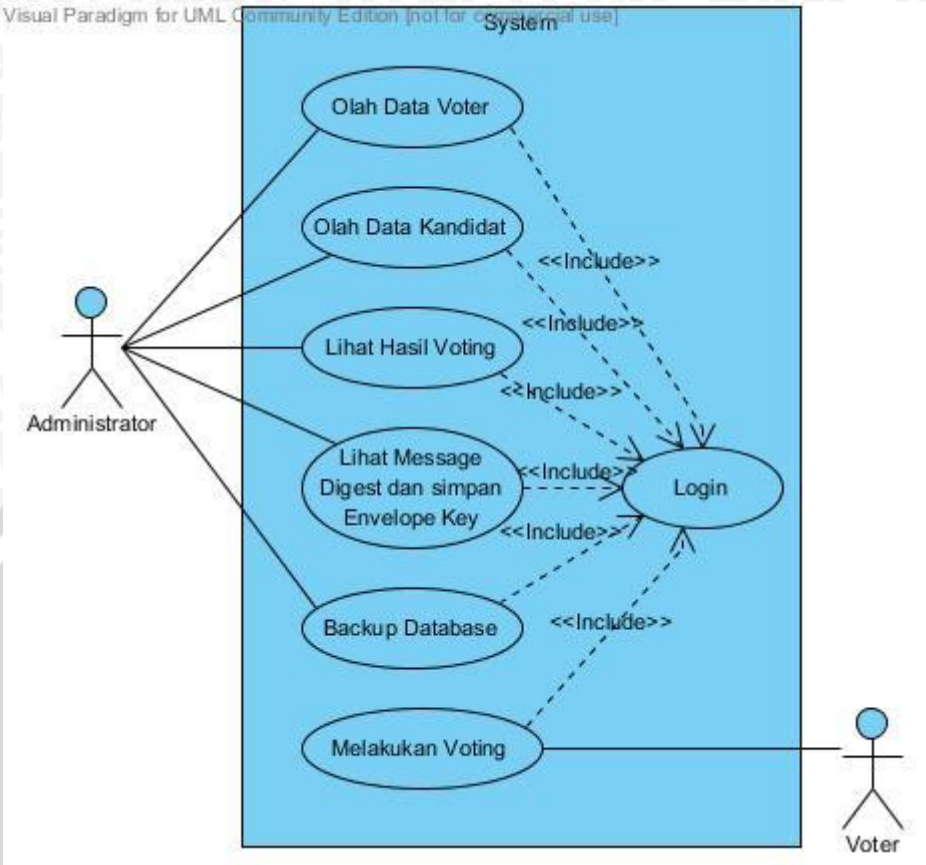
Tabel 3. 1 Identifikasi Aktor

Aktor	Deskripsi
Administrator	*Administrator* adalah aktor yang mengatur segala aktifitas sistem. Aktifitas ini meliputi *input data*, *lihat data*, dan *delete data*. *Administrator* inilah yang memiliki hak untuk mengelola elemen-elemen yang terdapat dalam sistem tersebut lebih lanjut.
Voter	*Voter* adalah aktor pengguna sistem yang memiliki hak untuk melakukan proses voting melalui akunnya.

3. Identifikasi Kebutuhan dan Analisis *Use Case*

Kebutuhan Sistem *e-voting* yang menerapkan *hash* dan *digital signature* untuk verifikasi data hasil voting terdiri dari kebutuhan fungsional dan kebutuhan nonfungsional. Pada skenario *use case* akan diberikan uraian nama *use case*, aktor yang berhubungan dengan *use case* tersebut, tujuan dari *use case* , deskripsi global tentang *use case*, kondisi awal yang harus dipenuhi, dan kondisi akhir yang diharapkan setelah berjalannya fungsional *use case*. Selain itu juga akan diberikan ulasan yang berkaitan dengan tanggapan dari sistem atas satu aksi yang diberikan oleh aktor.

Gambar 3.6 adalah diagram *use case* sistem e-voting dengan menerapkan *hash* dan *digital signature* untuk verifikasi data hasil voting.



Gambar 3. 6 Diagram *Use Case* Sistem E-Voting

Tabel 3. 2 Spesifikasi kebutuhan fungsional sistem e-voting

Nomor SRS	*Requirements*		Aktor
SRS_001_01	Sistem dapat mengolah data *voter*	Sistem dapat menampilkan data *voter*	Administrator
		Sistem dapat menambah data *voter*	
		Sistem dapat menghapus data *voter*	
SRS_001_02	Sistem dapat mengolah data kandidat	Sistem dapat menampilkan data kandidat	Administrator
		Sistem dapat menambah data kandidat	
		Sistem dapat menghapus data kandidat	
SRS_001_03	Sistem dapat menampilkan hasil *voting*		Administrator
SRS_001_04	Sistem dapat menampilkan *message digest* dan menyimpan *envelope-key*		Administrator
SRS_001_05	Sistem dapat melakukan *backup database*		Administrator
SRS_002_01	Sistem mampu melakukan *voting*		Voter

Secara lebih mendetail, masing-masing *use case* pada gambar di atas akan dijabarkan pada tabel-tabel di bawah.

1. *Use case* Olah Data *Voter*

Tabel 3. 3 Tabel *Use Case* Olah Data *Voter*

Skenario Kasus Pada Sistem	
Nama	Olah Data *Voter*
Tujuan	Untuk menampilkan dan manipulasi pada data *voter*
Deskripsi	*Use case* ini menjelaskan bagaimana administrator dapat melakukan proses manipulasi data *voter*.
Aktor	Administrator
Skenario Utama	
Kondisi Awal	Administrator telah melalui proses autentikasi dan otorisasi pada sistem dan telah memasuki halaman voter
Aksi Aktor	**Reaksi Sistem**
▪ **Aktor melihat data *voter***	Sistem menampilkan data *voter*
▪ **Aktor akan menambah data *voter***	Sistem menampilkan *form input* data *voter* lalu menyimpan pada sistem
▪ **Administrator menghapus data *voter* yang belum melakukan *voting***	Sistem melakukan penghapusan data *voter*
Kondisi Akhir	Sistem menampilkan daftar *voter*

2. *Use case* Olah Data Kandidat

Tabel 3. 4 Tabel *Use Case* Olah Data Kandidat

Skenario Kasus Pada Sistem	
Nama	Olah Data kandidat
Tujuan	Untuk menampilkan dan manipulasi pada data kandidat
Deskripsi	*Use case* ini menjelaskan bagaimana administrator dapat melakukan proses manipulasi data kandidat.
Aktor	Administrator

Skenario Utama	
Kondisi Awal	Administrator telah melalui proses autentikasi dan otorisasi pada sistem dan telah memasuki halaman kandidat.
Aksi Aktor	**Reaksi Sistem**
▪ **Aktor melihat data** kandidat	Sistem menampilkan data kandidat
▪ **Aktor akan menambah data** kandidat	Sistem menampilkan *form input* data kandidat lalu menyimpan pada sistem
▪ **Administrator menghapus data** kandidat **yang belum melakukan di-*vote***	Sistem melakukan penghapusan data kandidat
Kondisi Akhir	Sistem menampilkan daftar kandidat

3. *Use case* Lihat Hasil Voting

Tabel 3. 5 Tabel *Use Case* Lihat Hasil Voting

Skenario Kasus Pada Sistem	
Nama	Lihat Hasil Voting
Tujuan	Pengguna dapat melihat hasil voting
Deskripsi	*Use case* ini menjelaskan bagaimana administrator dapat melihat hasil voting
Aktor	Administrator
Skenario Utama	
Kondisi Awal	Administrator telah melalui proses autentikasi dan otorisasi pada sistem dan telah memasuki halaman result.
Aksi Aktor	**Reaksi Sistem**
▪ **Aktor melihat data hasil voting**	Sistem menampilkan data hasil voting
Kondisi Akhir	Sistem menampilkan halaman hasil voting

4. *Use case* Lihat *Message digest* dan Simpan *Envelope-Key*

Tabel 3. 6 Tabel *Use Case* Lihat *Message digest* dan *Envelope-Key*

Skenario Kasus Pada Sistem	
Nama	Lihat *Message digest* dan Simpan *Envelope-Key*
Tujuan	Untuk melihat *message digest* dan menyimpan *envelope-key*
Deskripsi	*Use case* ini menjelaskan bagaimana administrator dapat melihat *message digest* dan menyimpan *envelope-key*
Aktor	Administrator
Skenario Utama	
Kondisi Awal	Administrator telah melalui proses autentikasi dan otorisasi pada sistem dan telah melalui proses *backup database*
Aksi Aktor	**Reaksi Sistem**
▪ **Administrator melihat *message digest* dan menyimpan *envelope-key***	Sistem menampilkan *message digest* dan menyimpan *envelope-key*
Kondisi Akhir	Sistem menampilkan halaman yang berisi *message digest* dan menyimpan *envelope-key*

5. *Use case* Backup Database

Tabel 3. 7 Tabel *Use Case* Backup Database

Skenario Kasus Pada Sistem	
Nama	Backup Database
Tujuan	Untuk melakukan backup pada database
Deskripsi	*Use case* ini menjelaskan bagaimana administrator dapat melakukan proses backup database
Aktor	Administrator
Skenario Utama	
Kondisi Awal	Administrator telah melalui proses autentikasi dan otorisasi pada sistem dan telah memasuki halaman *result*
Aksi Aktor	**Reaksi Sistem**
▪ **Aktor menekan tombol backup**	Sistem melakukan backup database

database	
Kondisi Akhir	Sistem menyimpan hasil backup dalam sistem

6. *Use case Download Database* dan *Message digest*

Tabel 3. 8 Tabel *Use Case Download Database* dan *Message digest*

Skenario Kasus Pada Sistem	
Nama	*Download Database* dan *Message digest*
Tujuan	Untuk melakukan *download database* dan *message digest*
Deskripsi	*Use case* ini menjelaskan bagaimana administrator dapat melakukan *download database* dan *message digest*
Aktor	Administrator
Skenario Utama	
Kondisi Awal	Administrator telah melalui proses autentikasi dan otorisasi pada sistem dan telah memasuki halaman *result* dan melakukan *backup database*
Aksi Aktor	**Reaksi Sistem**
▪ **Aktor menekan tombol *download database***	Sistem melakukan *download database*
▪ **Aktor menekan tombol *download hash.***	Sistem melakukan *download Message digest*
Kondisi Akhir	Aktor menerima *file database* dan *hash* dari sistem

7. *Use case* Melakukan *Vote*

Tabel 3. 9 Tabel *Use Case* Melakukan *Vote*

Skenario Kasus Pada Sistem	
Nama	Melakukan *Vote*
Tujuan	*Voter* dapat melakukan *vote*
Deskripsi	*Use case* ini menjelaskan bagaimana *voter* melakukan proses *vote*
Aktor	*Voter*

Skenario Utama		
Kondisi Awal	*Voter* telah melalui proses autentikasi dan otorisasi pada sistem dan telah memasuki halaman *vote*	
Aksi Aktor		**Reaksi Sistem**
▪ ***Voter* melihat data** kandidat		Sistem menampilkan data kandidat
▪ ***Voter* memilih *radio* pada** kandidat **yang dipilih dan memencet tombol *vote***		Sistem melakukan input data *voting* yang dimasukkan oleh *voter* ke dalam sistem
Kondisi Akhir		*Voter* selesai melakukan *vote* sehingga halaman *vote* dihilangkan

Kebutuhan non-fungsional sistem e-voting dengan menerapkan *hash* dan *digital signature* untuk verifikasi data hasil voting ditunjukkan pada tabel 3.10.

Tabel 3. 10 Spesifikasi kebutuhan non-fungsional

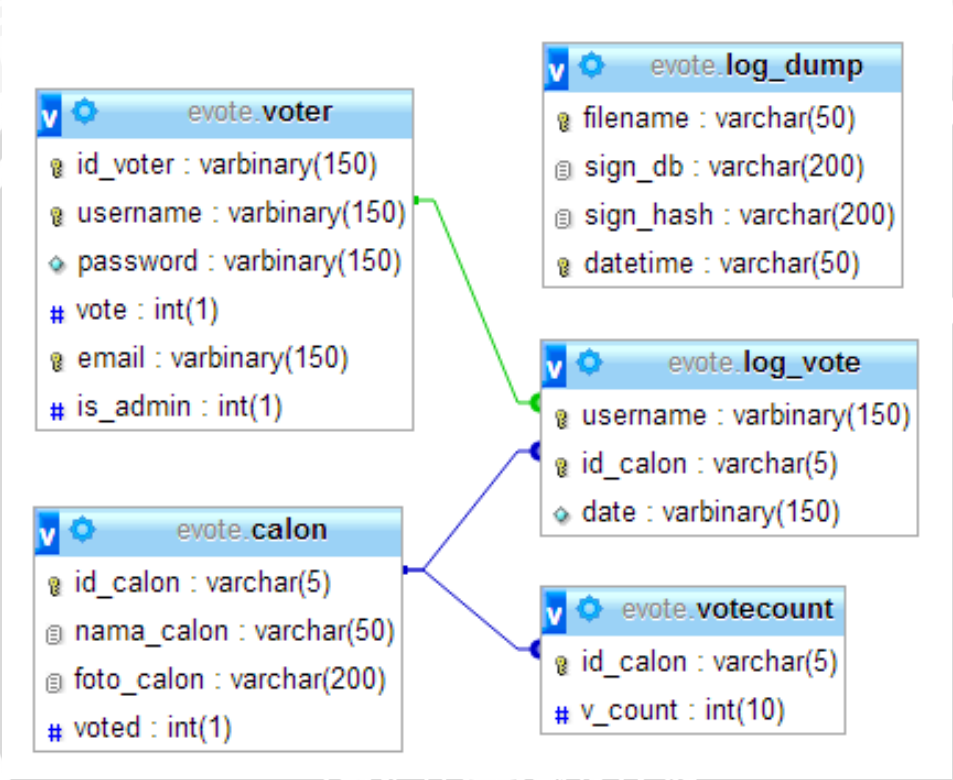
Parameter	Deskripsi Kebutuhan
Interoperability	Fungsi-fungsi semua operasi dalam perangkat lunak harus dapat berjalan dengan baik, khususnya pada pengolahan data.
Portability	Perangkat lunak harus dapat digunakan di berbagai *browser*.
Security	Keamanan yang diterapkan pada perangkat lunak hanya difokuskan pada aspek integritas. Perangkat lunak harus dapat memberi digital signature dan memverifikasi *digital signature* yang ada pada data hasil voting yang terkirim. Sehingga dapat memverifikasi pengirim dari data hasil voting apakah berasal dari pengirim yang sebenarnya atau bukan. Perangkat lunak harus dapat melakukan enkripsi dan dekripsi pada hasil *e-voting*.

