# LAPORAN PRAKTIK KERJA LAPANGA­­N

**APLIKASI DOKUMENTASI PERSURATAN BERBASIS *WEB***

**PADA BALAI BESAR PERIKANAN BUDIDAYA AIR PAYAU**

**(BBPBAP) JEPARA JAWA TENGAH**

****

**Disusun oleh :**

**INDRA MALIK**

**24010313120007**

**DEPARTEMEN ILMU KOMPUTER/ INFORMATIKA**

**FAKULTAS SAINS DAN MATEMATIKA**

**UNIVERSITAS DIPONEGORO**

**SEMARANG**

**2016**

# HALAMAN PENGESAHAN

Laporan PKL dengan judul :

**“Aplikasi Dokumentasi Persuratan Berbasis *Web* Pada Balai Besar Perikanan Budidaya Air Payau (BBPBAP) Jepara Jawa Tengah”**

Disusun oleh :

Nama : Indra Malik

NIM : 24010313120007

Departemen : Ilmu Komputer/ Informatika

Fakultas : Sains dan Matematika

Telah diseminarkan pada seminar Laporan PKL pada tanggal 10 Oktober 2016

Semarang, 10 Oktober 2016

Mengetahui,

Koordinator PKL

Khadijah, S.Kom, M.Cs

NIP. 198903032015042002

Menyetujui,

Dosen Pembimbing

Nurdin Bahtiar, S.Si, M.T

NIP. 197907202003121002

Mengetahui,

Ketua Departemen Ilmu Komputer/Informatika

FSM UNDIP

Ragil Saputra, S.Si, M.Cs

NIP. 19801021200501100

# ABSTRAK

Balai Besar Perikanan Budidaya Air Payau Jepara Jawa Tengah bertugas untuk melaksanakan pengembangan dan penerapan teknik pembenihan, pembudidayaan, pengelolaan kesehatan ikan dan pelestarian lingkungan budidaya Perikanan. Dalam instansi tersebut, pengelolaan dan pendokumentasian berbagai macam surat, seperti surat masuk dan surat keluar masih dilakukan secara konvensional dan kurang efektif. Pengelolaan dan pendokumentasian yang dilakukan secara konvensional ini menyebabkan butuhnya waktu yang cukup lama untuk mencari surat yang tepat bila sedang membutuhkan surat yang dimaksud. Oleh karena itu, perlu dibangun sebuah aplikasi yang memiliki fitur untuk mengelola dan mendokumentasikan surat masuk dan surat keluar. Aplikasi Dokumentasi Persuratan Berbasis *Web* ini dirancang dengan metode pengembangan *Waterfall Model* berbasis *web* menggunakan bahasa pemrograman PHP, sistem manajemen basis data MySQL*,* dan *framework* Codeigniter. Berdasarkan hasil pengujian yang dilakukan dengan metode *black box*, sistem informasi ini telah memenuhi kebutuhan fungsional. Aplikasi Dokumentasi Persuratan Berbasis *Web* ini diharapkan dapat melakukan berbagai aktivitas pengelolaan data surat masuk dan surat keluar barang dengan efisien dan efektif. Tujuan dari aplikasi ini adalah untuk meningkatkan kemudahan dan efisiensi pendokumentasian serta pengelolaan surat masuk dan surat keluar di Balai Besar Perikanan Budidaya Air Payau (BBPBAP) Jepara.

**Kata Kunci:** Balai Besar Perikanan Budidaya Air Payau (BBPBAP) Jepara Jawa Tengah, Aplikasi Dokumentasi Persuratan, *Object Oriented.*

# ABSTRACT

Great Hall of Brackish Water Aquaculture Jepara, Central Java commissioned to carry out the development and application of techniques seeding, cultivation, fish health management and environmental sustainability of aquaculture Fisheries. In these agencies, management and documentation of various sorts of letters, such as incoming and outgoing mail is still done conventionally and less effective. Management and documentation is done conventionally quickness This causes quite a long time to find the right letters when you are in need of the letter in question. Therefore, it is necessary to build an application that has the features to manage and document incoming and outgoing mail. Web-Based Documentation mailing application is designed with Model-based development methods Waterfall web using PHP programming language, database management system MySQL, and CodeIgniter framework. Based on the results of tests performed by the black box method, information systems have met the functional requirements. Web-Based Application Documentation mailing is expected to perform a variety of data management activities of incoming mail and outgoing goods efficiently and effectively. The purpose of this application is to increase the ease and efficiency of documentation and management of incoming and outgoing mail at the Great Hall of Brackish Water Aquaculture (BBPBAP) Jepara.

**Keywords**: Great Hall of Brackish Water Aquaculture Jepara Central Java, Application Documentation Mailing, Object Oriented.

# KATA PENGANTAR

Segala puji syukur penulis panjatkan kepada Allah SWT atas karunia-Nya yang diberikan, sehingga penulis dapat menyelesaikan laporan Praktik Kerja Lapangan (PKL) yang berjudul “Aplikasi Dokumentasi Persuratan Berbasis *Web* Pada Balai Besar Perikanan Budidaya Air Payau (BBPBAP) Jepara Jawa Tengah”.

Dalam penyusunan laporan ini tentulah banyak mendapat bimbingan dan bantuan dari berbagai pihak. Untuk itu, pada kesempatan ini penulis mengucapkan rasa hormat dan terima kasih kepada:

1. Ragil Saputra, S.Si, M.Cs. selaku Ketua Departemen Ilmu Komputer/ Informatika yang membantu dalam proses perijinan PKL di Balai Besar Perikanan Budidaya Air Payau (BBPBAP) Jepara.
2. Khadijah, S.Kom, MCs. selaku Koordinator PKL yang membantu dalam proses perijinan PKL di Balai Besar Perikanan Budidaya Air Payau (BBPBAP) Jepara.
3. Nurdin Bahtiar, S.Si, M.T, selaku dosen pembimbing yang telah membantu dalam proses perijinan dan bimbingan hingga terselesaikannya laporan PKL ini.
4. Bapak Abdullah selaku bidang teknologi dan pihak bagian divisi pelayanan masyarakat pada Balai Besar Perikanan Budidaya Air Payau (BBPBAP) Jepara.
5. Semua pihak yang telah membantu kelancaran dalam pelaksanaan PKL, yang tidak dapat penulis sebutkan satu persatu.

Penulis menyadari bahwa dalam laporan ini masih banyak kekurangan baik dari segi materi ataupun dalam penyajiannya, kritik dan saran sangat penulis harapkan untuk perbaikan pada penulisan ilmiah yang akan datang.

Penulis berharap laporan ini dapat bermanfaat bagi pembaca pada umumnya dan penulis sendiri pada khususnya.

Semarang, 10 Oktober 2016

Penulis,

Indra Malik

24010313120007

# DAFTAR ISI

[HALAMAN SAMPUL i](#_Toc464548021)

[HALAMAN PENGESAHAN ii](#_Toc464548022)

[ABSTRAK iii](#_Toc464548023)

[ABSTRACT iv](#_Toc464548024)

[KATA PENGANTAR v](#_Toc464548025)

[DAFTAR ISI vi](#_Toc464548026)

[DAFTAR GAMBAR ix](#_Toc464548027)

[DAFTAR TABEL xi](#_Toc464548028)

[BAB I PENDAHULUAN 1](#_Toc464548029)

[1.1 Latar Belakang Masalah 1](#_Toc464548030)

[1.2 Rumusan Masalah 2](#_Toc464548031)

[1.3 Tujuan dan Manfaat 2](#_Toc464548032)

[1.4 Ruang Lingkup 2](#_Toc464548033)

[1.5 Sistematika Penulisan 3](#_Toc464548034)

[BAB II TINJAUAN ORGANISASI 4](#_Toc464548035)

[2.1 Profil Organisasi 4](#_Toc464548036)

[2.2 Visi 5](#_Toc464548037)

[2.3 Misi 5](#_Toc464548038)

[2.4 Struktur Organisasi 5](#_Toc464548039)

[BAB III DASAR TEORI 7](#_Toc464548040)

[3.1 Pengertian Aplikasi Berbasis *Web* 7](#_Toc464548041)

[3.2 Model Proses Perangkat Lunak 7](#_Toc464548042)

[3.3 Desain Perangkat Lunak 10](#_Toc464548043)

[3.4 *Unified Modeling Language* (UML) 11](#_Toc464548044)

[3.4.1 *Things* 11](#_Toc464548045)

[3.4.2 *Relationship* 12](#_Toc464548046)

[3.4.3 Diagram 12](#_Toc464548047)

[3.5 Konsep *Object Oriented* 13](#_Toc464548048)

[3.6 PHP (*Hypertext Preprocessor*) 14](#_Toc464548049)

[3.7 *Framework* Codeigniter 14](#_Toc464548050)

[3.8 DBMS MySQL 15](#_Toc464548051)

[BAB IV ANALISIS DAN PERANCANGAN 16](#_Toc464548052)

[4.1 *Requirements Analysis* (Analisis Kebutuhan) 16](#_Toc464548053)

[4.1.1 Deskripsi Kebutuhan Perangkat Lunak 16](#_Toc464548054)

[4.1.2 Deskripsi Umum Perangkat Lunak 16](#_Toc464548055)

[4.1.3 Karakteristik Pengguna 16](#_Toc464548056)

[4.1.4 Kebutuhan Perangkat Lunak 17](#_Toc464548057)

[4.1.5 Kebutuhan Fungsional Aplikasi 17](#_Toc464548058)

[4.1.6 Kebutuhan Non Fungsional Sistem 18](#_Toc464548059)

[4.2 *Problem Analysis* (Analisis Permasalahan) 18](#_Toc464548060)

[4.2.1 Daftar *Use Case* 18](#_Toc464548061)

[4.2.2 Pemodelan *Use case* Diagram 19](#_Toc464548062)

[4.2.3 *Use case* Detail 20](#_Toc464548063)

[4.2.4 *Activity* *Diagram* 30](#_Toc464548064)

[4.3 *Class Analysis* 34](#_Toc464548065)

[4.4 Perancangan 36](#_Toc464548066)

[4.4.1 Realisasi *Use Case* Tahap Perancangan 36](#_Toc464548067)

[4.4.2 Perancangan *Database* 38](#_Toc464548068)

[4.5 Pemodelan *Class Diagram* 40](#_Toc464548069)

[BAB V IMPLEMENTASI DAN ANALISIS HASIL 41](#_Toc464548070)

[5.1 Spesifikasi Perangkat 41](#_Toc464548071)

[5.2 Implementasi Basis Data 41](#_Toc464548072)

[5.3 Implementasi *Activity Diagram* 44](#_Toc464548073)

[5.4 Implementasi Antarmuka 48](#_Toc464548074)

[5.4 Pengujian 54](#_Toc464548075)

[BAB VI PENUTUP 56](#_Toc464548076)

[6.1 Kesimpulan 56](#_Toc464548077)

[6.2 Saran 56](#_Toc464548078)

[DAFTAR PUSTAKA 57](#_Toc464548079)