Pada penelitian ini yang difokuskan adalah pada *security*, khususnya pada aspek integritas data. Pengujian yang dilakukan menggunakan pengujian verifikasi yaitu dengan menguji validitas *digital signature* dan melakukan dekripsi pada hasil *e-voting* yang terkirim.

3.2.2 Perancangan Perangkat Lunak/Keras

1. Perancangan Database

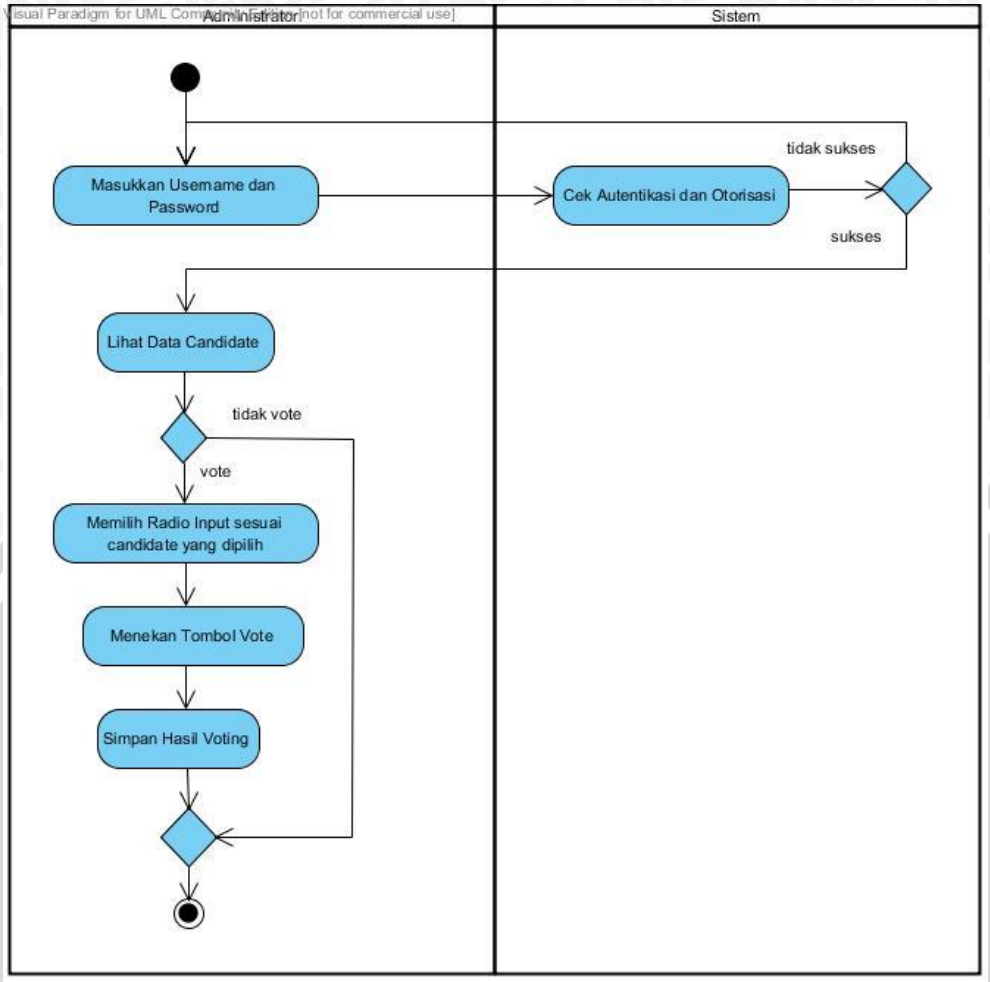
Perancangan database dijelaskan pada gambar 3.7. Database memiliki 5 tabel yaitu tabel voter yang menyimpan data *voter*, tabel calon yang menyimpan data kandidat, tabel log_vote menyimpan hasil voting, tabel *votecount* menyimpan jumlah *voting*, dan tabel log_dump yang menyimpan keterangan *backup* yang dilakukan pada *database*.



Gambar 3. 7 Rancangan Database Sistem

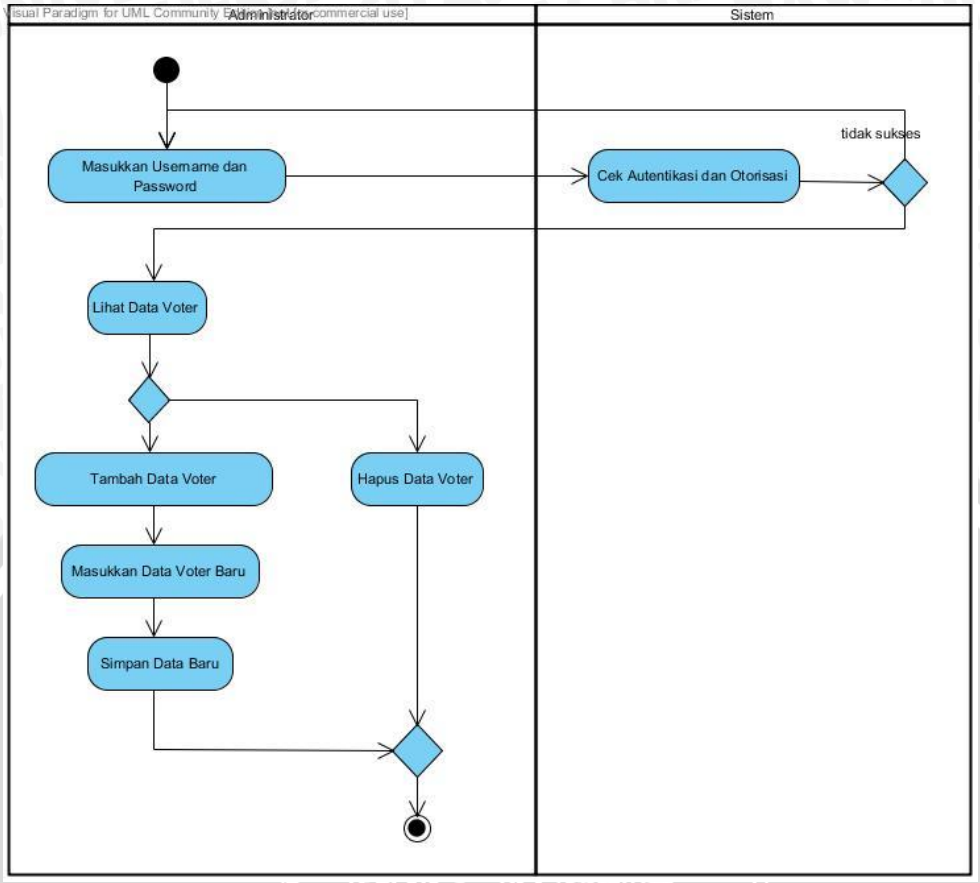
2. Perancangan Aktifitas

Activity diagram digunakan untuk menjelaskan alur proses dari penggunaan sistem. *Activity diagram* dikelompokkan menjadi 5 bagian, yaitu *activity diagram* untuk *use case* melakukan *voting*, olah data *voter*, olah data kandidat, lihat *result*, dan *backup* dan download *database* dan *digital signature*.



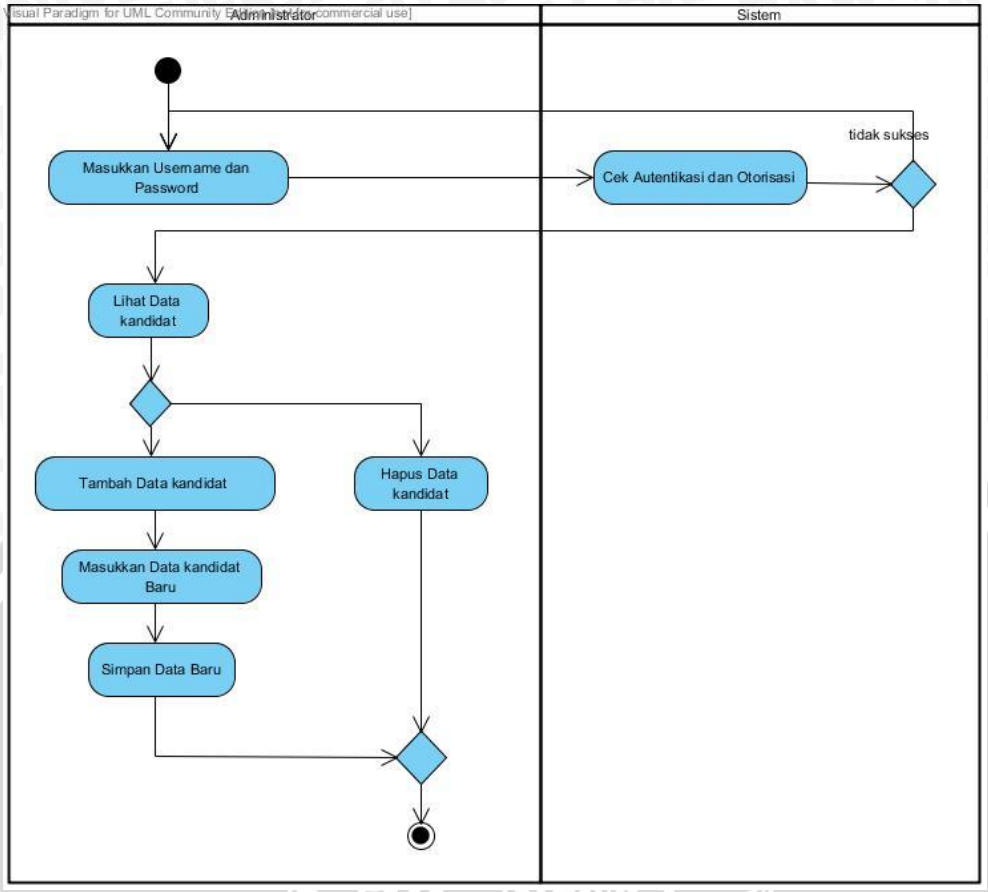
Gambar 3. 8 Diagram Aktivitas *Voting* untuk *Voter*

Pada Gambar 3.8 menunjukan diagram aktivitas *voting* untuk *voter*. Sebelum melakukan proses *voting*, *voter* harus melalui proses autentikasi dan otorisasi yang dilakukan oleh sistem. Setelah proses autentikasi dan otorisasi berhasil maka *voter* dapat melakukan *voting* dengan masuk ke halaman *vote*. Jika tidak melakukan *voting*, *voter* dapat langsung *logout*. Setelah masuk ke halaman *vote*, *voter* dapat memilih kandidat dengan memilih *radio input* kemudian menekan tombol *vote*. Setelah itu sistem akan menyimpan hasil *vote*. *Voter* yang telah melakukan *voting* tidak dapat melakukan *voting* ulang atau mengubah hasil *voting* sehingga halaman *vote* dihilangkan setelah selesai melakukan *voting*.



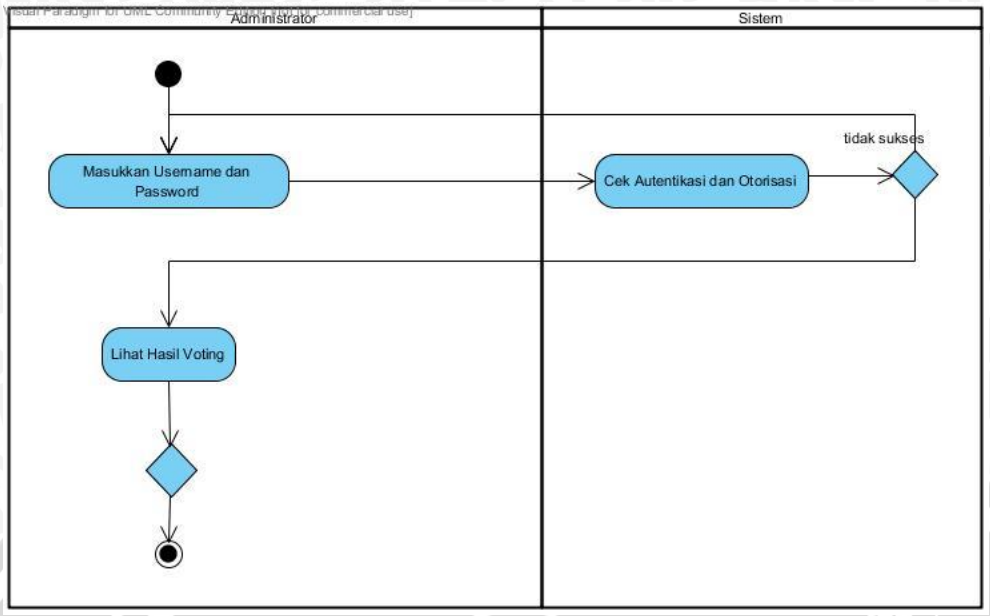
Gambar 3. 9 Diagram Aktivitas Olah Data Voter untuk Administrator

Pada Gambar 3.9 menunjukan diagram aktivitas olah data voter untuk administrator. Sebelum melakukan proses *voting*, administrator harus melalui proses autentikasi dan otorisasi yang dilakukan oleh sistem. Setelah proses autentikasi dan otorisasi berhasil maka administrator dapat melihat data *voter* pada halaman *voter*. Kemudian administrator dapat melakukan penambahan dan penghapusan *voter*. *Voter* yang dapat dihapus hanya *voter* yang belum melakukan *voting*.



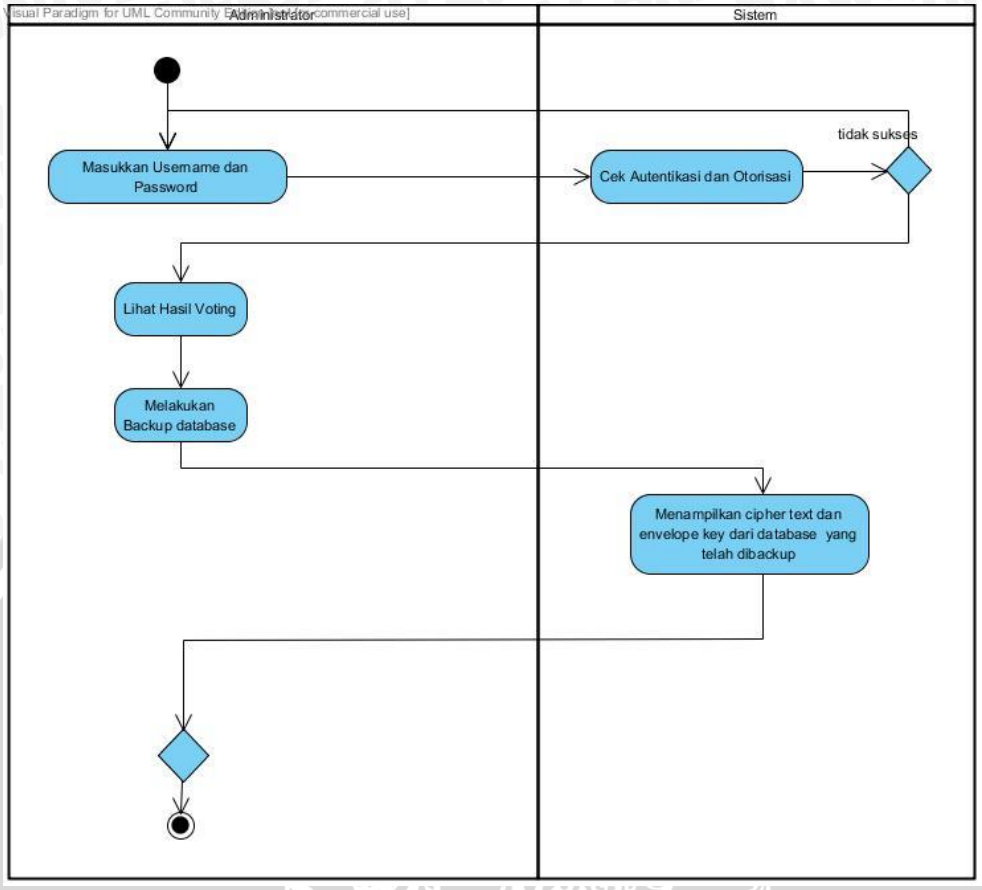
Gambar 3. 10 Diagram Aktivitas Olah Data Kandidat untuk Administrator

Pada Gambar 3.10 menunjukan diagram aktivitas olah data kandidat untuk administrator. Sebelum melakukan pengolahan data kandidat, administrator harus melakukan autentikasi dan otorisasi. Setelah melakukan autentikasi dan otorisasi, administrator dapat melihat semua data kandidat yang ada pada sistem. Selain itu administrator juga dapat melakukan menambah kandidat dan menghapus kandidat.



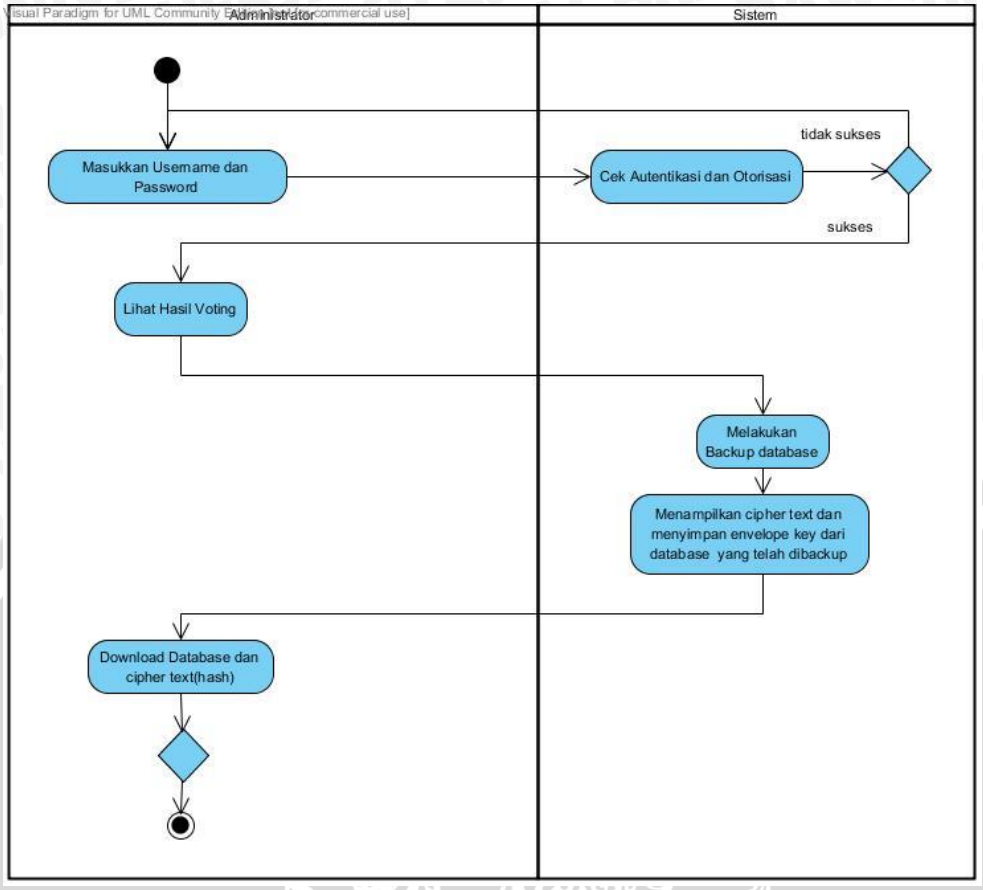
Gambar 3. 11 Diagram Aktivitas Lihat Hasil *Voting*

Pada Gambar 3.11 menunjukan diagram aktivitas lihat hasil *voting* untuk administrator. Sebelum melihat hasil voting, administrator harus melakukan autentikasi dan otorisasi. Setelah melakukan autentikasi dan otorisasi, administrator dapat melihat semua data hasil *voting*. Selain itu administrator juga dapat melakukan *backup database*.



Gambar 3. 12 Diagram Aktivitas Lihat Message Digest dan Simpan *Envelope Key*

Pada Gambar 3.12 menunjukan diagram aktivitas lihat *message digest* dan simpan *envelope-key* untuk administrator. Administrator dapat melihat hasil hash atau disebut sebagai *message digest* dari database yang telah di*backup*. Hasil *hash* ini nanti akan disimpan oleh administrator untuk dicocokkan dengan hasil *hash* yang telah di dekripsi untuk mengecek ada tidaknya perubahan pada *database* yang dikirim.



Gambar 3. 13 Diagram Aktivitas *Backup Database*

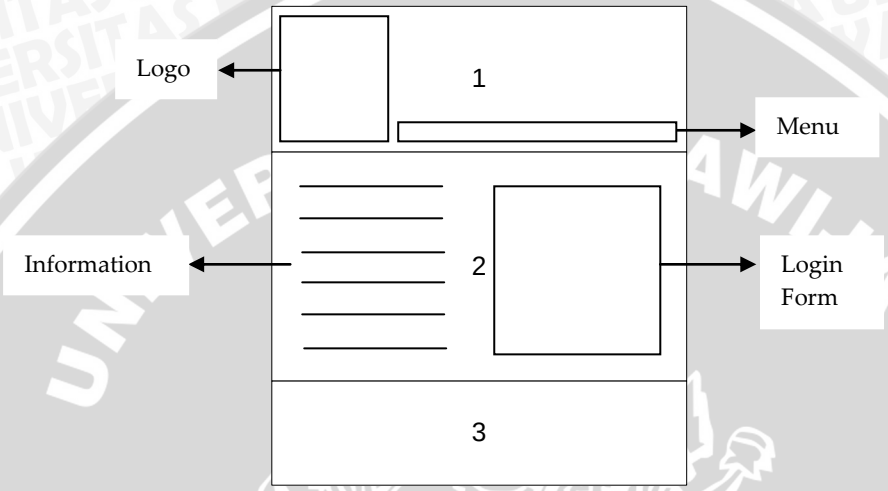
Pada Gambar 3.13 menunjukan diagram aktivitas *backup database*. Administrator dapat melakukan *backup database* hasil voting. Administrator hanya akan menekan tombol dan kemudian sistem akan melakukan proses backup secara otomatis dan kemudian menampilkan *message digest* dan menyimpan *envelope key* dari database yang telah di*backup.*

3. Perancangan Antarmuka

Pada bagian ini akan dijelaskan tentang perancangan antarmuka sistem sistem e-voting dengan menerapkan *hash* dan *digital signature* untuk verifikasi data hasil voting. Antarmuka perangkat lunak ini akan digunakan oleh pengguna untuk berinteraksi dengan sistem ini.

- **Perancangan Antarmuka Halaman Login**

Halaman *log in* merupakan antarmuka pengguna untuk sistem *e-voting* yang berfungsi sebagai akses masuk *user* administrator dan *voter* untuk menggunakan masing-masing hak akses yang telah ditentukan. Gambar 3.7 akan menunjukkan perancangan tampilan antarmuka dari halaman *log in*.



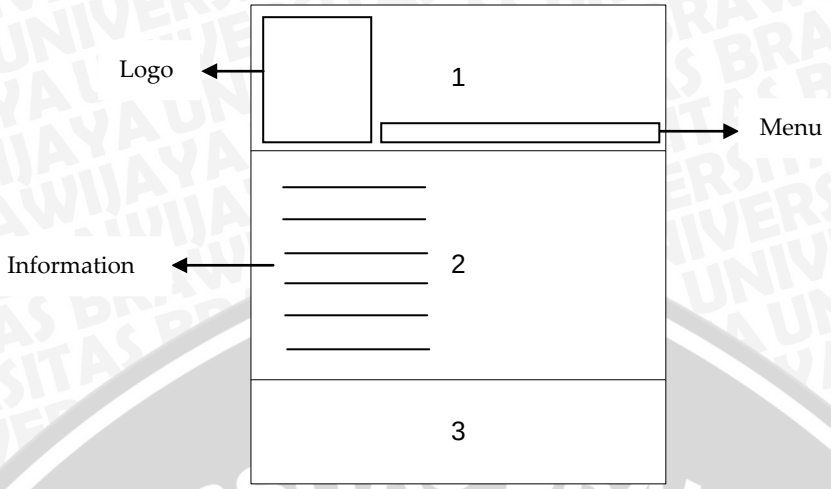
Gambar 3. 14 Tampilan antarmuka halaman *log in*

Halaman login menampilkan sekilas mengenai sistem e-voting. Gambar 3.14 memiliki keterangan sebagai berikut :

1. *Header* berisi identitas dan logo sistem e-voting serta menu untuk registrasi pengguna
2. Konten. Dapat berupa pembukaan maupun sambutan yang diperlukan untuk proses voting dan juga form untuk proses *login*.
3. *Footer*

- **Perancangan Antarmuka Halaman Utama**

Halaman utama merupakan salah satu antarmuka pengguna untuk sistem *e-voting* yang berfungsi sebagai halaman yang menunjukkan berhasilnya *user* melakukan proses *log in*. Gambar 3.15 akan menunjukkan perancangan tampilan antarmuka dari halaman utama untuk *user administrator* dan dosen.



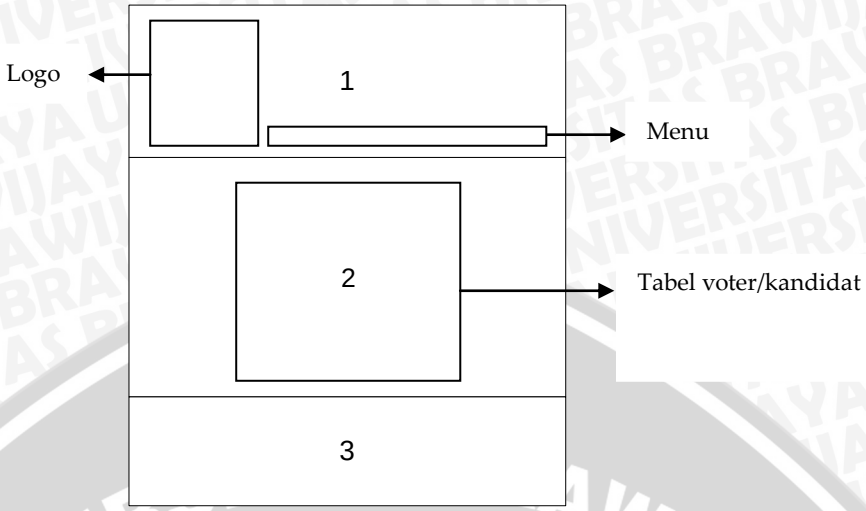
Gambar 3. 15 Tampilan antarmuka halaman utama *user* administrator dan *voter*

Halaman utama menampilkan sekilas mengenai sistem e-voting. Gambar 3.10 memiliki keterangan sebagai berikut :

1. *Header* berisi identitas dan logo sistem e-voting serta menu untuk registrasi pengguna
2. Konten. Dapat berupa pembukaan maupun sambutan yang diperlukan untuk proses voting.
3. *Footer*

- **Perancangan Antarmuka Halaman Daftar Data *Voter*/Kandidat**

Gambar 3.16 akan menunjukkan perancangan tampilan antarmuka dari halaman daftar data *voter*/kandidat. Halaman daftar data menampilkan semua data voter maupun kandidat. Untuk voter yang ditampilkan berupa id voter, nama voter, e-mail voter, keterangan status vote, serta pilihan untuk menghapus voter dengan syarat belum melakukan *voting*. Sedangkan untuk kandidat yang ditampilkan adalah foto kandidat, id kandidat, nama kandidat, dan pilihan untuk menghapus kandidat.



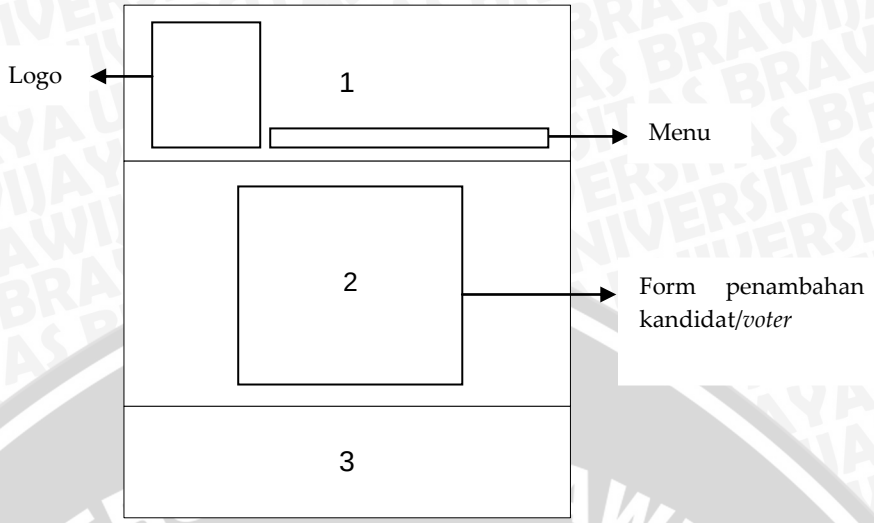
Gambar 3. 16 Tampilan Tampilan antarmuka halaman daftar data *voter*/kandidat *user* administrator

Gambar 3.16 memiliki keterangan sebagai berikut :

1. *Header* berisi identitas dan gambar tentang sistem e-vote dan menu.
2. Daftar data *voter*/kandidat. Di bagian bawah terdapat tombol untuk melakukan penambahan voter/kandidat.
3. *Footer*

- **Perancangan Antarmuka Halaman Penambahan *Voter*/Kandidat**

Halaman penambahan *voter*/kandidat merupakan salah satu antarmuka sistem *e-voting* yang berfungsi menampilkan *form input* data *voter*/kandidat yang akan ditambahkan. Gambar 3.17 akan menunjukkan perancangan tampilan antarmuka dari halaman penambahan *voter*/kandidat.