[LAMPIRAN 59](#_Toc464548080)

[Lampiran 1. 59](#_Toc464548081)

[Lampiran 2. 64](#_Toc464548082)

[Lampiran 3. 66](#_Toc464548083)

[Lampiran 4. 68](#_Toc464548084)

[Lampiran 5. 70](#_Toc464548085)

[Lampiran 6. 72](#_Toc464548086)

# DAFTAR GAMBAR

[Gambar 2.1 Struktur Organisasi BBPBAP Jepara 6](file:///D:\Indra\Ilmu%20Komputer\Semester%207\Tugas\Seminar\Draft.docx#_Toc464542830)

[Gambar 3.1 Waterfall Model………………………………………………………………..8](#_Toc464542838)

[Gambar 4.1 *Use* *Case Diagram* Aplikasi Dokumentasi Persuratan………………………..19](#_Toc464542905)

[Gambar 4.2 Sketsa Halaman *Form Login* 20](#_Toc464542906)

[Gambar 4.3 Sketsa Halaman Dasbor 21](#_Toc464542907)

[Gambar 4.4 Sketsa Halaman Form Masukan Surat Masuk 22](#_Toc464542908)

[Gambar 4.5 Sketsa Halaman Form Masukan Surat Keluar 23](#_Toc464542909)

[Gambar 4.6 Sketsa Antarmuka Halaman *File* Surat Masuk 24](#_Toc464542910)

[Gambar 4.7 Sketsa Antarmuka Halaman *File* surat keluar 24](#_Toc464542911)

[Gambar 4.8 Sketsa Halaman Form Ubah Surat Masuk 26](#_Toc464542912)

[Gambar 4.9 Sketsa Halaman Form Ubah Surat Keluar 26](#_Toc464542913)

[Gambar 4.10 Sketsa Halaman Daftar Surat Masuk Dengan *Alert* Menghapus 28](#_Toc464542914)

[Gambar 4.11 Sketsa Halaman Daftar Surat Keluar Dengan *Alert* Menghapus 28](#_Toc464542915)

[Gambar 4.12 Sketsa Halaman *Update Password* Administrator 29](#_Toc464542916)

[Gambar 4.13 Diagram Aktivitas *Login* 30](#_Toc464542917)

[Gambar 4.14 Diagram Aktivitas Masukan Surat Masuk Dan Surat Keluar 31](#_Toc464542918)

[Gambar 4.15 Diagram Aktivitas Menampilkan *File* Surat Masuk Dan Surat Keluar 31](#_Toc464542919)

[Gambar 4.16 *Activity Diagram* Untuk Mengubah Data Surat Masuk dan Surat Keluar 32](#_Toc464542920)

[Gambar 4.17 Diagram Aktivitas Untuk Menghapus Surat Masuk dan Surat Keluar 33](#_Toc464542921)

[Gambar 4.18 Diagram Aktivitas Untuk Mengubah *Password* Administrator 33](#_Toc464542922)

[Gambar 4.19 *Class Diagram* Aplikasi Dokumentasi Persuratan 40](#_Toc464542923)

[Gambar 5.1 Implementasi Tabel admin…………………………………………………....42](#_Toc464542957)

[Gambar 5.2 Implementasi Tabel suratmasuk 43](#_Toc464542958)

[Gambar 5.3 Implementasi Tabel suratkeluar 43](#_Toc464542959)

[Gambar 5.4 Antarmuka Halaman *Login* 49](#_Toc464542960)

[Gambar 5.5 Halaman Dasbor 49](#_Toc464542961)

[Gambar 5.6 Halaman Form Masukan Surat Masuk 50](#_Toc464542962)

[Gambar 5.7 Halaman Form Masukan Surat Keluar 50](#_Toc464542963)

[Gambar 5.8 Halaman Tabel Daftar Surat Masuk 51](file:///D:\Indra\Ilmu%20Komputer\Semester%207\Tugas\Seminar\Draft.docx#_Toc464542964)

[Gambar 5.9 Tabel Surat Keluar 51](#_Toc464542965)

[Gambar 5.10 *Update* Surat Masuk 52](file:///D:\Indra\Ilmu%20Komputer\Semester%207\Tugas\Seminar\Draft.docx#_Toc464542966)

[Gambar 5.11 *Update* Surat Keluar 52](file:///D:\Indra\Ilmu%20Komputer\Semester%207\Tugas\Seminar\Draft.docx#_Toc464542967)

[Gambar 5.12 *Alert* Muncul Ketika *Link* “Ubah” Atau “Hapus” Diklik 53](#_Toc464542968)

[Gambar 5.13 *Alert* Muncul Ketika *Link* “Ubah” Atau “Hapus” Diklik 53](#_Toc464542969)

[Gambar 5.14 Halaman Form *Update* *password* administrator 54](#_Toc464542970)

# DAFTAR TABEL

[Tabel 3.1 Tabel SRS 8](#_Toc464543335)

[Tabel 4.1 Karakteristik Pengguna………………………………………………………….16](#_Toc464543347)

[Tabel 4.2 Kebutuhan Fungsional 17](#_Toc464543348)

[Tabel 4.3 Kebutuhan Non Fungsional 18](#_Toc464543349)

[Tabel 4.4 Daftar *Use Case* 18](#_Toc464543350)

[Tabel 4.5 *Use case* Detail untuk Melakukan *Login* 20](#_Toc464543351)

[Tabel 4.6 *Use case* Detail Memasukkan Data Surat Masuk dan Surat Keluar 21](#_Toc464543352)

[Tabel 4.7 *Use case* Detail Menampilkan *File* Surat Masuk dan Surat Keluar 23](#_Toc464543353)

[Tabel 4.8 *Use case* Detail Mengubah Data Surat Masuk dan Surat Keluar 25](#_Toc464543354)

[Tabel 4.9 *Use Case* Detail Menghapus Data Surat Masuk dan Surat Keluar 27](#_Toc464543355)

[Tabel 4.10 *Use Case* Detail Mengubah *Password* Administrator 29](#_Toc464543356)

[Tabel 4.11 Identifikasi *Class Analysis* 34](#_Toc464543357)

[Tabel 4.12 Tanggung jawab dan Atribut *Class* 34](#_Toc464543358)

[Tabel 4.13 Identifikasi *Class* Perancangan *Use Case* *Login* 36](#_Toc464543359)

[Tabel 4.14 Identifikasi *Class* Perancangan *Use Case* memasukkan data surat masuk dan surat keluar 36](#_Toc464543360)

[Tabel 4.15 Identifikasi *Class* Perancangan *Use Case* menampilkan *file* surat masuk dan surat keluar 37](#_Toc464543361)

[Tabel 4.16 Identifikasi *Class* Perancangan *Use Case* mengubah data surat masuk dan surat keluar 37](#_Toc464543362)

[Tabel 4.17 Identifikasi *Class* Perancangan *Use Case* menghapus data surat masuk dan surat keluar 37](#_Toc464543363)

[Tabel 4.18 Identifikasi *Class* Perancangan *Use Case* mengubah *password* administrator 38](#_Toc464543364)

[Tabel 4.19 Daftar Tabel administrator 38](#_Toc464543365)

[Tabel 4.20 Daftar Tabel suratmasuk 39](#_Toc464543366)

[Tabel 4.21 Daftar Tabel suratkeluar 39](#_Toc464543367)

[Tabel 5.1 Tabel Rencana Pengujian………………………………………………………..55](#_Toc464543398)

# BAB I PENDAHULUAN

Bab ini membahas latar belakang masalah, rumusan masalah, tujuan dan manfaat, serta sistematika penulisan laporan yang dibuat.

## 1.1 Latar Belakang Masalah

Balai Besar Perikanan Budidaya Air Payau (BBPBAP) Jepara memiliki tugas pokok, yaitu melaksanakan pengembangan dan penerapan teknik perbenihan, pembudidayaan, pengelolaan kesehatan ikan dan pelestarian lingkungan budidaya khususnya di wilayah Jepara. Selain itu, BBPBAP merupakan salah satu pusat pengembangan IPTEK di bidang akuakultur sehingga menjadikannya tempat yang menarik untuk menimba ilmu tidak terkecuali Ilmu Komputer yang berperan di dalamnya.

Kantor Balai Besar Perikanan Budidaya Air Payau (BBPBAP) Jepara hampir setiap minggunya menerima berbagai surat masuk dari banyak instansi, baik itu dari pemerintahan maupun swasta. Biasanya isi dari surat-surat tersebut seputar perizinan untuk praktik kerja lapangan, kunjungan ilmiah, dan lain sebagainya. Menanggapi surat-surat masuk tersebut, BBPBAP Jepara juga menyurati pihak terkait untuk memberikan jawaban apakah diterima atau tidak. Seringkali, kejadian di lapangan surat masuk yang sudah diterima sulit untuk dicari karena banyaknya surat-surat yang ada. Hal ini dapat diatasi dengan perancangan suatu aplikasi yang dapat berjalan secara efektif dan efisien. Maka dari itu perlu dirancang suatu aplikasi dokumentasi persuratan berbasis *web* pada Balai Besar Perikanan Budidaya Air Payau (BBPBAP) Jepara.

Praktik Kerja Lapangan (PKL) yang akan dilakukan adalah pembuatan Aplikasi Dokumentasi Persuratan Berbasis *Web* Pada Balai Besar Perikanan Budidaya Air Payau (BBPBAP) Jepara. Aplikasi ini dibuat dengan berbasis *web* dikarenakan cara penggunaanya yang mudah. Aplikasi ini ditujukan untuk membantu dalam mengelola data surat masuk dan surat keluar, seperti memasukkan, mengubah, melihat, menghapus, dan mengirim data yang dikelola langsung oleh seorang administrator.

## 1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah dijelaskan, maka dapat dibuat rumusan masalah yaitu bagaimana membuat suatu aplikasi dokumentasi persuratan berbasis *web* dengan tujuan mempermudah dalam pendokumentasian data surat masuk dan surat keluar pada Balai Besar Perikanan Budidaya Air Payau (BBPBAP) Jepara.

## 1.3 Tujuan dan Manfaat

Tujuan dilaksanakannya Praktik Kerja Lapangan (PKL) ini adalah untuk menghasilkan sebuah Aplikasi Dokumentasi Persuratan Berbasis *Web* pada Balai Besar Perikanan Budidaya Air Payau (BBPBAP) Jepara dan berfungsi secara baik serta digunakan oleh administrator sebagai pengguna pada sistem informasi tersebut.

Manfaat dilaksanakannya Praktik Kerja Lapangan (PKL) ini adalah :

1. Bagi Organisasi

Balai Besar Perikanan Budidaya Air Payau (BBPBAP) Jepara dapat menggunakan aplikasi ini untuk mengelola data surat masuk dan surat keluar. Melalui pengelolaan data secara terorganisir dapat mempermudah administrator untuk mendokumentasikan berbagai surat yang ada.