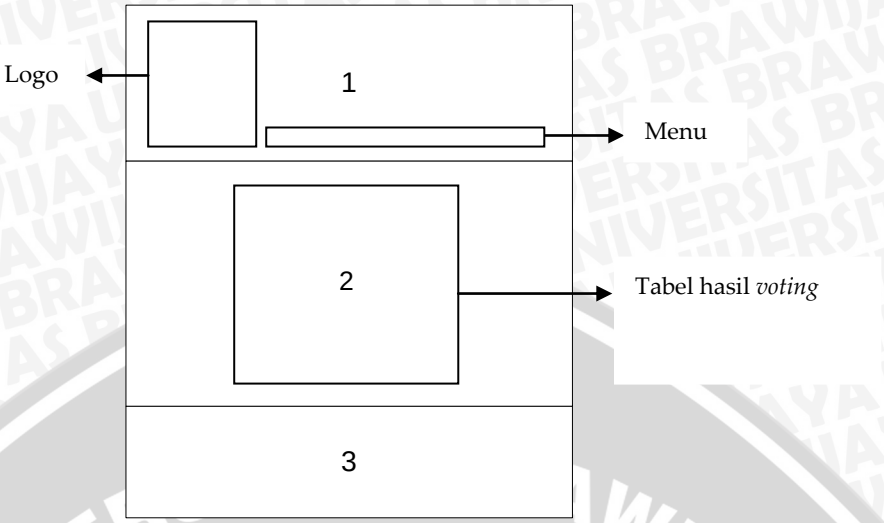
Gambar 3. 17 Tampilan antarmuka halaman penambahan *voter/kandidat*

Gambar 3.17 memiliki keterangan sebagai berikut :

1. *Header* berisi identitas dan gambar tentang sistem e-vote dan menu.
2. Konten yang menampilkan *form input* data registrasi user
3. *Footer*

- **Perancangan Antarmuka Halaman *Result***

Halaman *result* adalah halaman yang menampilkan hasil *voting*. Pada halaman ini terdapat keterangan kandidat beserta jumlah suara yang telah diperoleh. Di bagian bawah terdapat tombol untuk melakukan *backup database.* Gambar 3.18 akan menunjukkan perancangan tampilan antarmuka dari halaman *result.*



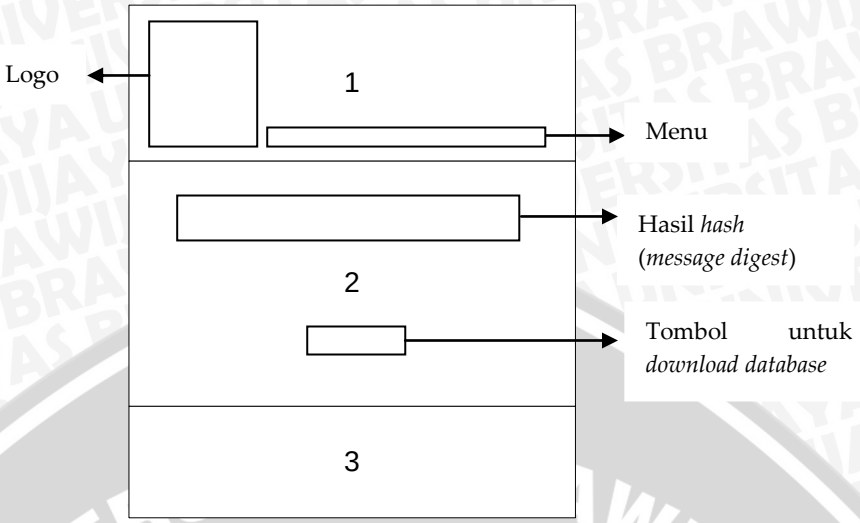
Gambar 3. 18 Tampilan antarmuka halaman *result*

Gambar 3.18 memiliki keterangan sebagai berikut :

1. *Header* berisi identitas dan gambar tentang sistem e-vote dan menu
2. Konten yang menampilkan data kandidat dan suara yang telah diperoleh. Di bagian bawah terdapat tombol untuk melakukan *backup database*.
3. *Footer*

- **Perancangan Antarmuka Halaman *Backup***

Halaman *backup* merupakan salah satu antarmuka pengguna untuk sistem *e-voting* yang berfungsi menampilkan hasil *hash* (*message digest*) dari *database* yang telah di *backup*. Gambar 3.19 akan menunjukkan perancangan tampilan antarmuka dari halaman *backup*.



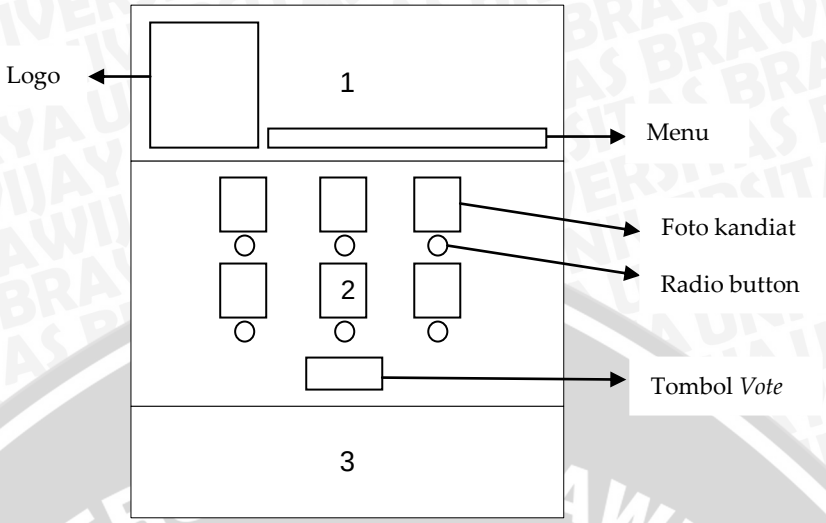
Gambar 3. 19 Tampilan antarmuka halaman *backup*

Gambar 3.19 memiliki keterangan sebagai berikut :

1. *Header* berisi identitas dan gambar tentang sistem e-vote dan menu
2. Konten yang menampilkan hasil hash dari database yang telah di *backup* dan juga menyimpan *envelope key* yang digunakan untuk membuka enkripsi. Pada bagian bawahnya terdapat tombol untuk melakukan *download hash* dan file database yang telah dienkripsi serta file *envelope-key* dan diberi *digital signature*.
3. *Footer*

- **Perancangan Antarmuka Halaman *Voting***

Gambar 3.20 akan menunjukkan perancangan tampilan antarmuka dari halaman managemen *voting*. Halaman *voting* adalah halaman yang menampilkan daftar kandidat untuk dipilih. Proses pemilihan dengan memilih kandidat kemudian menekan tombol *vote* pada akhir bagian konten.



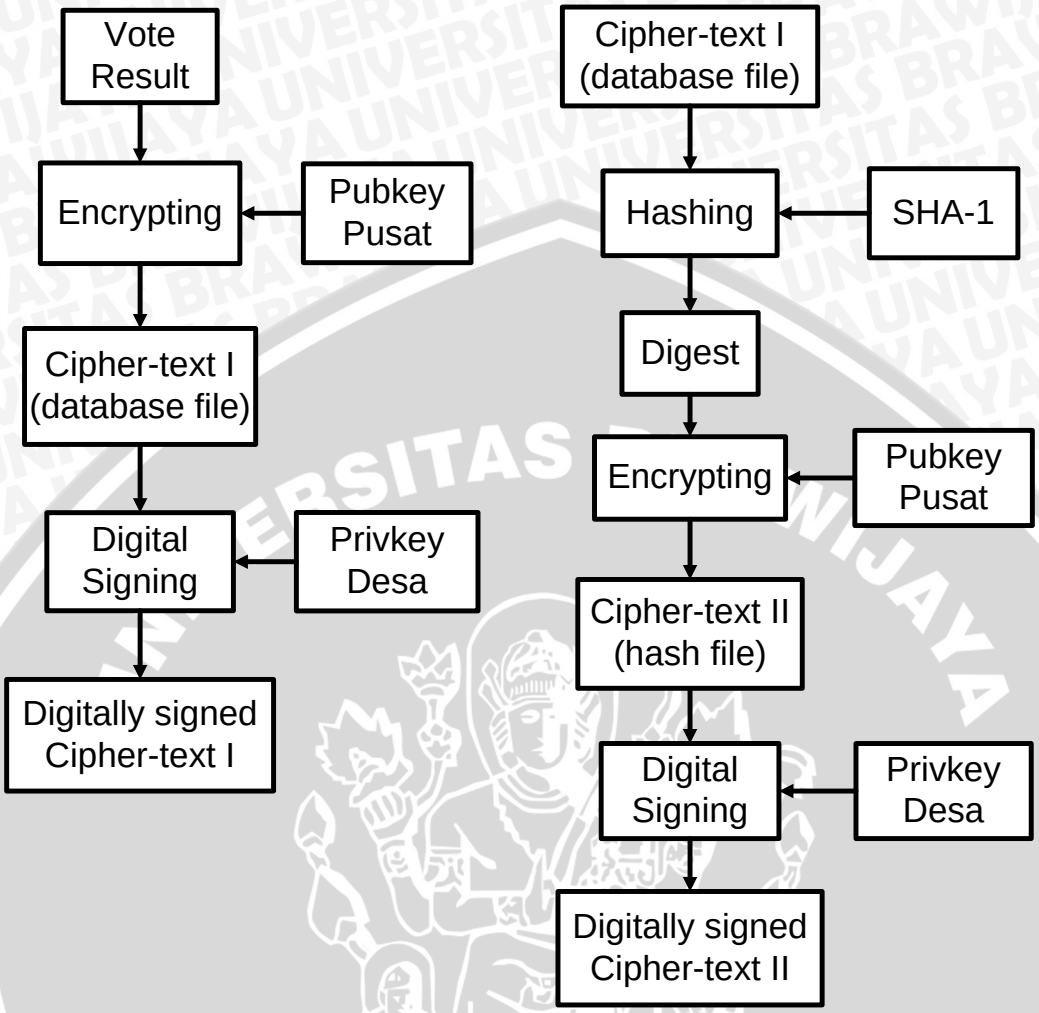
Gambar 3. 20 Tampilan antarmuka halaman *voting*

Gambar 3.20 memiliki keterangan sebagai berikut :

1. *Header* berisi identitas dan gambar tentang sistem e-vote dan menu
2. Konten yang menampilkan daftar kandidat yang dapat dipilih dengan menggunakan radio kemudian menekan tombol vote yang ada pada akhir bagian konten.
3. *Footer*

4. Perancangan *Hashing* dan *Digital Signing*

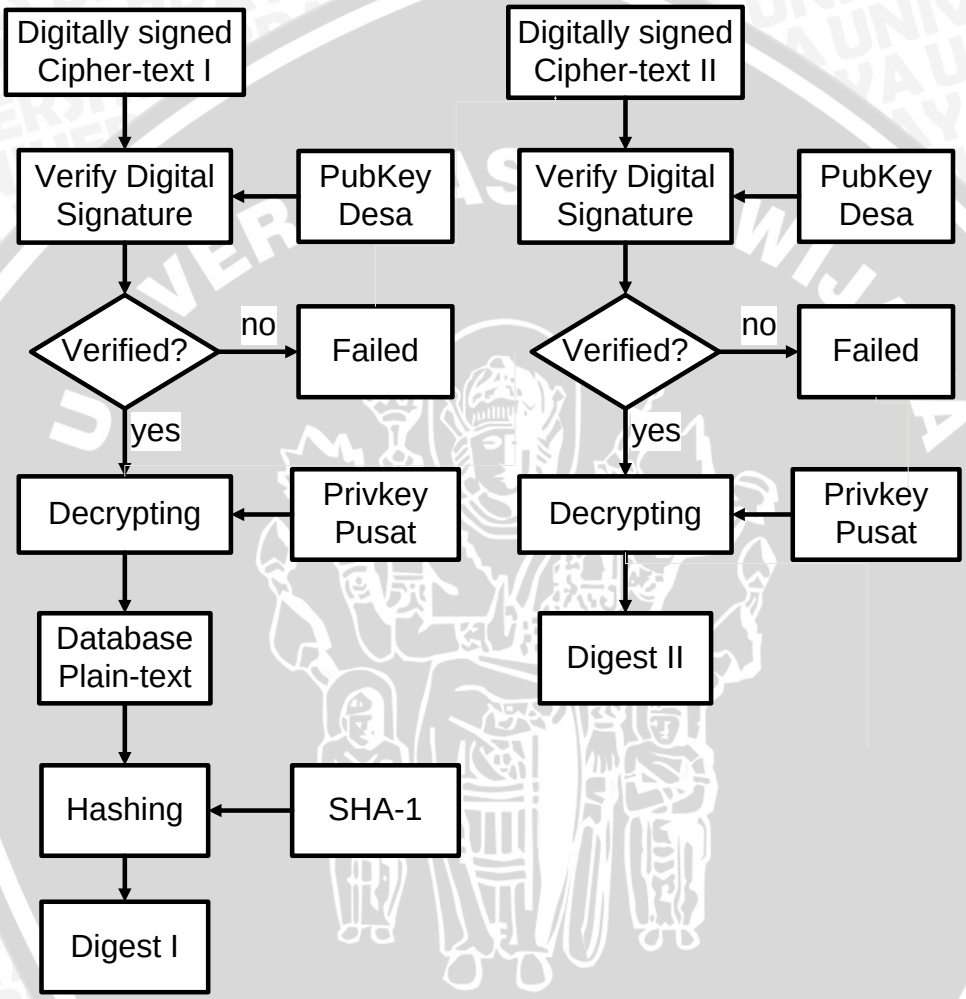
Pada bagian ini akan dijelaskan tentang perancangan proses *hashing* dan *digital signing* pada sistem perangkat lunak. Tujuan dari perancangan ini adalah untuk membuat proses *hashing* dan *digital signing* se-efisien mungkin. Pada diagram 3.21 berikut ini akan menjelaskan bagaimana proses *hashing* dan *digital signing* dibuat.