2. Bagi Mahasiswa

Mahasiswa mendapatkan pengetahuan dan pengalaman baru dalam bidang perencanaan, analisis, perancangan, pembuatan, dan pengujian aplikasi berbasis *web*. Selain itu, mahasiswa dapat menerapkan secara langsung ilmu pengetahuan yang telah didapatkan di perkuliahan.

## 1.4 Ruang Lingkup

Ruang lingkup dari aplikasi yang akan dibangun pada Balai Besar Perikanan Budidaya Air Payau (BBPBAP) Jepara adalah :

1. Persoalan yang berhubungan dengan perancangan, pembuatan dan pengujian Aplikasi Dokumentasi Persuratan Berbasis *Web* Pada Balai Besar Perikanan Budidaya Air Payau (BBPBAP) Jepara.
2. Aplikasi ini ini dirancang dengan menggunakan metode *Waterfall*, yang akan dilakukan hanya sampai pada tahap implementasi dan pengujian.
3. Aplikasi yang dibuat berbasis *web*, menggunakan bahasa pemrograman PHP, *framework* Code Igniter, dan basis data MySQL

## 1.5 Sistematika Penulisan

Untuk memberikan suatu gambaran yang urut dan jelas mengenai pembahasan penyusunan Aplikasi Dokumentasi Berbasis *Web* pada Balai Besar Perikanan Budidaya Air Payau (BBPBAP) Jepara berikut ini disesuaikan dengan sistematika pembahasan, yaitu:

BAB I PENDAHULUAN

Bab ini membahas latar belakang masalah, rumusan masalah, tujuan dan manfaat, ruang lingkup, serta sistematika penulisan laporan yang dibuat.

BAB II TINJAUAN INSTANSI

Bab ini membahas mengenai mengenai data umum Balai Besar Perikanan Budidaya Air Payau (BBPBAP) Jepara, tempat penulis melaksanakan praktik kerja lapangan. Data umum tersebut meliputi sejarah instansi, profil instansi, visi-misi instansi, dan struktur instansi.

BAB III DASAR TEORI

Bab ini menjelaskan tentang teori-teori yang digunakan untuk membangun Aplikasi Dokumentasi Persuratan Berbasis *Web* pada Balai Besar Perikanan Budidaya Air Payau (BBPBAP) Jepara dan teori lain yang mendukung pengembangannya.

BAB IV ANALISIS DAN PERANCANGAN

Bab ini menjelaskan tentang pembahasan yang meliputi deskripsi umum perangkat lunak dan desain rancangan dari hasil Aplikasi Dokumentasi Persuratan Berbasis *Web* pada Balai Besar Perikanan Budidaya Air Payau (BBPBAP) Jepara.

BAB V IMPLEMENTASI DAN PENGUJIAN

Bab ini menjelaskan tentang hasil implementasi dari hasil rancangan dan hasil pengujian yang dilakukan terhadap Aplikasi Dokumentasi Persuratan Berbasis *Web* pada Balai Besar Perikanan Budidaya Air Payau (BBPBAP) Jepara.

BAB VI PENUTUP

Bab ini merupakan kesimpulan dari bab-bab yang dibahas sebelumnya dan saran bagi instansi dan pembaca sebagai bahan masukan.

# BAB II TINJAUAN ORGANISASI

## 2.1 Profil Organisasi

Nama Instansi : Balai Besar Perikanan Budidaya Air Payau (BBPBAP) Jepara

Alamat : Jalan Cik Lanang, Kec Jepara, Jawa Tengah, Indonesia

Telepon/ Fax : (0291) 591125

*E-mail* : bbpbapjpr@gmail.com

*Web*site : http://bbpbapjepara.djpb.kkp.go.id/

Balai Besar Perikanan Budidaya Air Payau Jepara didirikan pada tahun 1971, diawali dengan berdirinya lembaga *Research Center* Udang (RCU) yang secara hirarki berada di bawah Badan Penelitian dan Pengembangan Perikanan, Departemen Pertanian.

Sasaran utamanya adalah meneliti siklus hidup udang windu (*Penaeus monodon*) dari proses kematangan telur (gonad), perkembangan larva hingga dewasa secara terkendali untuk selanjutnya dibudidayakan di tambak.

Pada tahun 1978 berdasarkan SK Menteri Pertanian RI No. : 306/Kpts/Org/5/1978 tentang susunan organisasi dan tatalaksana balai, telah diatur dan ditetapkan lembaga yang semula bernama *Research Center* Udang menjadi Balai Budidaya Air Payau (BBAP). BBAP Jepara ini merupakan Unit Pelaksana Teknis (UPT) yang berada dibawah Direktorat Jenderal Perikanan, Departemen Pertanian. Seiring dengan perkembangan kemajuan teknologi akuakultur, dimana komoditas yang dikembangkan tidak hanya terbatas pada udang windu saja, tetapi juga komoditas ikan bersirip, ekinodermata dan moluska air.

Selanjutnya selain keberhasilannya dalam hal teknik ablasi mata, pada periode 1979-1988 BBAP Jepara juga telah berhasil melakukan pengkajian teknologi pembenihan udang skala rumah tangga (*backyard hatchery*). Dalam waktu yang singkat usaha *backyard hatchery* ini telah berkembang dan meningkatkan pendapatan masyarakat pesisir dan nelayan sekitar Jepara. Sejak tahun 1993 usaha ini mulai berkembang ke daerah-daerah lain di Indonesia.

Pada era masa kepemimpinan Presiden KH. Abdulrahman Wahid, telah dibentuk Departemen Eksplorasi Laut dan Perikanan yang merupakan cikal bakal Kementerian Kelautan dan Perikanan. Hingga akhirnya berdasarkan SK Menteri Kelautan dan Perikanan No. : 26C/MEN/2001, BBAP Jepara mengalami perubahan nama dan status (eselonisasi) menjadi Balai Besar Perikanan Budidaya Air Payau (BBPBAP), peningkatan status dari eselon III menjadi eselon II. Kedudukan BBPBAP Jepara merupakan Unit Pelaksana Teknis yang secara administratoristratif dan teknis bertanggung jawab pada Direktorat Jenderal Perikanan Budidaya, Kementerian Kelautan & Perikanan.

## 2.2 Visi

Visi dari Balai Besar Perikanan Budidaya Air Payau (BBPBAP) adalah “Penghasil Teknologi Budidaya Air Payau Yang Inovatif, Adaptif Untuk Mewujudkan Peningkatan Produksi Yang Berdaya Saing, Ramah Lingkungan dan Berkelanjutan”.

## 2.3 Misi

Misi dari Balai Besar Perikanan Budidaya Air Payau (BBPBAP) adalah “Meningkatkan kerekayasaan dan layanan prima dalam penerapan sistem budidaya air payau yang efektif, efisien dan menguntungkan”.

## 2.4 Struktur Organisasi

Struktur organisasi dari Balai Besar Perikanan Budidaya Air Payau (BBPBAP) Jepara sebagai berikut,

Kepala Balai : I Made Suitha, A.Pi

Kepala Bagian Tata Usaha : Ir. Wiwik Malistyani

Kepala Sub Bagian Kepegawaian : Ir. Hasan Rosyadi

Kepala Sub Bagian Keuangan dan Umum : M. Arifin

Kabid Uji Terap Teknik dan Kerjasama : Drs. Tri Prasetyo PU

Kabid Pengujian dan Dukungan Teknis : Yahya, SH

Kasie Uji Trap Teknik : V.Ap Sapto Adi, SP

Kasie Kerjasana dan Informasi : Agus Setiyadi, SH

Kasie Dukungan Teknis : Yulianto, ST

Kasie Produksi dan Pengujian : Drs. Dwi Soelistinarto



Gambar 2.1 Struktur Organisasi BBPBAP Jepara

# BAB III DASAR TEORI

Bab ini akan menjelaskan tentang teori-teori yang digunakan untuk membangun Aplikasi Dokumentasi Berbasis *Web* Persuratan pada Balai Besar Perikanan Budidaya Air Payau (BBPBAP) Jepara.

## 3.1 Pengertian Aplikasi Berbasis *Web*

Pada awalnya sistem berbasis *web* dibangun dengan hanya menggunakan bahasa yang disebut HTML (*HyperText Markup Langauge*). Pada perkembangan berikutnya, sejumlah skrip dan objek dikembangkan untuk memperluas kemampuan HTML, seperti PHP (Fathsansyah, 2007).

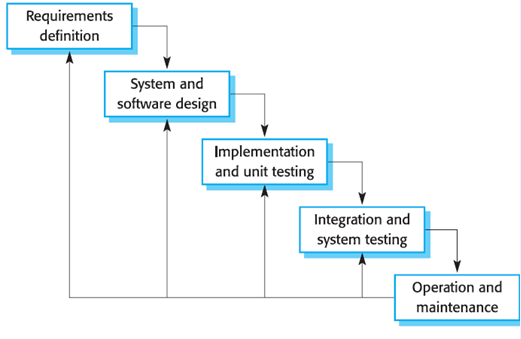
Arsitektur sistem berbasis *web* meliputi klien, *web* *server*, *middleware* dan basis data. Klien berinteraksi dengan *web* *server*. Secara internal, *web* *server* berkomunikasi dengan *middleware* dan *middleware* yang berkomunikasi dengan basis data. Contoh *middleware* adalah PHP. Pada mekanisme sistem berbasis *web*, terjadi tambahan proses yaitu *server* menerjemahkan kode PHP menjadi kode HTML. Kode PHP yang diterjemahkan oleh mesin PHP yang akan diterima oleh klien.

## 3.2 Model Proses Perangkat Lunak

Pengembangan perangkat lunak melewati beberapa tahapan dari sistem itu direncanakan sampai sistem itu dirancang, diimplementasikan, dioperasikan dan dipelihara. Siklus ini disebut siklus hidup pengembangan sistem (*System Development Life Cycle*).

Model yang digunakan dalam pengembangan Aplikasi Dokumentasi Persuratan pada Balai Besar Perikanan Budidaya Air Payau (BBPBAP) Jepara adalah model sekuensial linier (*Waterfall* Model).

Model sekuensial linier (disebut juga dengan siklus hidup klasik atau *Waterfall* Model). Model ini adalah model klasik yang bersifat sistematis, berurutan dalam membangun perangkat lunak mulai pada tingkat dan kemajuan sistem pada seluruh analisis, desain, kode, pengujian dan pemeliharaan.



Gambar 3.1 Waterfall Model

Tahap-tahap dalam *Waterfall* Model menurut referensi Pressman (2001) adalah sebagai berikut.

1. *Requirements Definition*

Seluruh kebutuhan *software* harus bisa didapatkan dalam tahap ini, kemudian dianalisis dan didefinisikan termasuk didalamnya kegunaan *software* yang diharapkan pengguna dan batasan *software*. Tahap ini harus dikerjakan secara lengkap untuk bisa menghasilkan desain yang lengkap. Tahap ini menghasilkan SRS (*Software Requirement Specification*). SRS adalah dokumen yang berisi deskripsi lengkap mengenai kemampuan *software*. SRS diperlukan karena banyak kesalahan yang timbul pada tahap requirement dan tidak terdeteksi sejak dini, serta untuk menghemat biaya perbaikan. Standar penulisan SRS yang digunakan adalah IEEE/ANSI 830-1998. Setiap *requirement* dalam SRS diberikan *unique identifier* label. Berikut format pendefinisan kebutuhan fungsional dalam SRS pada tabel 3.1.