Gambar 3. 21 Diagram Perancangan Proses *Hashing* dan *Digital Signing*

Pada awal proses ini, hasil *voting* sudah didapatkan dari proses *voting* yang dilakukan oleh *voter*. Pada proses pertama, hasil *voting* dienkripsi menggunakan *public key* yang diberikan oleh pusat yang kemudian menghasilkan *cipher-text I*. Kemudian hasil enkripsi ini diberikan *digital signature* dengan melakukan *digital signing* menggunakan *private key* dari desa yang menghasilkan file database terenkripsi yang memiliki *digital signature* (*Digitally Signed Cipher-text I*). Proses yang kedua adalah *cipher-text I* di-*hash* menggunakan SHA-1 yang menghasilkan *digest*. Kemudian *digest* tersebut dienkripsi lagi menggunakan *public key* pusat dan menghasilkan *cipher-text II*. Hasil enkripsi tersebut kemudian diberikan digital signature menggunakan private key desa yang menghasilkan file hash yang terenkripsi (*Digitally Signed Cipher-text II*). Dua file

yang terlah diberi *digital signature* inilah yang nantinya dikirimkan ke pusat untuk dilakukan verifikasi dan kemudian di dekripsi menggunakan private key yang dimiliki oleh pusat. Pada gambar 3.22 dijelaskan bagaimana proses verifikasi dan dekripsi dilakukan oleh pusat.



Gambar 3. 22 Diagram Perancangan Proses Verifikasi Pengirim dan Dekripsi

Proses verifikasi dan dekripsi berlangsung di pusat karena hasil *voting* telah diterima oleh pusat. Dua file utama yaitu file database dan file hash dilakukan proses verifikasi digital signature untuk memverifikasi pengirim file. Apabila terverifikasi maka dapat dilakukan proses dekripsi, sedangkan bila file tersebut tidak terverifikasi maka proses dekripsi gagal dilakukan. File yang tidak terverifikasi mengindikasikan bahwa file yang diterima oleh pusat adalah file

yang berasal dari sumber yang salah. Proses dekripsi dari file hash adalah plain-text yang berupa digest. Sedangkan proses dekripsi dari file database akan menghasilkan plain-text berupa file dump database yang akan diimport ke dalam database pusat. Untuk mendeteksi perubahan pada data, hasil dump database dilakukan hash yang menghasilkan digest. Digest ini kemudian dicocokkan dengan digest hasil dekripsi dari file hash. Jika cocok maka file tidak mengalami perubahan pada saat proses pengiriman.

BAB IV
IMPLEMENTASI

Bab ini membahas mengenai tahapan implementasi sistem perangkat lunak *e-voting* yang menerapkan *hash* dan *digital signature* untuk verifikasi data hasil voting berdasarkan hasil yang telah didapatkan dari analisis kebutuhan dan proses perancangan perangkat lunak. Implementasi terdiri atas penjelasan tentang spesifikasi sistem, batasan-batasan dalam implementasi, implementasi basis data, implementasi algoritma, dan implementasi antarmuka.

4.1 Spesifikasi Lingkungan Sistem

Sistem *e-voting* yang menerapkan *hash* dan *digital signature* untuk verifikasi data hasil voting dikembangkan dalam lingkungan implementasi yang terdiri dari perangkat keras dan perangkat lunak.

4.1.1 Spesifikasi Lingkungan Perangkat Keras

Spesifikasi lingkungan perangkat keras yang dipakai dalam proses pengembangan sistem *e-voting* yang menerapkan *hash* dan *digital signature* untuk verifikasi data hasil voting dijelaskan pada Tabel 4.1.

Tabel 4. 1 Spesifikasi lingkungan perangkat keras komputer

Notebook Compaq 510	
Processor	Intel (R) Pentium(R) Core 2 Duo T5870 (2.0 GHz, 2 MB L2 cache, 800 MHz FSB)
Memory (RAM)	2 GB
Harddisk	250 GB HDD

4.1.2 Spesifikasi Lingkungan Perangkat Lunak

Spesifikas lingkungan perangkat lunak yang dipakai dalam proses pengembangan sistem *e-voting* yang menerapkan *hash* dan *digital signature* untuk verifikasi data hasil voting dijelaskan pada Tabel 4.2.

Tabel 4. 2 Spesifikasi lingkungan perangkat lunak computer

Notebook Acer Aspire 2930Z	
Operating System	Microsoft Windows 7 Home Basic 32-bit
Programming Language	*HyperText Markup Language (HTML), Hypertext Preprocessor (PHP), Javascript*
Software Development Kit	*Google Chrome versi 30.0.1599.101 m*
Basis data *Management System*	MySQL 5.1
Integrated Development Environment	*Adobe Dreamweaver* CS5, *SQLyog Ultimate – MySQL GUI v11.11 (32 bit)*

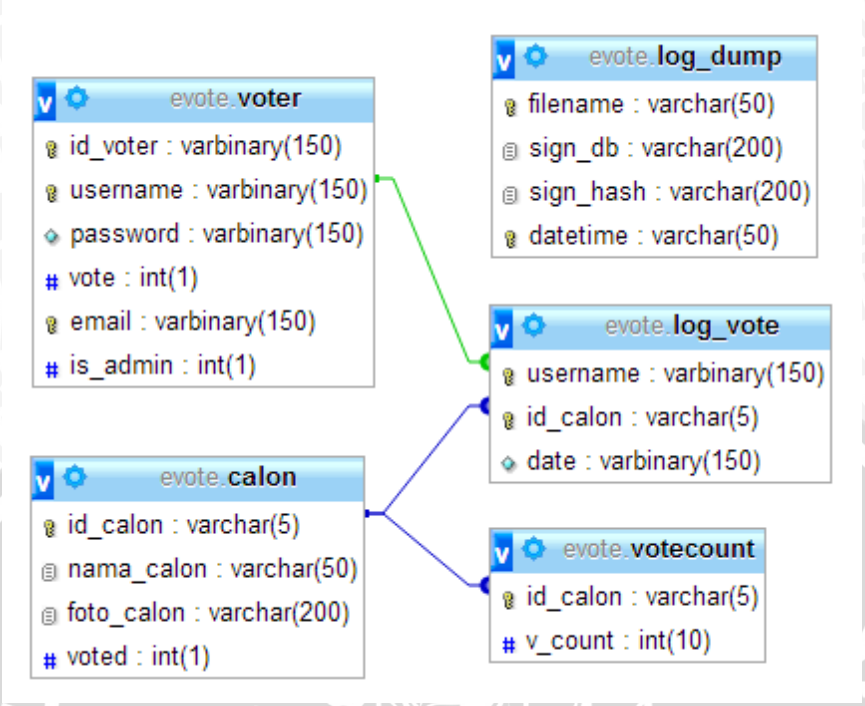
4.2 Batasan – Batasan Implementasi

Beberapa batasan dalam mengimplementasikan sistem *e-voting* yang menerapkan *hash* dan *digital signature* untuk verifikasi data hasil voting adalah sebagai berikut:

1. Pembuatan aplikasi sistem e-vote ini dikerjakan dengan bahasa pemrograman PHP dengan *Framework CI* dan basis data MySQL.
2. Aplikasi hanya dilakukan pengujian pada *PC browser*.
3. Algoritma hash yang digunakan adalah *Secure Hash Algorithm-1*.
4. Digital signature dibuat menggunakan algoritma RSA.
5. Beberapa sampel tempat makan dan menu makanan di Universitas Brawijaya yang telah disortir dalam aplikasi ini.
6. Gambar untuk kandidat yang dapat dimasukkan hanya format .png, .jpeg, dan .gif .

4.3 Implementasi Basis Data

Implementasi penyimpanan data dilakukan dengan basis data *management system* MySQL. Hasil implementasi penyimpanan data ini berupa *script* SQL. Hasil implementasi SQL pada basis data ini dimodelkan dalam diagram konseptual *entity relationship.* Gambar 4.1 menggambarkan diagram konseptual *entitiy relationship* dari sistem *e-voting* yang menerapkan *hash* dan *digital signature* untuk verifikasi data hasil voting.



Gambar 4. 1 Diagram ER konseptual dari sistem

4.4 *Hashing* dan *Digital Signing*

Sistem *e-voting* yang menerapkan *hash* dan *digital signature* untuk verifikasi data hasil *voting* fitur utama yaitu *backup database* dan *digital signing*. Implementasi fitur-fitur ini akan direpresentasikan dalam bentuk *code* dengan bahasa pemrograman PHP pada framework CI.

4.4.1 Implementasi *Backup Database*

Operasi *backup database* bertujuan untuk melakukan dump pada database dan menyimpannya ke dalam sebuah file yang nantinya dapat di download oleh administrator. Berikut ini adalah sekilas kode untuk melakukan *backup database*.

```
1.  ...
2.          $link = mysql_connect($host,$user,$pass);
3.          mysql_select_db($name,$link);
4.
5.
6.          if($tables == '*')
7.          {
```

```
8.              $tables = array();
9.              $result = mysql_query('SHOW TABLES');
10.             while($row = mysql_fetch_row($result))
11.             {
12.                     $tables[] = $row[0];
13.             }
14.     }
15.     else
16.     {
17.             $tables    =    is_array($tables)    ?    $tables    :
18. explode(',',$tables);
19.     }
20.
21.     $return='';
22.
23.     foreach($tables as $table)
24.     {
25.             $result = mysql_query('SELECT * FROM '.$table);
26.             $num_fields = mysql_num_fields($result);
27.
28.             //$return.= 'DROP TABLE '.$table.';';
29.             //$row2  =  mysql_fetch_row(mysql_query('SHOW  CREATE
30. TABLE '.$table));
31.             //$return.= "\n\n".$row2[1].";\n\n";
32.
33.             for ($i = 0; $i < $num_fields; $i++)
34.             {
35.                     while($row = mysql_fetch_row($result))
36.                     {
37.                             $return.=   'INSERT   INTO   '.$table.'
38. VALUES(';
39.                             for($j=0; $j<$num_fields; $j++)
40.                             {
41.                                     $row[$j] = addslashes($row[$j]);
42.                                     $row[$j]                          =
43. str_replace("\n","\\n",$row[$j]);
44.                                     if  (isset($row[$j]))  {  $return.=
45. '"'.$row[$j].'"' ; } else { $return.= '""'; }
46.                                     if      ($j<($num_fields-1))      {
47. $return.= ','; }
48.                             }
49.                             $return.= ");\n";
```

```
50.                         }
51.                     }
52.                     $return.="\n\n\n";
53.             }
54.
55.     ...
56.
```

Gambar 4. 2 Implementasi *Backup Database*

Penjelasan implementasi *bakcup database* dari Gambar 4.2 yaitu:

1. Baris 2-3 digunakan untuk melakukan koneksi pada database.
2. Baris 6-21 digunakan untuk mendapatkan tabel-tabel yang akan di *dump* dalam database dan disimpan dalam variabel.
3. Baris 23-56 merupakan perulangan yang dikerjakan pada tiap tabel. Perulangan ini befungsi untuk melakukan backup meliputi drop table, create table, dan proses inserting.

4.4.2 Implementasi Enkripsi

```
1.      $fp=fopen ("public.pem","r");
2.      $pub_key=fread ($fp,8192);
3.      fclose($fp);
4.      $PK="";
5.      $PK=openssl_get_publickey($pub_key);
6.      if (!$PK) {
7.          echo "Cannot get public key";
8.      }
9.
10.     $encrypted = '';
11.     $a_envelope = array();
12.     $a_key = array($PK);
13.     if (openssl_seal($return, $encrypted, $a_envelope, $a_key)
14.     === FALSE)
15.         die('Failed to encrypt data');
16.
17.     $hash_encrypted = '';
18.     if (openssl_public_encrypt($hash, $hash_encrypted, $pub_key)
19.     === FALSE)
20.         die('Failed to encrypt data');
21.
22.     $sql_name = 'db-backup.sql';
```

```
23.         $handle = fopen($sql_name,'w+');
24.         fwrite($handle,$encrypted);
25.         fclose($handle);
26.
27.         $handle1 = fopen('hash-db.txt','w+');
28.         fwrite($handle1,$hash_encrypted);
29.         fclose($handle1);
30.
31.         openssl_free_key($PK);
```

Gambar 4. 3 Implementasi *Enkripsi*

Penjelasan implementasi enkripsi dari Gambar 4.3 yaitu:

1. Baris 1-8 berfungsi pemanggilan kunci publik RSA untuk melakukan enkripsi.
2. Baris 13-15 berfungsi untuk *sealing* dokumen menggunakan kunci publik RSA.
3. Baris 17-20 berfungsi untuk mengenkripsi hash dari hasil *backup.*
4. Baris 22-29 berfungsi untuk menyimpan hasil enkripsi database maupun hash ke dalam file.

4.4.3 Implementasi Digital Signing

```
1.          $fp = fopen("digisign.pem", "r");
2.          $priv_key = fread($fp, 8192);
3.          fclose($fp);
4.          $pkeyid = openssl_get_privatekey($priv_key);
5.
6.          openssl_sign("db-backup.sql",     $signature_db,     $pkeyid,
7.  OPENSSL_ALGO_SHA1);
8.          openssl_sign("hash-db.txt",     $signature_hash,     $pkeyid,
9.  OPENSSL_ALGO_SHA1);
10.
11.         openssl_free_key($pkeyid);
```

Gambar 4. 4 Implementasi *Digital Signing*

Penjelasan implementasi ganti gambar tempat makan pada Gambar 4.4 yaitu :

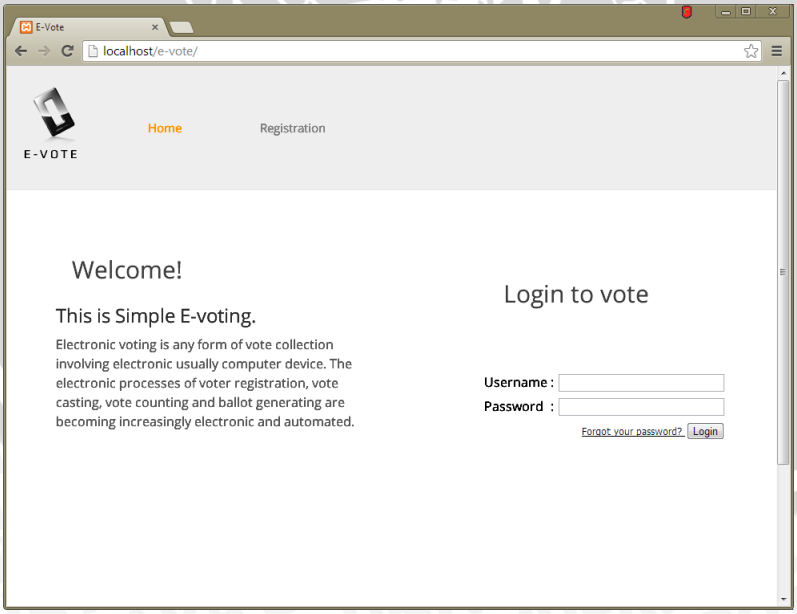
1. Baris 1-4 berfungsi untuk melakukan pemanggilan kunci publik untuk digital signature.
2. Baris 6-7 berfungsi untuk melakukan digital signing pada *file database*.
3. Baris 8-9 berfungsi untuk melakukan digital signing pada *file hash*.