Tabel 3.1 Tabel SRS

|  |  |
| --- | --- |
| SRS ID | Deskripsi |
| SRS-XXXX-F-YY | ……………..……………………………… |

Keterangan:

SRS : *Software Requirement Specification*

XXXX : *Nickname* aplikasi yang dibangun

FXX : F adalah fungsional dapat opsional berupa NF (Non Fungsional)

YY : Nomor urut SRS

1. *System and Software Design*

Desain dikerjakan setelah kebutuhan *software* selesai dianalisis secara lengkap. Tahap ini dilakukan sebelum melakukan *coding*. Tahap ini bertujuan untuk memberikan gambaran apa yang seharusnya dikerjakan dan bagaimana tampilannya. Tahap ini membantu dalam menspesifikasikan kebutuhan *hardware* dan sistem serta mendefinisikan arsitektur aplikasi secara keseluruhan.

1. *Implementation and Unit Testing*

Desain program diterjemahkan ke dalam kode-kode dengan menggunakan bahasa pemrograman yang sudah ditentukan. Selain itu dalam tahap ini juga dilakukan pemeriksaaan terhadap unit yang dibuat, apakah sudah memenuhi fungsi yang diinginkan atau belum.

1. *Integration and System Testing*

Penyatuan unit-unit program secara terintegrasi kemudian diuji secara keseluruhan (*system testing*).

1. *Operation and Maintenance*

Mengoperasikan program dilingkungannya dan melakukan pemeliharaan, seperti penyesuaian atau perubahan karena adaptasi dengan situasi sebenarnya.

Kelebihan *Waterfall* Model adalah lebih disiplin. Selain itu, dokumentasi selalu tersedia dalam setiap tahapan (*documentation driven*). Kekurangan yang utama dari model ini adalah kesulitan dalam mengakomodasi perubahan setelah proses dijalani. Fase sebelumnya harus lengkap dan selesai sebelum mengerjakan fase berikutnya. Serta, pengguna baru dapat melihat produk setelah selesai diprogram.

Masalah dengan *Waterfall* Model adalah perubahan sulit dilakukan karena sifatnya yang kaku. Karena sifat kakunya, model ini cocok ketika kebutuhan dikumpulkan secara­ lengkap sehingga perubahan bisa ditekan sekecil mungkin. Tapi pada kenyataannya jarang sekali pengguna yang bisa memberikan kebutuhan secara lengkap, perubahan kebutuhan adalah sesuatu yang wajar terjadi.

## 3.3 Desain Perangkat Lunak

Desain merupakan proses iteratif yang menerjemahkan *requirement* ke dalam sebuah ‘*blue print*’ untuk pembangunan *software* (Pressman, 2001). Tiga karakteristik yang berfungsi sebagai pedoman bagi evaluasi suatu desain yang baik adalah :

1. Desain harus mengimplementasi keseluruhan persyaratan eksplisit yang dibebankan dalam model analisis, dan harus mengakomodasi semua persyaratan implisit yang diinginkan pelanggan.
2. Desain harus menjadi pedoman yang besifat *readable*, dapat dipahami bagi mereka yang menghasilkan kode dan yang menguji serta yang memelihara perangkat lunak.
3. Desain harus memberikan suatu gambaran lengkap mengenai perangkat lunak, yang menekankan data, dan domain perilaku dari perspektif implementasi.

Desain *interface* menggambarkan bagaimana perangkat lunak berkomunikasi dalam dirinya sendiri, dengan sistem yang berinteroperasi dengannya, dan dengan manusia yang menggunakaanya. Beberapa pedoman umum dalam desain *interface* adalah :

1. Konsisten, menggunakan format yang konsisten untuk pemilihan menu, perintah *input*, tampilan, dan fungsi lainnya dalam interaksi manusia dan komputer.
2. Berikan umpan balik yang berarti kepada pemakai untuk memastikan bahwa ada komunikasi dua arah (antara *interface* dan pemakai).
3. Mintalah verifikasi terhadap sembarang aksi destruktif yang signifikan, misalnya konfirmasi penghapusan *file*.
4. Usahakan adanya efisiensi dalam dialog, gerakan, dan pemikiran, serta kurangi jumlah informasi yang harus diingat di antara aksi-aksi.
5. Jangan membanjiri pemakai dengan data, gunakan format representasi yang memungkinkan asimilasi informasi yang cepat.
6. Gunakan huruf besar dan kecil, indentasi, dan pengelompokan teks untuk membantu pemahaman.
7. Nonaktifkan perintah yang tidak sesuai di dalam konteks aksi yang sedang berlangsung.

## 3.4 *Unified Modeling Language* (UML)

*Unified Modeling Language* (UML) adalah keluarga notasi grafis yang didukung oleh meta-model tunggal yang membantu pendeskripsikan dan desain perangkat lunak, khususnya sistem yang dibangun menggunakan pemrograman berorientasi objek (OO). (Fowler, 2003).

Bangunan dasar metodologi UML menggunakan tiga bangunan dasar untuk mendeskripsikan sistem/ perangkat lunak yang akan dikembangkan, yaitu *things*, *relationship*, dan diagram (Nugroho, 2005).

### 3.4.1 *Things*

Things adalah bangunan dasar berbasis objek yang utama dari UML. *Things* dalam UML terdiri dari 4 macam, yaitu:

1. *Structural things*

*Structural things* merupakan bagian yang relatif statis dalam model UML. Bagian yang relatif statis dapat berupa elemen-elemen yang bersifat fisik maupun konseptual (Nugroho, 2005).

1. *Behavioral things*

*Behavioral things* merupakan bagian yang dinamis pada model UML, biasanya merupakan kata kerja dari model UML, yang mencerminkan perilaku sepanjang ruang dan waktu (Nugroho, 2005).

1. *Grouping things*

*Grouping things* merupakan bagian pengorganisasi dalam UML. Dalam penggambaran model yang rumit kadang diperlukan penggambaran paket yang menyederhanakan model. Paket-paket ini kemudian dapat didekomposisi lebih lanjut. Paket berguna bagi pengelompokkan sesuatu, misalnya model-model dan subsistem-subsistem (Nugroho, 2005).

1. *Annotational things*

*Annotational things* merupakan bagian yang memperjelas model UML dan dapat berupa komentar-komentar yang menjelaskan fungsi serta ciri-ciri setiap elemen dalam model UML (Nugroho, 2005).

### 3.4.2 *Relationship*

*Relationship* merupakan bangunan dasar UML yang berfungsi sebagai penghubung antar-*things.* *Relationship* dalam UML terdiri dari 4 macam, yaitu:

1. Kebergantungan

Kebergantungan merupakan hubungan dimana perubahan yang terjadi pada suatu elemen mandiri (independen) akan mempengaruhi elemen yang bergantung padanya elemen yang tidak mandiri (independen) (Nugroho, 2005).

1. Asosiasi

Asosiasi merupakan hubungan antara objek satu dengan objek lainnya, bagaimana hubungan suatu objek dengan objek lainnya. Suatu bentuk asosiasi adalah agregasi yang menampilkan hubungan suatu objek dengan bagian-bagiannya (Nugroho, 2005).

1. Generalisasi

Generalisasi merupakan hubungan dimana objek anak (*descendent*) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (*ancestor*). Arah dari atas ke bawah dari objek induk ke objek anak dinamakan spesialisasi, sedangkan arah berlawanan sebaliknya dari arah bawah ke atas dinamakan generalisasi (Nugroho, 2005).

1. Realisasi

Realisasi merupakan operasi yang benar-benar dilakukan oleh suatu objek (Nugroho, 2005).

### 3.4.3 Diagram

Diagram adalah presentasi grafis dari sekumpulan elemen. Diagram merupakan kombinasi antara things dan relationship yang dibuat untuk memvisualisasikan sistem dari sejumlah perspektif yang berbeda, sehingga diagram merupakan proyeksi terhadap sistem. Ada beberapa macam diagram dalam UML, yaitu:

1. *Use Case* Diagram

*Use case* adalah abstraksi dari interaksi antara sistem dan aktor. *Use case* merupakan konstruksi untuk mendeskripsikan bagaimana sistem akan terlihat di mata pengguna. Sedangkan *use case* diagram memfasilitasi komunikasi antara analis dan pengguna serta antara analis dan klien (Nugroho, 2005).

1. *Class* Diagram

*Class* adalah dekripsi kelompok obyek-obyek dengan properti, perilaku (operasi) dan relasi yang sama, sehingga dengan adanya *class* diagram dapat memberikan pandangan global sebuah sistem (Nugroho, 2005).

1. *Activity* Diagram

*Activity* Diagram menggambarkan rangkaian aliran dari aktivitas. Digunakan untuk mendeskripsikan aktivitas yang dibentuk dalam suatu operasi sehingga dapat juga digunakan untuk aktivitas lainnya seperti *use case* atau interaksi (Nugroho, 2005).

## 3.5 Konsep *Object Oriented*

Pemrograman berorientasi objek (*object oriented programming*, OOP) memandang aplikasi perangkat lunak sebagai kumpulan objek yang saling berinteraksi di dalam sistem. Merancang sebuah aplikasi menggunakan teknik *object oriented* dilakukan dengan membagi fungsi-fungsi berdasar pembagian tanggung jawab. Pembagian tanggung jawab ditetapkan kepada setiap *class* yang dibuat. Setiap *class* menyediakan pelayanan untuk mengerjakan operasi tertentu. Operasi-operasi ini dilaksanakan oleh objek-objek yang dibuat dari *class* tersebut (Aziz, 2005).

Ada empat prinsip dasar yang utama dari *object oriented*, sebagai berikut:

1. Abstraksi

Abstraksi adalah pemodelan yang menyangkut aspek yang sangat utama, penting, khusus atau esensial dari sesuatu dengan mengabaikan rincian detail yang kurang atau tidak penting darinya. Hasilnya adalah menghilangkan yang rinci guna mendapatkan yang umum. Abstraksi mengatur kompleksitas dengan mengkonsentrasikan pada karakteristik yang esensial dari sebuah entitas yang membuatnya berbeda dari semua jenis entitas lainnya.

1. Enkapsulasi

Enkapsulasi adalah menyembunyikan cara pengimplementasian suatu benda dari pengguna, sehingga pengguna hanya tergantung dan berhubungan dengan antarmuka luarnya saja. Enkapsulasi sering disebut dengan "penyembunyian informasi". Hal ini memungkinkan pengguna mengoperasikan suatu sistem tanpa harus mengetahui cara/ mekanisme implementasi dari antarmukanya.