4.5 Antarmuka

Implementasi antarmuka sistem *e-voting* yang menerapkan *hash* dan *digital signature* untuk verifikasi data hasil voting digunakan oleh pengguna untuk berinteraksi dengan sistem perangkat lunak.

4.5.1 Implementasi Antarmuka Halaman Login

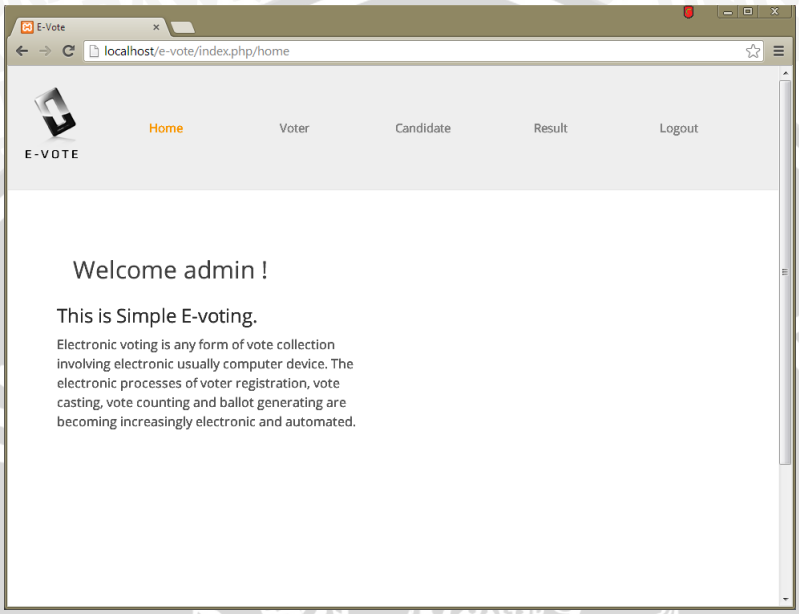
Halaman login merupakan halaman pertama yang akan dibuka ketika sistem dijalankan. Dari halaman login pengguna mendapatkan informasi bagaimana melakukan registrasi maupun pengumuman penting. Gambar 4.5 menunjukkan implementasi tampilan antarmuka dari halaman login yang mengacu pada perancangan antarmuka halaman login.



Gambar 4. 5 Implementasi antarmuka halaman log in

4.5.2 Implementasi Antarmuka Halaman Utama

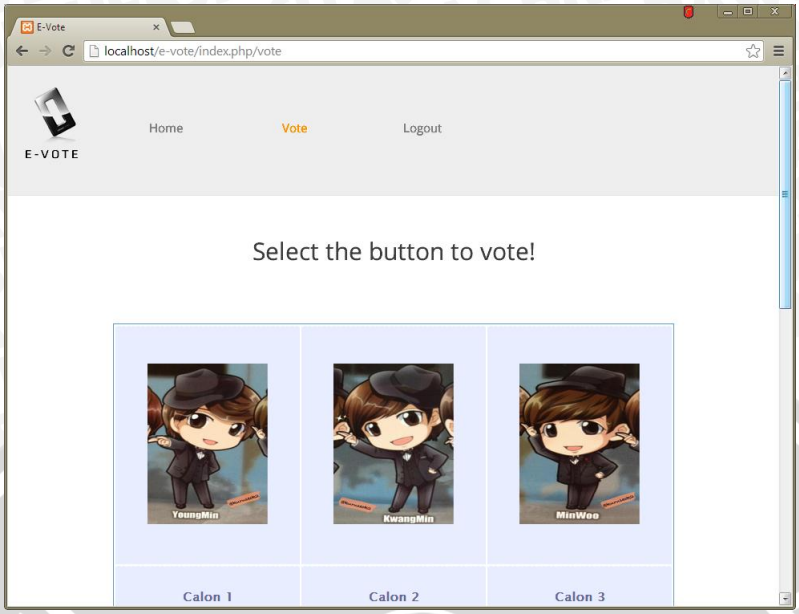
Halaman utama merupakan halaman pertama yang akan dibuka ketika proses login sukses. Dari halaman beranda pengguna mendapatkan pengumuman penting. Gambar 4.6 menunjukkan implementasi tampilan antarmuka dari halaman utama yang mengacu pada perancangan antarmuka halaman utama.



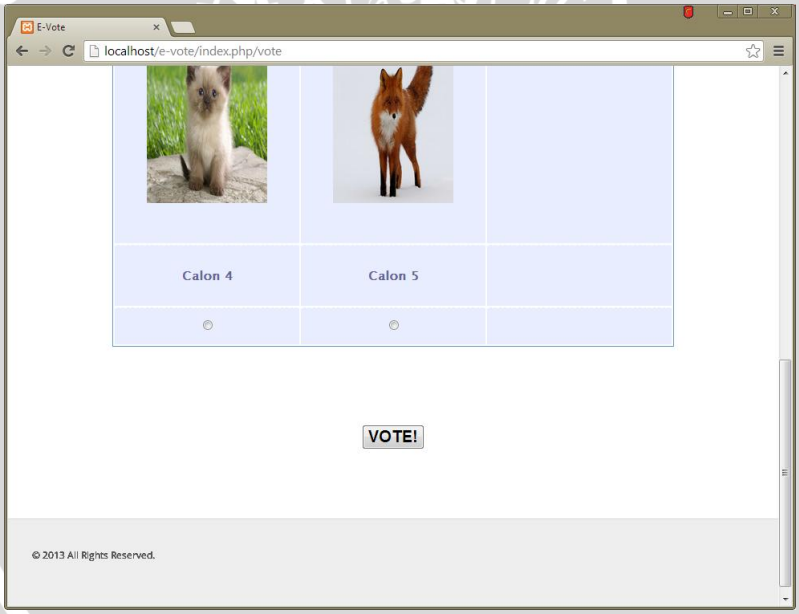
Gambar 4. 6 Implementasi antarmuka halaman utama

4.5.3 Implementasi Antarmuka Halaman *Vote*

Halaman *vote* merupakan halaman yang digunakan untuk melakukan proses *voting*. Pada halaman ini terdapat foto kandidat dan tombol *vote* untuk melakukan *voting*. Gambar 4.7 dan gambar 4.8 menunjukkan implementasi tampilan antarmuka dari halaman *vote* yang mengacu pada perancangan antarmuka halaman *vote*.



Gambar 4. 7 Implementasi antarmuka halaman *vote*(1)

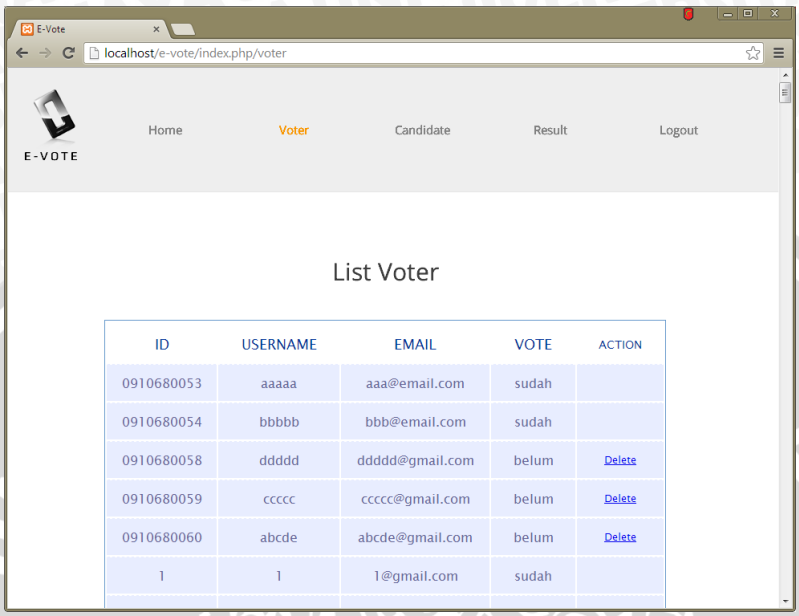


Gambar 4. 8 Implementasi antarmuka halaman *vote*(2)

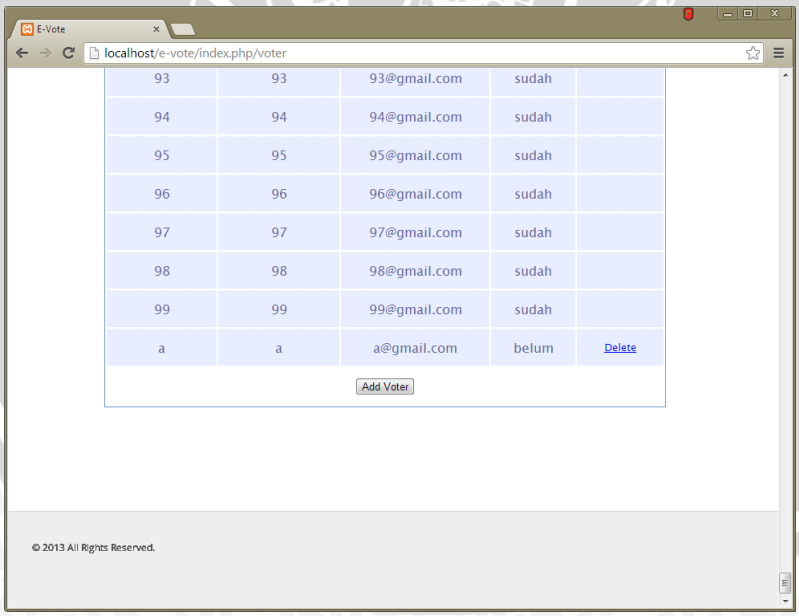
4.5.4 Implementasi Antarmuka Halaman Daftar Data Voter

Halaman daftar data *voter* merupakan halaman yang menampilkan daftar *voter* yang terlah terdaftar beserta data-datanya meliputi ID, *username*, *email*, dan status *vote*. Gambar 4.9 dan 4.10 menunjukkan implementasi tampilan antarmuka

dari halaman daftar data *voter* yang mengacu pada perancangan antarmuka halaman daftar data *voter*.



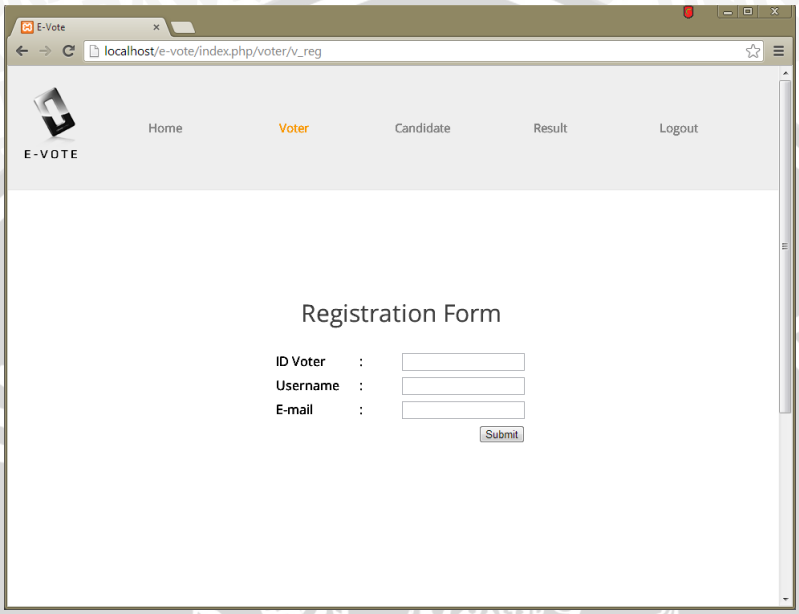
Gambar 4. 9 Implementasi antarmuka halaman *voter*(1)



Gambar 4. 10 Implementasi antarmuka halaman *voter*(2)

4.5.5 Implementasi Antarmuka Halaman Penambahan Voter

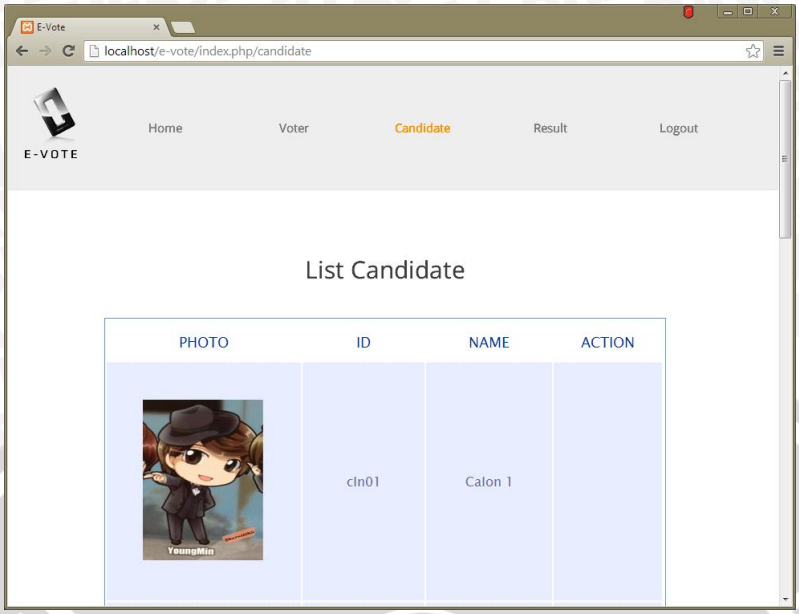
Halaman penambahan *voter* berfungsi menambahkan data *voter*. Gambar 4.11 menunjukkan implementasi tampilan antarmuka dari tambah data *voter* yang mengacu pada perancangan antarmuka halaman penambahan *voter*.