1. Modulariti

Modulariti adalah memecah sesuatu yang kompleks atau rumit menjadi bagian-bagian kecil yang dapat dikendalikan atau diatur. Penjelasan rinci dari modulariti adalah mengatur kompleksitas dengan memecah sesuatu yang besar atau kompleks ke dalam kelompok bagian-bagian kecil yang lebih mudah diatur. Bagian-bagian kecil tersebut kemudian dapat dibangun atau dibuat secara tersendiri dan tidak tergantung pada bagian lainnya.

1. Hirarki

Hirarki adalah urutan atau aturan dari tingkatan abstraksi menjadi seperti struktur pohon. Hirarki membuat pembentukan sesuatu berdasarkan kepada beberapa aturan yang khusus atau berdasarkan peringkat.

## 3.6 PHP (*Hypertext Preprocessor*)

PHP singkatan dari PHP: *Hypertext Preprocessor* yaitu bahasa pemrograman *web* *server*-*side* yang bersifat *open source*. PHP dibuat oleh Rasmus Lerdorf yang pertama kali diberi nama FI (*Form Interpreted*) dan digunakan untuk mengelola form dari *web*. Rasmus menulis dua versi pertama dari PHP. Namun, ia juga berpartisipasi dalam pengembangan versi-versi PHP selanjutnya yang dikembangkan oleh sebuah kelompok pengembang, termasuk di dalamnya Andi Gutmans dan Zeev Suraski yang selanjutnya mendirikan Zend Technologies (Anhar, 2010).

## 3.7 *Framework* Codeigniter

CodeIgniter adalah sebuah *web aplication framework* yang bersifat *open source* digunakan untuk membangun aplikasi PHP dinamis. Tujuan utama Codeigniter untuk membantu pengembang untuk mengerjakan aplikasi lebih cepat dari pada menulis semua code dari awal. CodeIgniter menyediakan berbagai macam *library* yang dapat mempermudah dalam pengembangan (Daqiqil, 2011).

CodeIgniter sendiri dibangun menggunakan konsep Model – *View* – *Controller* (MVC). MVC sebenarnya adalah teknik pemrograman yang memisahkan alur pikir (*businnes logic*), penyimpanan data (*data logic*) dan antarmuka aplikasi (*presentic logic*).

Adapun komponen-komponen MVC antara lain (Daqiqil, 2011):

1. Model

Model berhubungan dengan data dan interaksi ke database atau *web service*. Model juga merepresentasikan struktur data dari aplikasi yang bisa berupa basis data maupun data lain, misalnya dalam bentuk *file* teks, *file* XML maupun *web service*. Biasanya di dalam model akan berisi *class* dan fungsi untuk mengambil, melakukan *update* dan menghapus data *website*. Sebuah aplikasi *web* biasanya menggunakan basis data dalam menyimpan data, maka pada bagian Model biasanya akan berhubungan dengan perintah-perintah *query* SQL.

1. *View*

*View* berhubungan dengan segala sesuatu yang akan ditampilkan ke *end-user*. Bisa berupa halaman *web*, rss, javascript dan lain-lain. Kita harus menghindari adanya logika atau pemrosesan data di *view*. Di dalam *view* hanya berisi variabel-variabel yang berisi data yang siap ditampilkan. *View* dapat dikatakan sebagai halaman *website* yang dibuat dengan menggunakan HTML dan bantuan CSS atau JavaScript. *View* hanya dikhususkan untuk menampilkan data-data hasil dari model dan *controller*.

1. *Control*

*Controller* bertindak sebagai penghubung data dan *view*. Di dalam *controller* inilah terdapat *class-class* dan fungsi-fungsi yang memproses permintaan dari *View* ke dalam struktur data di dalam model. *Controller* juga tidak boleh berisi kode untuk mengakses basis data karena tugas mengakses data telah diserahkan kepada model. Tugas *controller* adalah menyediakan berbagai variabel yang akan ditampilkan pada *view*, memanggil model untuk melakukan akses ke basis data, menyediakan penanganan kesalahan (*error*), mengerjakan proses logika dari aplikasi serta melakukan validasi atau cek terhadap *input*.

## 3.8 DBMS MySQL

MySQL (*My Structure Query Language*) adalah sebuah perangkat lunak sistem manajemen basis data SQL atau DBMS (*Database Management System*) dari sekian banyak DBMS seperti Oracle, MS SQL, dan lain lain. MySQL merupakan DBMS yang *multithread*, *multi-user* yang bersifat gratis di bawah lisensi GNU *General Public License* (GPL). Tidak seperti Apache yang merupakan *software* yang dikembangkan oleh komunitas umum, dan hak cipta untuk kode sumber dimiliki oleh penulisnya masing masing. MySQL dimiliki dan disponsori oleh sebuah perusahaan Swedia, yaitu MySQL AB (Anhar, 2010)

Basis data sendiri adalah Himpunan kelompok data yang saling berhubungan dan diorganisasi sedemikian rupa agar dapat dimanfaatkan kembali dengan cepat dan mudah (Evadeva, 2007).

# BAB IV ANALISIS DAN PERANCANGAN

## 4.1 *Requirements Analysis* (Analisis Kebutuhan)

Analisis kebutuhan dilakukan untuk mengetahui prioritas kebutuhan sistem. Tujuannya adalah mengidentifikasi kebutuhan yang diinginkan dari sistem yang dibuat.

### 4.1.1 Deskripsi Kebutuhan Perangkat Lunak

Kebutuhan perangakat lunak merupakan kondisi atau kemampuan yang harus dimiliki oleh perangkat lunak. Berikut akan disajikan definisi kebutuhan perangkat lunak yang meliputi deskripsi umum perangkat lunak, kebutuhan perangkat lunak, kebutuhan fungsional, dan kebutuhan non fungsional.

### 4.1.2 Deskripsi Umum Perangkat Lunak

Aplikasi Dokumentasi Persuratan dimanfaatkan untuk mengelola dan mendokumentasikan data surat masuk serta surat keluar pada Balai Besar Perikanan Budidaya Air Payau Jepara. Aplikasi ini membantu mendokumentasikan persuratan secara efisien dan efektif, administrator dapat memasukkan, melihat detail, mengubah, menghapus, dan mengirim data surat masuk dan surat keluar dengan mudah. Selain itu, aplikasi ini memiliki tampilan yang *user friendly* sehingga administrator yang sudah berusia lanjut pun dapat memanfaatkan aplikasi ini dengan baik.

### 4.1.3 Karakteristik Pengguna

Aplikasi dokumentasi persuratan ini memiliki satu jenis pengguna yaitu Administrator seperti tertera pada tabel 4.1 berikut:

Tabel 4.1 Karakteristik Pengguna

|  |  |  |
| --- | --- | --- |
| No. | Aktor | Deskripsi |
| 1. | Administrator | Mengelola semua fungsi keseluruhan aplikasi, seperti memasukkan, mengubah, menampilkan *file*, menghapus data surat masuk serta surat keluar, dan mengubah *password* administrator. |

### 4.1.4 Kebutuhan Perangkat Lunak

Terdapat beberapa perangkat lunak yang dibutuhakan oleh Aplikasi Dokumentasi Persuratan, diantaranya:

1. Sistem Operasi

Sistem operasi minimal yang dibutuhkan adalah Windows XP.

1. *Web Browser*

*Web* *browser* digunakan untuk mengakses Aplikasi Dokumentasi Persuratan, rekomendasi Google Chrome dan Opera.

1. *Database Server*

*Database server* digunakan untuk mengelola *database*, *database* *server* yang digunakan adalah MySQL.

1. *Web* *Server*

*Web* *Server* dibutuhkan untuk menjalankan proses bisnis dan menerima permintaan pengguna, rekomendasi *web* *server* adalah Apache.

### 4.1.5 Kebutuhan Fungsional Aplikasi

Kebutuhan fungsional Aplikasi Dokumentasi Persuratan dapat dilihat pada Tabel 4.2 berikut:

Tabel 4.2 Kebutuhan Fungsional

| No | SRS-ID | Deskripsi |
| --- | --- | --- |
| 1. | SRS-ADP-F-1 | Aplikasi memiliki fitur untuk administrator masuk ke dalam aplikasi dengan memvalidasi *username* dan *password* yang dimasukkan. |
| 2. | SRS-ADP-F-2 | Aplikasi dapat memasukkan data surat masuk dan surat keluar ke dalam *database* serta *file-*nya ke dalam *folder* yang ditentukan pada komputer. |
| 3. | SRS-ADP-F-3 | Aplikasi dapat menampilkan *file* surat masuk dan surat keluar. |
| 4. | SRS-ADP-F-4 | Aplikasi dapat mengubah data surat masuk dan surat keluar. |
| 5. | SRS-ADP-F-5 | Aplikasi dapat menghapus data surat masuk dan surat keluar. |
| 6. | SRS-ADP-F-6 | Aplikasi dapat menyediakan fasilitas untuk mengubah *password* bagi administrator. |

### 4.1.6 Kebutuhan Non Fungsional Sistem

Kebutuhan non fungsional Aplikasi Dokumentasi Persuratan dapat dilihat pada Tabel 4.3 berikut:

Tabel 4.3 Kebutuhan Non Fungsional

|  |  |  |
| --- | --- | --- |
| No | SRS-ID | Deskripsi |
| 1. | SRS-ADP-NF-1 | Javascript pada *web* *browser* harus ada dan aktif. |

## 4.2 *Problem Analysis* (Analisis Permasalahan)

Analisis permasalahan dilakukan terhadap pengguna (*user*) serta fungsi-fungsi utama Aplikasi Dokumentasi Persuratan. Dari analisis yang dilakukan, permasalahan-permasalahan yang ada ditransformasikan ke dalam daftar *use case* aplikasi yang akan dibangun.

### 4.2.1 Daftar *Use Case*

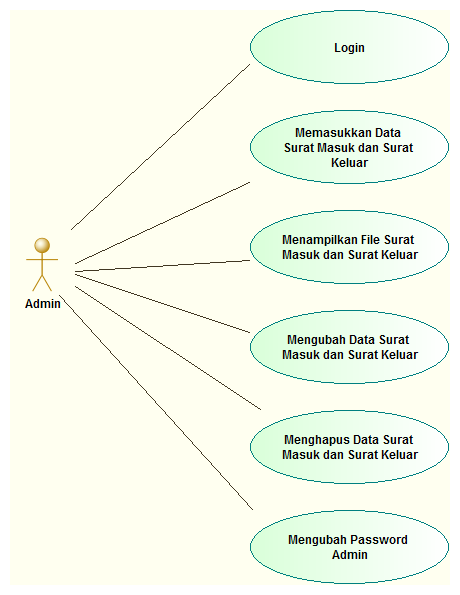
Aplikasi Dokumentasi Persuratan memiliki daftar *use case* skenario seperti yang tertera pada Tabel 4.4 berikut:

Tabel 4.4 Daftar Use Case

| No | USE CASE-ID | *Use Case* | Deskripsi | Aktor |
| --- | --- | --- | --- | --- |
| 1. | UC–ADP–1 | Melakukan *login* | Administrator dapat masuk ke dalam aplikasi dengan mengisikan *username* dan *password* yang ada pada form secara benar. | Administrator |
| 2. | UC–ADP–2 | Memasukkan data surat masuk dan surat keluar | Administrator dapat memasukkan data surat masuk dan surat keluar ke dalam *database* serta *file*-nya ke dalam *folder* yang ditentukan pada komputer. | Administrator |
| 3. | UC–ADP–3 | Menampilkan *file* surat masuk dan surat keluar | Administrator dapat menampilkan *file* surat masuk dan surat keluar. | Administrator |
| 4. | UC–ADP–4 | Mengubah data surat masuk dan surat keluar | Administrator dapat mengubah data surat masuk dan surat keluar. | Administrator |
| 5. | UC–ADP–5 | Menghapus data surat masuk dan surat keluar | Administrator dapat menghapus data surat masuk dan surat keluar. | Administrator |
| 6. | UC–ADP–6 | Mengubah *password* administrator | Administrator dapat mengubah dan reset *password*-nya sendiri | Administrator |

### 4.2.2 Pemodelan *Use case* Diagram

*Use case* diagram terbentuk dari serangkaian aktor dan *use case* yang ada pada sistem. Dari daftar aktor dan daftar *use case* tersebut diperoleh *use case* diagram dari Aplikasi Dokumentasi Persuratan seperti yang ditunjukkan oleh Gambar 4.1 dan tertera di lampiran.



Gambar 4.1 *Use* *Case diagram* Aplikasi Dokumentasi Persuratan

### 4.2.3 *Use case* Detail

*Use case* detail merupakan penjelasan dari masing-masing *use case*. *Use case* detail meliputi nomor, nama, aktor yang berinteraksi, kondisi awal, skenario utama, kondisi akhir, skenario alternatif, dan sketsa antarmuka pada masing-masing *use case*.

1. Melakukan Login

Tabel 4.5 *Use case* Detail untuk Melakukan *Login*

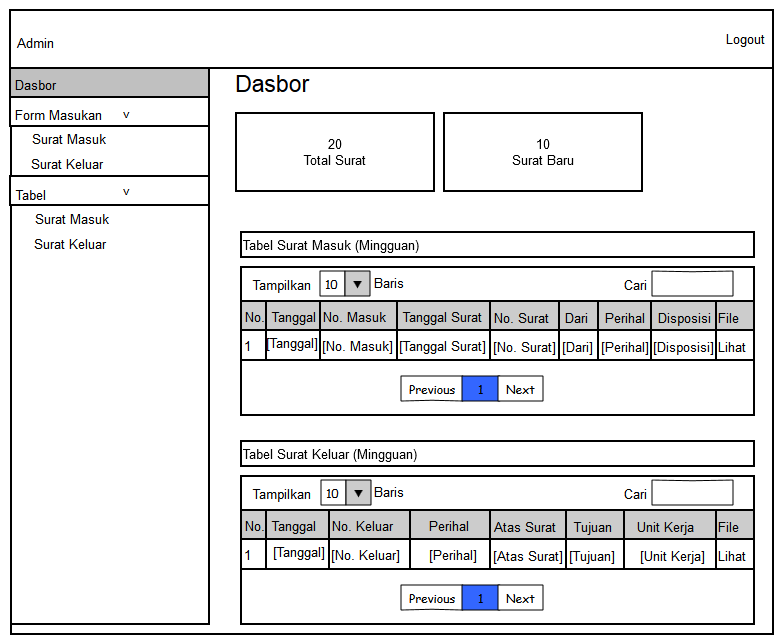
|  |  |
| --- | --- |
| *Use Case*: *Login* | |
| ID | UC-ADP-1 |
| *Actors* | Administrator |
| *Preconditions* |  |
| *Flow of Events* | 1. Aplikasi menampilkan form *login* yang berisi masukan *username* dan *password*.  2. Administrator mengisi form *login* dengan benar, kemudian menekan tombol *login*. |
| *Postconditions* | Aplikasi menampilkan halaman Dasbor dan Administrator telah berhasil *login* ke dalam aplikasi. |

Berikut ini merupakan sketsa antarmuka dari *use case* melakukan *login* Aplikasi Dokumentasi Persuratan yang dapat dilihat pada gambar 4.2 dan 4.3

1. Gambar 4.2 merupakan sketsa antarmuka *login* berfungsi untuk melakukan masuk ke aplikasi.
2. Gambar 4.3 merupakan sketsa antarmuka masuk ke sistem untuk berfungsi untuk tampilan awal aplikasi.



Gambar 4.2 Sketsa Halaman *Form Login*



Gambar 4.3 Sketsa Halaman Dasbor

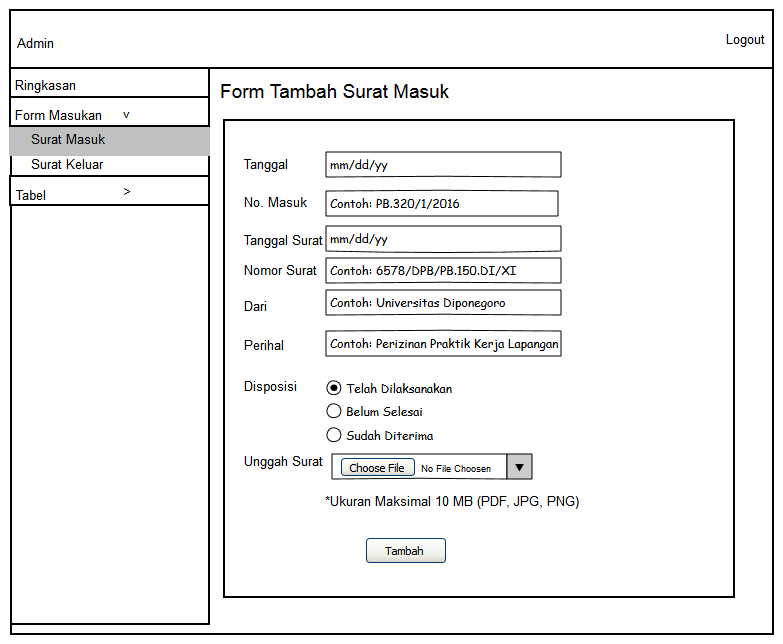
1. Memasukkan Data Surat Masuk dan Surat Keluar

Tabel 4.6 *Use case* Detail Memasukkan Data Surat Masuk dan Surat Keluar

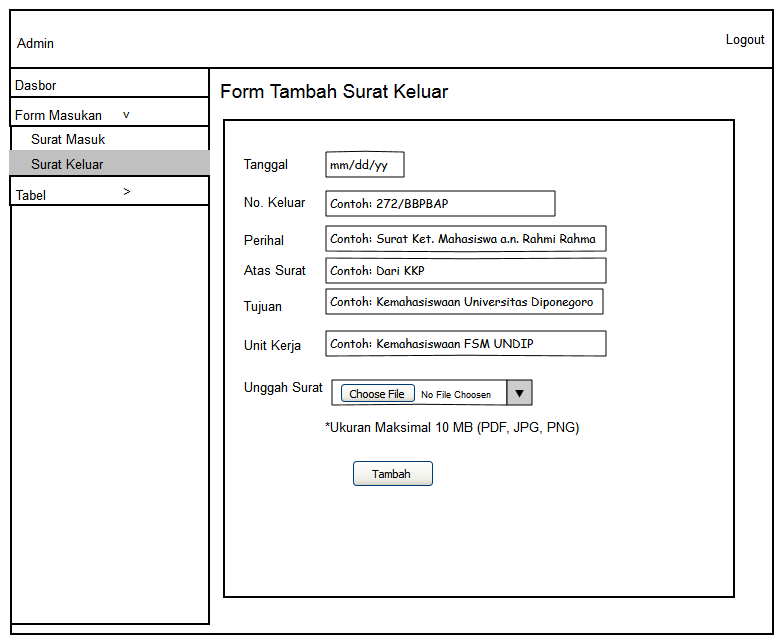
| *Use Case*: Memasukkan Data Surat Masuk dan Surat Keluar | |
| --- | --- |
| ID | UC-ADP-2 |
| *Actors* | Administrator |
| *Preconditions* | Administrator telah berhasil *login* ke dalam aplikasi |
| *Flow of Events* | 1. Administrator mengklik menu form masukan lalu muncul menu *drop down*, lalu mengklik salah satu menu surat masuk atau surat keluar.  2. Aplikasi menampilkan halaman form masukan surat masuk atas surat keluar tergantung dari menu yang dipilih administrator.  3. Administrator mengisi semua *field* pada form yang disediakan dan mengklik tombol tambah bila selesai mengisi semua *field*. |
| *Postconditions* | Data surat masuk atau surat keluar berhasil ditambahkan ke dalam aplikasi dan disimpan di dalam *database* serta diunggah secara otomatis ke dalam *folder* surat masuk atau surat keluar di dalam komputer. |

Berikut ini merupakan sketsa antarmuka dari *use case* memasukkan data surat masuk dan surat keluar Aplikasi Dokumentasi Persuratan yang dapat dilihat pada gambar 4.4 dan 4.5

* 1. Gambar 4.4 merupakan sketsa antarmuka form masukan surat masuk.
  2. Gambar 4.5 merupakan sketsa antarmuka form masukan surat keluar.



Gambar 4.4 Sketsa Halaman Form Masukan Surat Masuk



Gambar 4.5 Sketsa Halaman Form Masukan Surat Keluar

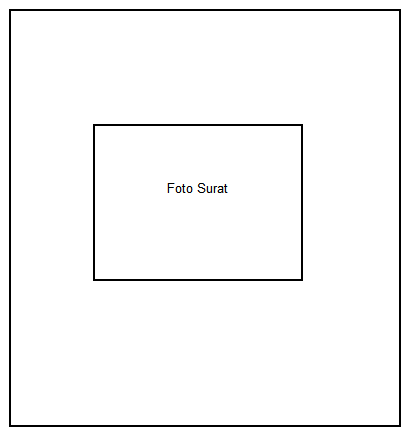
1. Menampilkan File Surat Masuk dan Surat Keluar

Tabel 4.7 *Use case* Detail Menampilkan *File* Surat Masuk dan Surat Keluar

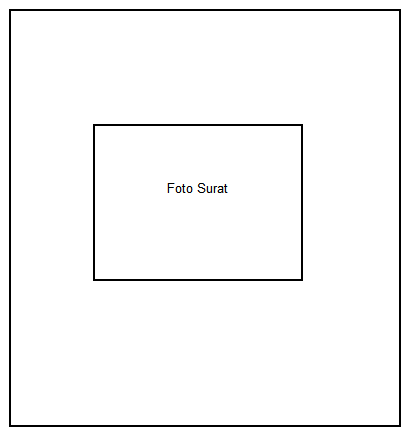
|  |  |
| --- | --- |
| *Use Case*: Menampilkan *File* Surat Masuk dan Surat Keluar | |
| ID | UC-ADP-2 |
| *Actors* | Administrator |
| *Preconditions* | Administrator telah berhasil *login* ke dalam aplikasi |
| *Flow of Events* | 1. Administrator mengklik menu tabel surat, lalu muncul menu *drop down*, lalu mengklik tabel daftar surat masuk atau surat keluar.  2. Aplikasi menampilkan halaman tabel daftar surat masuk atau surat masuk tergantung dari menu yang dipilih administrator.  3. Administrator mengklik link “Lihat” pada salah satu surat.  4. Aplikasi menampilkan halaman *file* surat. |
| *Postconditions* |  |

Berikut ini merupakan sketsa antarmuka dari *use case* menampilkan *file* surat masuk dan surat keluar Aplikasi Dokumentasi Persuratan yang dapat dilihat pada gambar 4.6 dan 4.7

* 1. Gambar 4.6 merupakan sketsa antarmuka halaman *file* surat masuk.
  2. Gambar 4.7 merupakan sketsa antarmuka halaman *file* surat keluar.