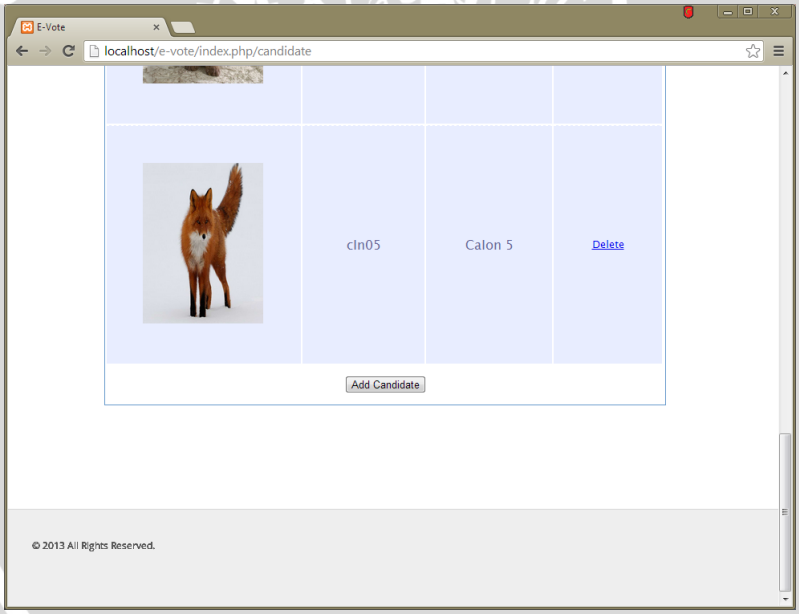
Gambar 4. 11 Implementasi antarmuka halaman penambahan *voter*

4.5.6 Implementasi Antarmuka Halaman Daftar Data Kandidat

Halaman daftar data kandidat merupakan halaman yang menampilkan daftar kandidat yang telah terregistrasi beserta dengan datanya meliputi foto kandidat, id kandidat, nama kandidat, dan pilihan untuk menghapus kandidat yang belum pernah di-*vote*. Gambar 4.12 dan 4.13 menunjukkan implementasi tampilan antarmuka dari daftar data kandidat yang mengacu pada perancangan antarmuka halaman daftar data kandidat.



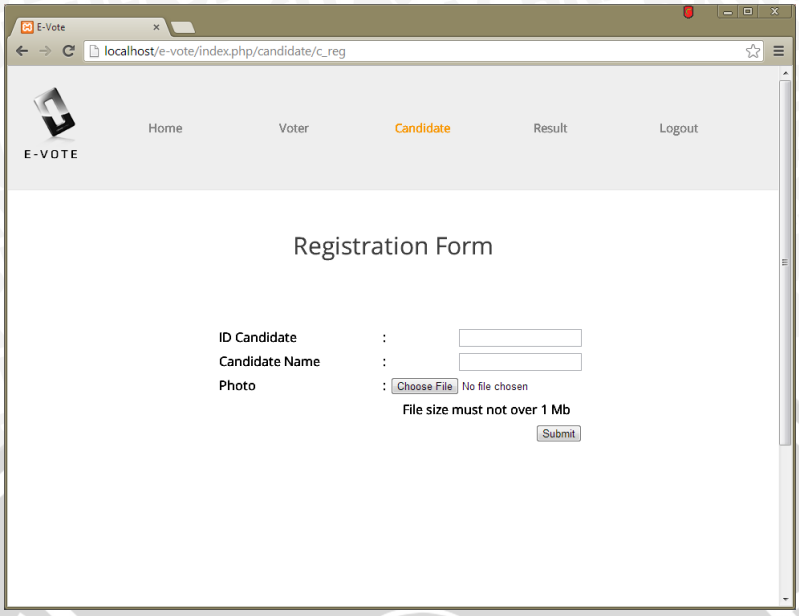
Gambar 4. 12 Implementasi antarmuka halaman daftar data kandidat(1)



Gambar 4. 13 Implementasi antarmuka halaman daftar data kandidat(2)

4.5.7 Implementasi Antarmuka Halaman Penambahan Kandidat

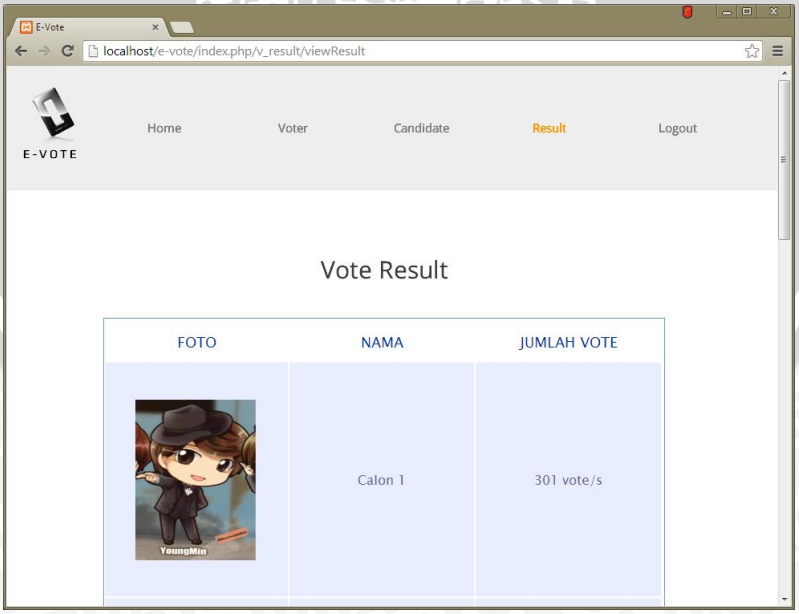
Halaman penambahan kandidat berfungsi menambahkan data kandidat. Gambar 4.14 menunjukkan implementasi tampilan antarmuka dari penambahan data kandidat yang mengacu pada perancangan antarmuka halaman penambahan kandidat.



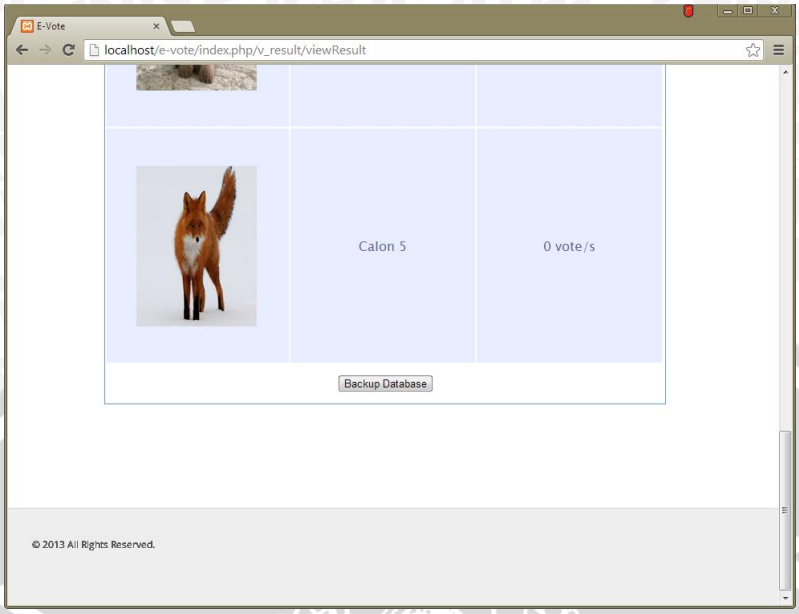
Gambar 4. 14 Implementasi antarmuka halaman penambahan kandidat

4.5.8 Implementasi Antarmuka Halaman *Result*

Halaman result merupakan halaman yang menampilkan daftar kandidat serta hasil voting yang diperoleh oleh tiap-tiap kandidat. Pada halaman ini juga ditampilkan tombol untuk melakukan *backup database*. Gambar 4.15 menunjukkan implementasi tampilan antarmuka dari halaman result yang mengacu pada perancangan antarmuka halaman result.



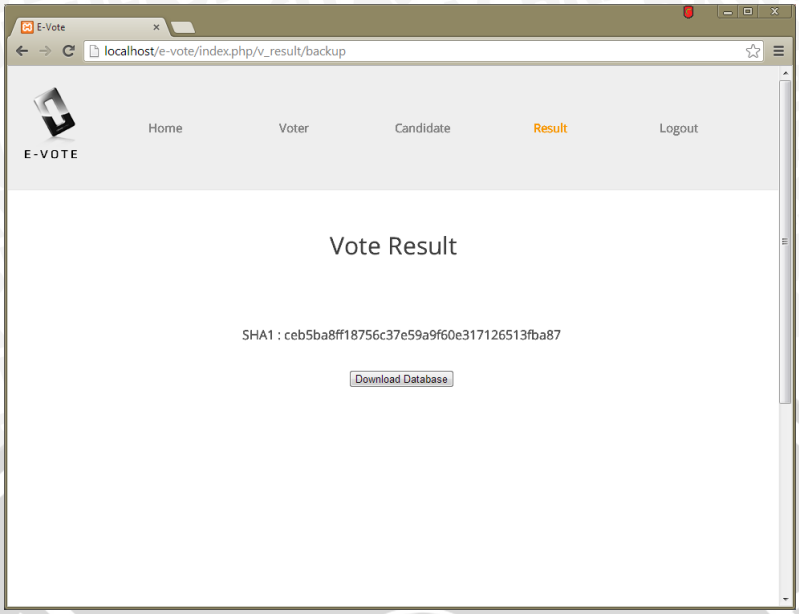
Gambar 4. 15 Implementasi antarmuka halaman *result*(1)



Gambar 4. 16 Implementasi antarmuka halaman *result*(2)

4.5.9 Implementasi Antarmuka Halaman *Backup*

Pada bagian akhir konten halaman result terdapat tombol yang akan melakukan proses backup pada database dan masuk ke halaman backup. Halaman backup berfungsi untuk melihat hash dari hasil voting yang telah di backup serta menyimpan *envelope key* yang nantinya digunakan untuk membuka seal enkripsi file database. Gambar 4.17 menunjukkan implementasi tampilan antarmuka dari halaman backup yang mengacu pada perancangan antarmuka halaman backup.



Gambar 4. 17 Implementasi antarmuka halaman *backup*

BAB V
PENGUJIAN DAN ANALISIS

Bab ini membahas mengenai tahapan pengujian dan pembahasan sistem perangkat lunak untuk sistem *e-voting* yang menerapkan *hash* dan *digital signature* untuk verifikasi data hasil voting yang telah dikembangkan melalui pengujian validasi, pengujian verifikasi, dan pengujian performa. Pengujian validasi menguji apakah sistem berjalan dengan benar menggunakan sebuah set *test case* yang merefleksikan penggunaan yang diharapkan dari sistem. Pengujian performa dilakukan untuk mengukur kecepatan jalannya sistem ketika diakses. Proses pembahasan bertujuan untuk mendapatkan kesimpulan dari hasil pengujian sistem *e-voting* yang menerapkan *hash* dan *digital signature* untuk verifikasi data hasil voting yang telah dilakukan.proses pembahasan mengacu pada dasar teori sesuai dengan hasil pengujian yang didapatkan. Proses pembahasan yang dilakukan meliputi pembahasan hasil pengujian validasi, pengujian verifikasi, dan pengujian performa.

5.1 Pengujian Validasi Berdasarkan Diagram *Use Case*

Pengujian validasi dilakukan untuk mengetahui apakah sistem yang dibangun sudah benar sesuai dengan yang dibutuhkan. *Item-item* yang telah dirumuskan dalam daftar kebutuhan merupakan hasil analisis kebutuhan yang akan menjadi acuan untuk melakukan pengujian validasi. Pengujian validasi berfungsi untuk menemukan kesesuaian antara kerja sistem dengan daftar kebutuhan yang telah dirancang sebelumnya. Pada skripsi ini dilakukan pengujian validasi terhadap kinerja sistem *e-voting*.

5.1.1 Hasil Pengujian Validasi

1. Kasus Uji Validasi Olah Voter (SRS_001_01)

Kasus uji validasi olah data voter oleh administrator sebagai berikut.

Tabel 5. 1 Kasus uji validasi olah data *voter* (SRS_001_01)

Nama kasus uji	Olah Data *Voter*
Tujuan pengujian	Untuk menguji validitas kinerja dari sistem dalam menyediakan fasilitas lihat data *voter* bagi administrator, menghapus *voter*, dan menambah data voter serta melihat data *voter* baru yang ditambahkan.
Prosedur uji	1. Administrator telah melakukan *login* 2. Masuk ke halaman *voter* 3. Administrator menekan tautan *delete* untuk menghapus *voter*. 4. Administrator menekan tombol *add voter* untuk menambah *voter*, mengisi data user, dan menekan tombol submit. 5. Administrator melihat data yang baru saja dimasukkan ada dalam daftar voter
Hasil yang diharapkan	• Sistem dapat menampilkan data *voter* sesuai dengan *voter* terdaftar. • Sistem dapat melakukan seleksi *user* mana yang sudah dan yang belum melakukan *voting* kemudian menampilkan tautan d*elete* pada *user* yang belum melakukan *voting* dan dapat melakukan penghapusan pada *voter* yang telah dipilih. • Sistem dapat menyimpan data baru dari voter. Masuk ke halaman voter dan lihat daftar voter jika data telah terisi semua.
Hasil yang didapatkan	• Sistem menampilkan data *voter* sesuai dengan data *voter* terdaftar. • Sistem melakukan seleksi *user* mana yang sudah dan yang belum melakukan *voting*

	kemudian menampilkan tautan *delete* pada user yang belum melakuakan *voting* dan dapat melakukan penghapusan pada *voter* yang telah dipilih • Sistem menyimpan data baru dari voter. Masuk ke halaman voter dan lihat daftar voter.
Status Validitas	Valid

2. Kasus Uji Validasi Olah Data Kandidat (SRS_001_02)

Kasus uji validasi olah data kandidat oleh administrator sebagai berikut.

Tabel 5. 2 Kasus uji validasi olah data kandidat (SRS_001_02)

Nama kasus uji	Olah Data Kandidat
Tujuan pengujian	Untuk menguji validitas kinerja dari sistem dalam menyediakan fasilitas lihat data kandidat, menghapus data kandidat, dan menambah data kandidat serta melihat data kandidat yang telah ditambahkan oleh administrator.
Prosedur uji	1. Administrator melakukan *login*. 2. Administrator masuk ke halaman kandidat. 3. Administrator menekan tautan *delete* untuk menghapus kandidat. 4. Administrator menekan tombol *add candidate* untuk menambah kandidat kemudian mengisi data kandidat, dan menekan tombol submit 5. Administrator melihat data yang baru saja dimasukkan ada dalam daftar kandidat
Hasil yang diharapkan	• Sistem dapat menampilkan data kandidat yang sesuai dengan kandidat terdaftar. • Sistem dapat menyimpan data kandidat baru. Masuk ke halaman daftar kandidat jika semua

	data terisi. • Sistem dapat melakukan seleksi kandidat mana yang sudah dan yang belum dikenai *voting* kemudian menampilkan tautan d*elete* pada kandidat yang belum dikenai *voting* dan dapat pelakukan penghapusan pada kandidat yang dipilih.
Hasil yang didapatkan	• Sistem menampilkan data kandidat sesuai dengan kandidat terdaftar. • Sistem menyimpan data kandidat baru. Masuk ke halaman daftar kandidat jika semua data terisi. • Sistem melakukan seleksi kandidat mana yang sudah dan yang belum dikenai *voting* kemudian menampilkan tautan d*elete* pada kandidat yang belum dikenai *voting* dan melakukan penghapusan pada kandidat yang telah dipilih.
Status Validitas	Valid

3. Kasus Uji Validasi Lihat Hasil *Voting* (SRS_001_03)

Kasus uji validasi lihat hasil *voting* oleh administrator sebagai berikut.