Gambar 4.6 Sketsa antarmuka halaman *file* surat masuk



Gambar 4.7 Sketsa antarmuka halaman *file* surat keluar

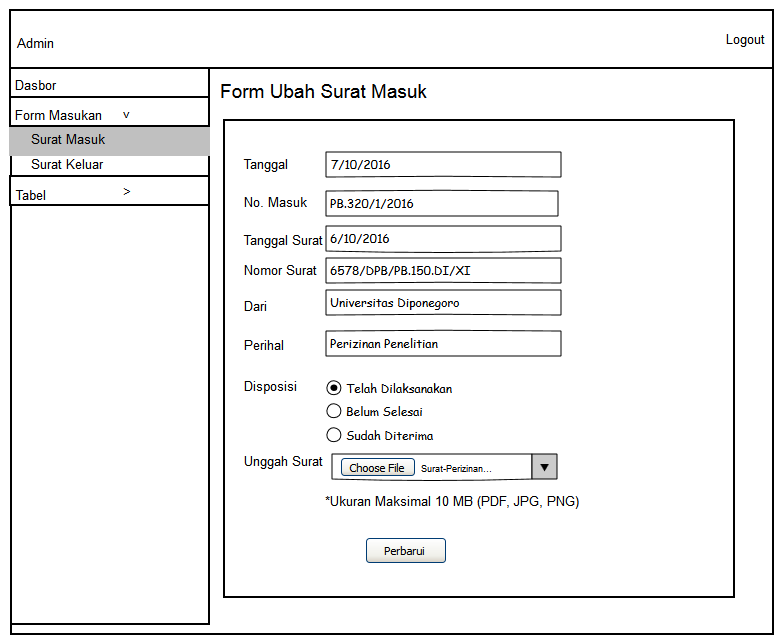
1. Mengubah Data Surat Masuk dan Surat Keluar

Tabel 4.8 *Use case* Detail Mengubah Data Surat Masuk dan Surat Keluar

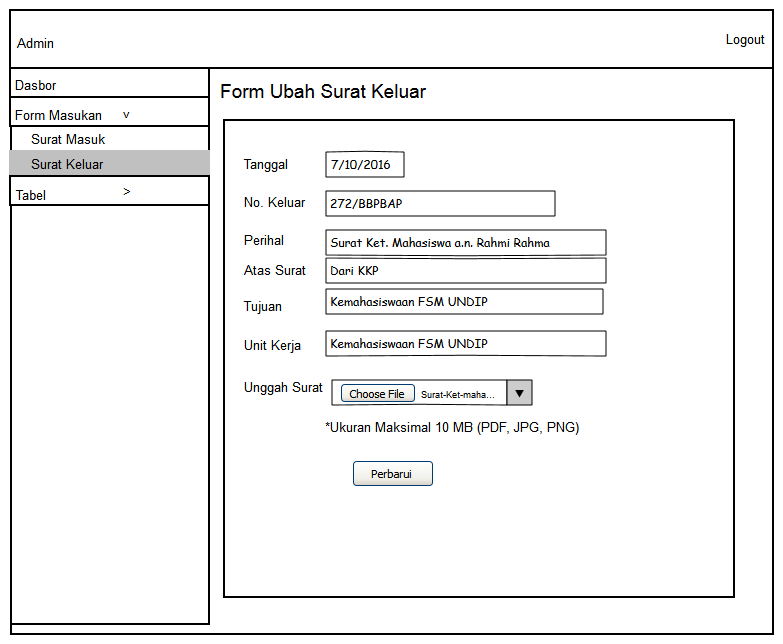
|  |  |
| --- | --- |
| *Use Case*: Mengubah Data Surat Masuk dan Surat Keluar | |
| ID | UC-ADP-4 |
| *Actors* | Administrator |
| *Preconditions* | Administrator telah berhasil *login* ke dalam aplikasi |
| *Flow of Events* | 1. Administrator mengklik menu “Tabel Surat” lalu muncul menu *drop down*, lalu mengklik salah satu menu “daftar surat masuk” atau “daftar surat keluar”.  2. Aplikasi menampilkan halaman daftar surat masuk atau surat keluar tergantung dari menu yang dipilih administrator.  3. Administrator mengklik link “Ubah”.  4. Aplikasi menampilkan halaman form ubah data surat masuk atau surat keluar.  5. Administrator mengubah data surat masuk atau surat keluar, lalu mengklik tombol “Perbarui”. |
| *Postconditions* | Data surat masuk atau surat keluar berhasil diubah dan disimpan di dalam database serta diunggah secara otomatis ke dalam *folder* surat masuk atau surat keluar di dalam komputer. |

Berikut ini merupakan sketsa antarmuka dari *use case* mengubah data surat masuk dan surat keluar Aplikasi Dokumentasi Persuratan yang dapat dilihat pada gambar 4.8 dan 4.9.

* 1. Gambar 4.8 merupakan sketsa antarmuka form ubah surat masuk.
  2. Gambar 4.9 merupakan sketsa antarmuka form ubah surat keluar.



Gambar 4.8 Sketsa Halaman Form Ubah Surat Masuk



Gambar 4.9 Sketsa Halaman Form Ubah Surat Keluar

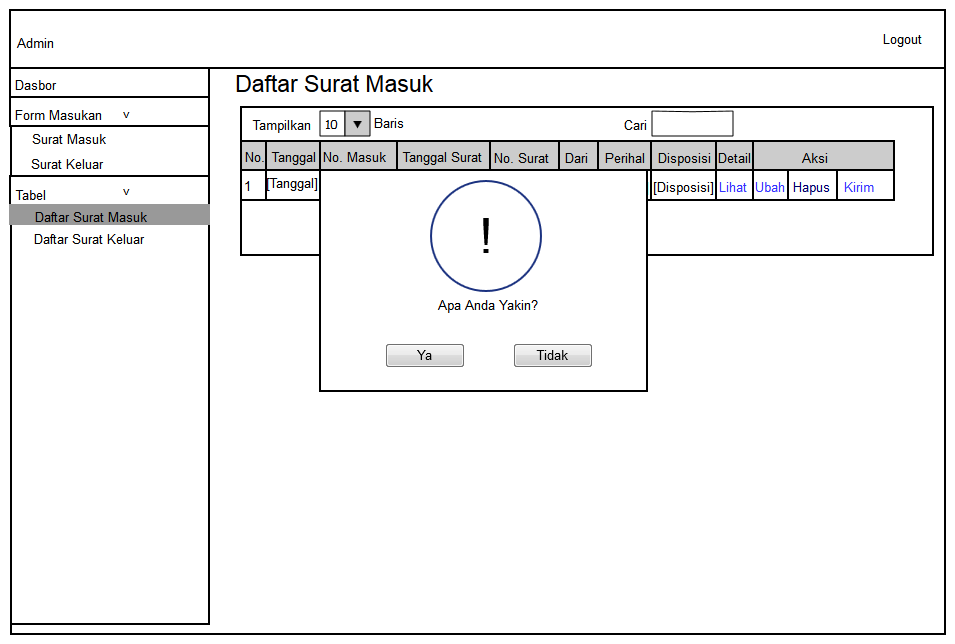
1. Menghapus Data Surat Masuk dan Surat Keluar

Tabel 4.9 *Use Case* Detail Menghapus Data Surat Masuk dan Surat Keluar

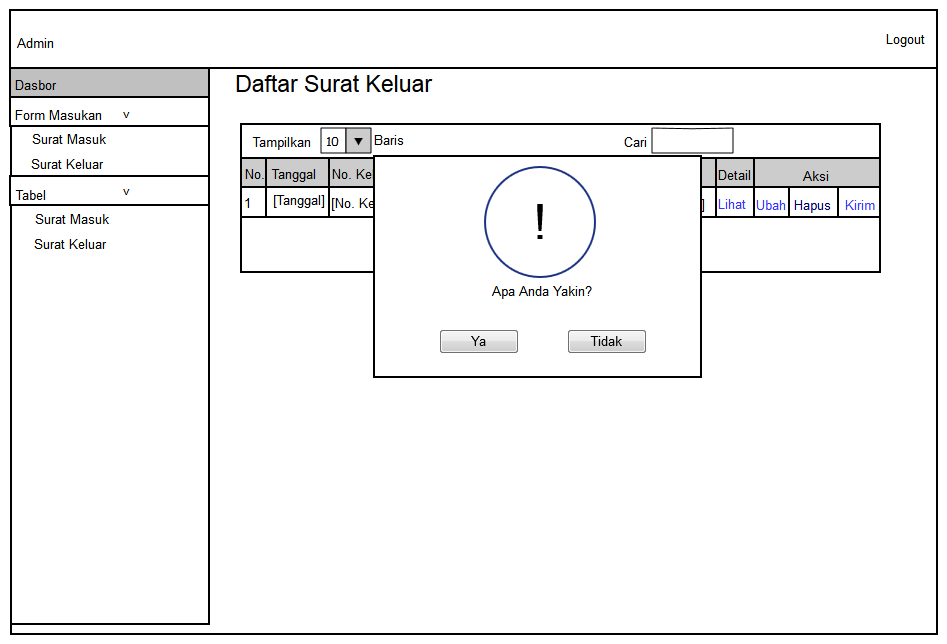
|  |  |
| --- | --- |
| *Use Case*: Menghapus Data Surat Masuk dan Surat Keluar | |
| ID | UC-ADP-5 |
| *Actors* | Administrator |
| *Preconditions* | Administrator telah berhasil *login* ke dalam aplikasi |
| *Flow of Events* | 1. Administrator mengklik menu tabel surat, lalu muncul menu *drop down*, lalu mengklik tabel daftar surat masuk atau surat keluar.  2. Aplikasi menampilkan halaman tabel daftar surat masuk atau surat masuk tergantung dari menu yang dipilih administrator.  3. Administrator mengklik link “Hapus” pada salah satu surat.  4. Aplikasi menampilkan alert dengan pesan “Apa Anda Yakin?” dan tombol “Iya“ serta “Tidak”.  5. Administrator mengklik tombol “Iya”.  6. Aplikasi menghapus satu record data pada tabel.  7. Aplikasi menampilkan alert “surat berhasil dihapus”. |
| *Postconditions* | Satu *record* data surat masuk atau surat keluar berhasil dihapus dari aplikasi dan database serta *file* yang sebelumnya diunggah juga dihapus dari *folder* surat masuk atau surat keluar. |

Berikut ini merupakan sketsa antarmuka dari *use case* menghapus data surat masuk dan surat keluar Aplikasi Dokumentasi Persuratan yang dapat dilihat pada gambar 4.10 dan 4.11

* 1. Gambar 4.10 merupakan sketsa antarmuka halaman daftar surat masuk dengan alert menghapus.
  2. Gambar 4.11 merupakan sketsa antarmuka halaman daftar surat keluar dengan alert menghapus.