Tabel 5. 3 Kasus uji validasi lihat hasil *voting* (SRS_001_03)

Nama kasus uji	Lihat data result *voting*
Tujuan pengujian	Untuk menguji validitas kinerja dari sistem dalam menyediakan fasilitas lihat data hasil *voting*.
Prosedur uji	1. Administrator telah melakukan *login* 2. Masuk ke halaman *result*
Hasil yang diharapkan	Sistem dapat menampilkan data hasil voting
Hasil yang didapatkan	Sistem menampilkan data hasil voting
Status Validitas	Valid

4. Kasus Uji Validasi Lihat Hasil *Hash* dan Simpan *Envelope Key* (SRS_001_04)

Kasus uji validasi lihat hasil *hash* dan simpan *envelope key* oleh administrator sebagai berikut.

Tabel 5. 4 Kasus uji validasi lihat hasil *hash* dan simpan *envelope key* (SRS_001_04)

Nama kasus uji	Lihat hasil *hash*
Tujuan pengujian	Untuk menguji validitas kinerja dari sistem dalam menyediakan fasilitas melihat hasil *hash*
Prosedur uji	1. Administrator melakukan *login* 2. Administrator masuk ke halaman *result* 3. Administrator menekan tombol *backup database* 4. Masuk ke halaman *backup*
Hasil yang diharapkan	Sistem dapat melakukan *hashing* pada hasil voting yang kemudian hasilnya ditampilkan pada halaman *backup* dan melakukan *generate envelope key* yang disimpan dalam file untuk diunduh bersama database.
Hasil yang didapatkan	Sistem melakukan *hashing* pada hasil voting yang kemudian hasilnya ditampilkan pada halaman *backup* dan melakukan *generate envelope key* yang sisimpan dalam file untuk diunduh bersama database.
Status Validitas	Valid

5. Kasus Uji Validasi *Backup Database* (SRS_001_05)

Kasus uji validasi olah data *backup database* oleh administrator sebagai berikut.

Tabel 5. 5 Kasus uji validasi *backup database* (SRS_001_05)

Nama kasus uji	*Backup database*
Tujuan pengujian	Untuk menguji validitas kinerja dari sistem dalam menyediakan fasilitas *backup database* dan dapat

	melakukan *download database* dan *message digest* oleh administrator
Prosedur uji	1. Administrator melakukan *login* 2. Administrator masuk ke halaman *result* 3. Administrator menekan tombol *backup database* 4. Masuk ke halaman backup 5. Administrator menekan tombol download hash dan menekan tombol download database
Hasil yang diharapkan	Sistem dapat melakukan *backup* terhadap *database* dan menyediakan fasilitas *download database* untuk dapat disimpan oleh administrator
Hasil yang didapatkan	Sistem melakukan *backup* terhadap *database* dan menyediakan fasilitas *download database* untuk dapat disimpan oleh administrator
Status Validitas	Valid

6. Kasus Uji Validasi *Voting* (SRS_002_01)

Kasus uji validasi *voting* oleh *voter* sebagai berikut.

Tabel 5. 6 Kasus uji validasi *voting* (SRS_002_01)

Nama kasus uji	*Voting*
Tujuan pengujian	Untuk menguji validitas kinerja dari sistem dalam menyediakan fasilitas *voting*
Prosedur uji	1. User melakukan *login* 2. User masuk ke halaman login 3. User memilih kandidat 4. User menekan tombol vote
Hasil yang diharapkan	Sistem dapat melakukan seleksi user yang belum dan sudah melakukan *vote*. Sistem dapat menampilkan halaman *vote* hanya pada user yang belum melakukan *vote*.
Hasil yang didapatkan	Sistem melakukan seleksi user yang belum dan sudah melakukan *vote*. Sistem menampilkan halaman *vote*

	pada user yang belum melakukan *vote*.
Status Validitas	Valid

5.1.2 Pembahasan Pengujian Validasi

Proses pembahasan terhadap hasil pengujian fungsional berdasarkan diagram *use case* dilakukan dengan melihat konformitas antara hasil kinerja sistem dengan daftar kebutuhan. Berdasarkan hasil pengujian fungsional berdasarkan diagram *use case*, dapat disimpulkan bahwa implementasi dan fungsionalitas sistem *e-voting* telah memenuhi kebutuhan yang telah dijabarkan pada tahap analisa kebutuhan. Hal tersebut terlihat pada hasil pengujian dimana, semua kasus pengujian memiliki status valid.

Dilihat dari kebutuhan non-fungsional untuk *control* dan *security*, sistem dapat mengontrol akses pada sistem, terlihat pada pengujian fungsional login sistem yang berhasil dilakukan.

Tabel 5. 7 Hasil uji validasi

No	Kasus Uji	Validitas
1	Olah data voter	Valid
2	Olah data kandidat	Valid
3	Lihat data result *voting*	Valid
4	*Backup database*	Valid
5	Lihat hasil *hash*	Valid
6	*Voting*	Valid

5.2 Pengujian Verifikasi

5.2.1 Hasil Pengujian Verifikasi

Pada skripsi ini, sistem hanya menerapkan aspek *integrity* sedangkan untuk menghasilkan sistem yang utuh harus menerapkan semua aspek keamanan. Pengujian verifikasi digunakan untuk mengetahui apakah proses hash dan digital signature yang telah dilaksanakan telah berfungsi sesuai dengan tujuan. Pada pengujian ini dapat diketahui apakah data yang di

dekripsi sesuai dengan data asli dan bagaimana jika terjadi perubahan pada data selama proses pengiriman.

Tabel 5. 8 Kasus uji verifikasi *digital signature*

Nama kasus uji	Verifikasi *digital signature*
Tujuan pengujian	Untuk menguji validitas hasil verifikasi *digital signature* pada file.
Prosedur uji	Melakukan verifikasi *signature* menggunakan kunci publik *signature* dengan mencocokkan pasangan kunci yang terdapat pada file.
Hasil yang diharapkan	*Digital signature* dapat diverifikasi menggunakan pasangan kunci publik. Jika file yang diverifikasi adalah file yang asli tanpa mengalami perubahan apapun maka *signature* dapat terverifikasi dengan baik. File yang mengalami perubahan nama ataupun konten di dalamnya dapat menghasilkan hasil verifikasi yang buruk dan *digital signature* tidak valid.
Hasil yang didapatkan	*Digital signature* diverifikasi menggunakan pasangan kunci. Status *signature* terverifikasi ketika memverifikasi file yang asli tanpa mengalami perubahan apapun. Status *signature* buruk ketika memverifikasi file yang telah mengalami perubahan nama maupun konten di dalamnya.
Status Validitas	Valid

Tabel 5. 9 Kasus uji verifikasi hasil enkripsi

Nama kasus uji	Verifikasi hasil enkripsi
Tujuan pengujian	Untuk menguji validitas hasil enkripsi pada file.
Prosedur uji	Melakukan dekripsi pada file yang telah melalui proses verifikasi dengan menggunakan pasangan kunci privat dan envelope key untuk file database.
Hasil yang diharapkan	File dapat terdekripsi dengan baik. Setiap dekripsi file dapat menghasilkan karakter yang dapat dibaca. Untuk

	dekripsi file *hash* dapat menghasilkan *message digest* yang sama dengan *message digest* yang ada pada web. Sedangkan untuk dekripsi file *database* dapat menghasilkan data yang sama dengan *database* asal dan dapat di import ke dalam database baru.
Hasil yang didapatkan	File terdekripsi dengan baik. Dekripsi tiap file menghasilkan karakter yang dapat dibaca. Untuk dekripsi file *hash* menghasilkan *message digest* yang sama dengan *message digest* yang ada pada web. Sedangkan untuk dekripsi file *database* menghasilkan data yang sama dengan *database* asal dan dapat diimport ke dalam database baru.
Status Validitas	Valid

Proses analisis bertujuan untuk mendapatkan kesimpulan dari hasil pengujian sistem *e-voting*, proses *hashing*, dan proses *digital signing* yang telah dilakukan. Proses analisis mengacu pada dasar teori sesuai dengan hasil pengujian yang didapatkan. Analisis dilakukan terhadap hasil pengujian di setiap tahap pengujian. Proses analisis yang dilakukan meliputi analisis hasil pengujian validasi dan analisis hasil pengujian validasi verifikasi.

5.2.2 Pembahasan Pengujian Verifikasi

Proses analisis terhadap pengujian verifikasi dilakukan dengan melihat konformitas antara hasil kinerja proses *hashing* dan *digital signing* dengan kebutuhan. Berdasarkan hasil pengujian verifikasi dapat disimpulkan bahwa implementasi *hash* dan *digital signature* pada sistem *e-voting* telah memenuhi kebutuhan yang telah dijabarkan pada tahap analisa kebutuhan. Hal tersebut terlihat pada hasil pengujian dimana, semua kasus pengujian memiliki status valid.

Tabel 5. 10 Hasil uji verifikasi

No	Kasus Uji		Hasil Verifikasi		Validitas
	Nama File	Konten	Signature	Dekripsi	
1	hash-db.txt	*Digest* awal terenkripsi	Verified	Sesuai dengan *digest* awal	Valid
2	db-backup.sql	*Dump database* terenkripsi	Verified	Sesuai dengan isi *database*	Valid
3	hash-db_.txt	*Digest* awal terenkripsi	Bad	Dekripsi gagal	Valid
4	db-backup_.sql	*Dump database* terenkripsi	Bad	Dekripsi gagal	Valid
5	hash-db.txt	Digest buatan	Bad	Dekripsi gagal	Valid
6	db-backup.sql	*Dump database* buatan	Bad	Dekripsi gagal	Valid
7	hash-db_.txt	Digest buatan	Bad	Dekripsi gagal	Valid
8	db-backup_.sql	*Dump database* buatan	Bad	Dekripsi gagal	Valid

Pada pengujian verifikasi, *digital signature* hanya dikatakan valid dan dekripsi dapat dilakukan jika file benar-benar asli tanpa perubahan dan berasal dari sumber yang sebenarnya. Jika dalam perjalanan file diubah-ubah atau ditukar dengan file yang lain maka akan menghasilkan signature yang tidak valid atau bad dan file akan gagal didekripsi.

BAB VI
PENUTUP

6.1 Kesimpulan

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

1. Perancangan sistem e-voting telah dibuat sesuai analisa kebutuhan untuk pengembangan sistem e-voting ini.
2. Sistem e-voting ini telah dibuat sesuai perancangan dan dapat digunakan untuk melakukan proses voting dan dapat menjaga integritas data hasil voting.
3. Proses *hashing* dan *digital signing* pada sistem e-voting ini telah berjalan dengan baik untuk dapat menjaga integritas data hasil voting sehingga dapat di verifikasi bahwa data hasil e-voting mengalami perubahan atau tidak selama proses pengiriman dan berasal dari pengirim yang sebenarnya atau tidak.

6.2 Saran

Saran yang dapat diberikan untuk pengembangan perangkat lunak ini antara lain:

1. Karena pada penelitian ini berfokus pada penjagaan integritas suatu data, untuk penelitian selanjutnya dapat ditambahkan untuk aspek keamanan lainnya seperti *confidentiality*, *authentication*, dan *availability*.
2. Penambahan fitur-fitur yang berguna untuk sistem e-voting sehingga pengguna tidak hanya dapat melakukan voting.
3. Sistem e-voting ini bisa terealisasikan sehingga pengguna dari semua kalangan masyarakat bisa lebih percaya pada hasil e-voting.

DAFTAR PUSTAKA

[AGI-12] PHP adalah - Hypertext Preprocessor . http://agiptek.com/index.php/php/101-php.html. 10 Desember 2013

[AZH-13] Azhar, Hanifah. 2013. Perbandingan Algoritma Fungsi Hash MD5 dengan SHA-1. Bandung:Institut Teknologi Bandung

[BPP-10] Badan Pengkajian dan Penerapan Teknologi . E-VOTING UNTUK PEMILU 2014. 2010. http://www.bppt.go.id/index.php/terkini/58-teknologi-material/425-e-voting-untuk-pemilu-2014. 18 Desember 2013

[BPT-10] Badan Pengkajian dan Penerapan Teknologi. E-Voting untuk Pemilu 2014. 2010. http://w1.bppt.go.id/index.php/terkini/58-teknologi-material/425-e-voting-untuk-pemilu-2014. 8 Desember 2013.

[CAU-11] Utama, Candra. 2011. CodeIgniter Framework. Bandung: Universitas Pasundan.

[KAT-13] Okky Feliantiar. Prospek e-voting menarik berbagai reaksi di Indonesia. 2013. http://khabarsoutheastasia.com/id/articles/apwi/articles/features/2013/05/09/feature-03. 8 Desember 2013.

[MAN-10] Pudja Mansyurin. 2010. Debian Web Server with OpenSSL (HTTPS). http://lebaksono.wordpress.com/2010/12/20/debian-web-server-with-openssl-https. 9 Desember 2013.

[MGL-07] Mogollon, Manuel. 2007. Cryptography and Security Sevices: Mechanisms and Applications. USA: Unversty of Dallas.

[NUG-09] Nugroho, Eddy Prasetyo. Komala Ratnasari. Kurniawan Nur Ramadhani, dkk. 2009. Rekayasa Perangkat Lunak. Bandung:Politeknik Telkom

[PRE-01] Pressman, R. (2001). S*oftware Engineering : A Practitioner's Approach, Fifth Edition*. McGraw Hill.

[ROK-11] Ali Rokhman. Prospek Penerapan E-Voting di Indonesia. 2011. http://map.unsoed.ac.id/2011/11/29/prospek-penerapan-e-voting-di-indonesia. 18 Desember 2013

[SAP-12] Edited by Saputro, Haris. 2012. Pembelajaran Praktek Basis Data (MySQL). Sumatra Selatan:Perguruan Tinggi AMIK AKMI Baturaja

[SAS-09] Aditya Sasongko . Tentang OpenSSL . 2009. http://littlebro-note.blogspot.com/2009/08/tentang-openssl.html diakses pada tanggal 9 Desember 2013

[SET-13] Sekretariat Kabinet Republik Indonesia. BPPT Sukses Uji Coba E-voting Berbasis E-KTP di Jembrana, Bali. 2013. http://setkab.go.id/berita-9725-bppt-sukses-uji-coba-e-voting-berbasis-e-ktp-di-jembrana-bali.html. 18 Desember 2013.

[SOE-13] Soewito, Benfano. 2013. Konsep Enkripsi dan Dekripsi Berdasarkan Kunci Tidak Simetris. Jakarta:Universitas Bina Nusantara.

[UPN-13] Universitas Pembangunan Nasional. http://www.library.upnvj.ac.id/pdf/2s1teknikinformasi/205511014/bab2.pdf. 8 Desember 2013.