Gambar 4.10 Sketsa halaman daftar surat masuk dengan *alert* menghapus



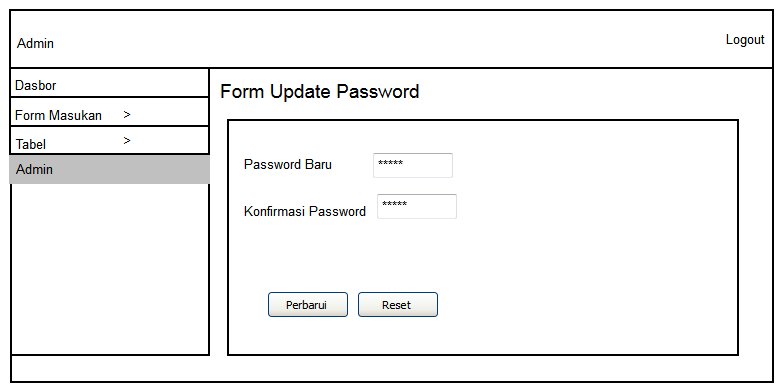
Gambar 4.11 Sketsa halaman daftar surat keluar dengan *alert* menghapus

1. Mengubah *Password* Administrator

Tabel 4.10 *Use Case* Detail Mengubah *Password* Administrator

|  |  |
| --- | --- |
| *Use Case*: Mengubah *Password* Administrator | |
| ID | UC-ADP-6 |
| *Actors* | Administrator |
| *Preconditions* | Administrator telah berhasil *login* ke dalam aplikasi |
| *Flow of Events* | 1. Administrator mengklik menu administrator  2. Aplikasi menampilkan halaman form untuk ubah *password*.  3. Administrator mengklik tombol perbarui untuk mengubah. |
| *Postconditions* | *Password* administrator telah berubah |

Berikut ini merupakan sketsa antarmuka dari *use case* mengubah data surat masuk dan surat keluar Aplikasi Dokumentasi Persuratan yang dapat dilihat pada gambar 4.12



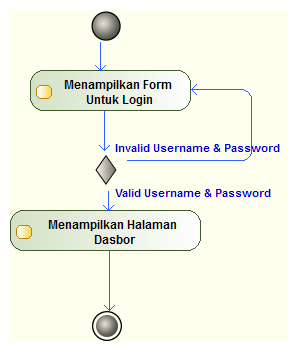
Gambar 4.12 Sketsa halaman *update password* administrator

### 4.2.4 *Activity* *Diagram*

*Activity Diagram* adalah sebuah diagram yang menggambarkan *workflow* (aliran kerja) berbagai aktivitas dari sebuah sistem, pengguna atau aliran sekuensial dari aktivitas-aktivitas tersebut. Perancangan *activity diagram* dalam dokumen desain Aplikasi Dokuementasi Persuratan Berbasis Web pada Balai Besar Perikanan Budidaya Air Payau (BBPBAP) Jepara menggunakan *Swimlane*. *Swimlane* adalah elemen visual yang digunakan dalam diagram alir proses yang menggambarkan siapa bekerja pada subset tertentu dari sebuah proses. Simbol *swimlane* membagi aktivitas alur kerja kedalam kelompok yang menunjukan pengguna yang akan menjalankan aktivitas didalamnya (Triandini, 2012 ).

1. Diagram Aktivitas *Login*

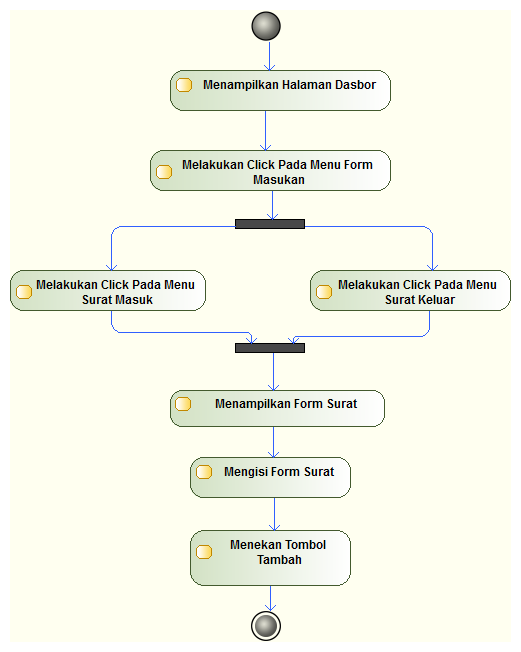
Diagram aktivitas untuk *login* menggambarkan aliran kerja bila ingin *login* ke dalam aplikasi. Mulanya ditampilkan halaman *login*, bila *username* dan *password* valid maka aktor akan *login* ke dalam aplikasi. Sebaliknya, bila *invalid* maka aplikasi akan terus menampilkan halaman *login*. Berikut diagram aktivitas yang ditunjukkan gambar 4.13 untuk *login*.



Gambar 4.13 Diagram Aktivitas *Login*

1. Diagram Aktivitas Memasukkan Data Surat Masuk dan Surat Keluar

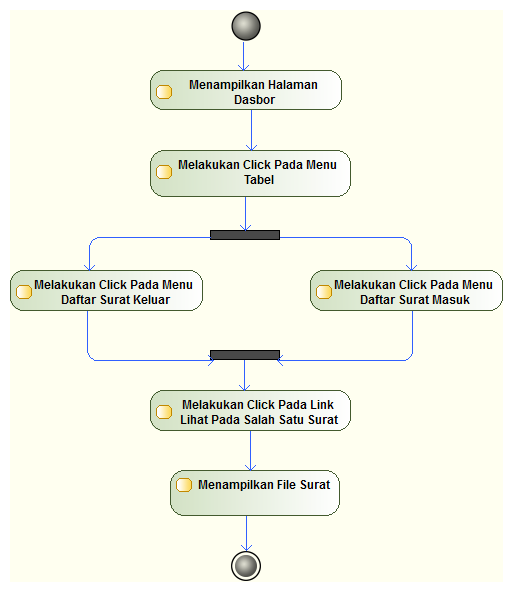
Aktivitas ini mengharuskan aktor untuk *login* ke dalam aplikasi terlebih dahulu. Setelah itu, aplikasi menyediakan fasilitas form masukan data surat masuk dan surat keluar. Berikut pemodelan diagram aktivitas yang ditunjukkan pada gambar 4.14.



Gambar 4.14 Diagram Aktivitas Masukan Surat Masuk dan Surat Keluar

1. Diagram Aktivitas Menampilkan *File* Surat Masuk dan Surat keluar

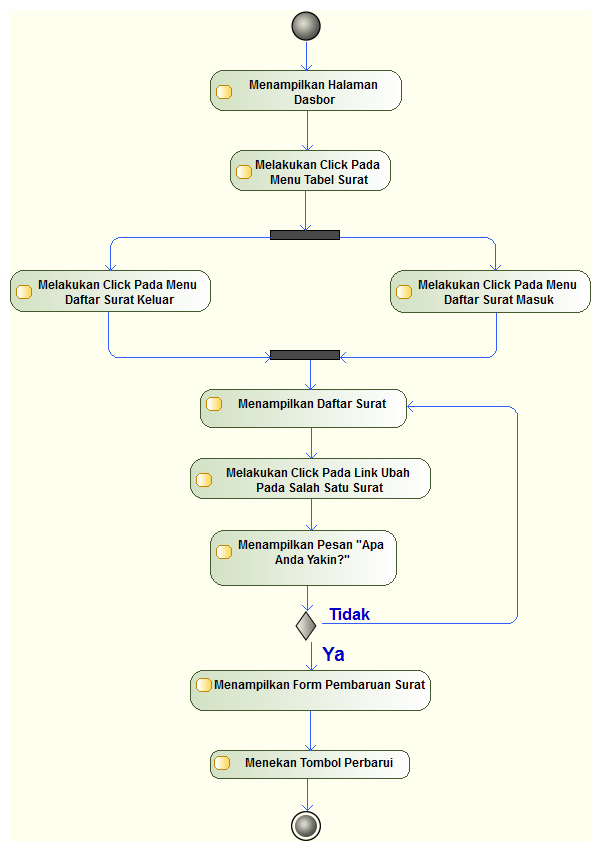
Aktivitas ini mengharuskan aktor untuk *login* ke dalam aplikasi. Aplikasi menyediakan fasilitas menampilkan *file* surat masuk dan surat keluar. Berikut adalah diagram aktivitasnya yang ditunjukkan gambar 4.15.



Gambar 4.15 Diagram Aktivitas Menampilkan *File* Surat Masuk dan Surat Keluar

1. Diagram Aktivitas Mengubah Data Surat Masuk dan Surat Keluar

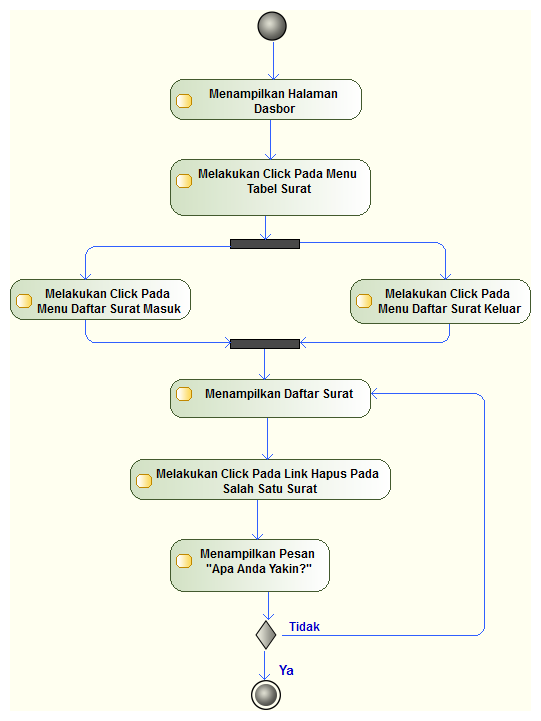
Aktivitas ini mengharuskan actor untuk *login* ke dalam aplikasi. Aplikasi menyediakan fasilitas ubah data surat masuk dan surat keluar yang ada. Berikut adalah diagram aktivitasnya yang ditunjukkan pada gambar 4.16.



Gambar 4.16 *Activity Diagram* Untuk Mengubah Data Surat Masuk dan Surat Keluar

1. Diagram Aktivitas untuk Menghapus Data Surat Masuk dan Surat Keluar

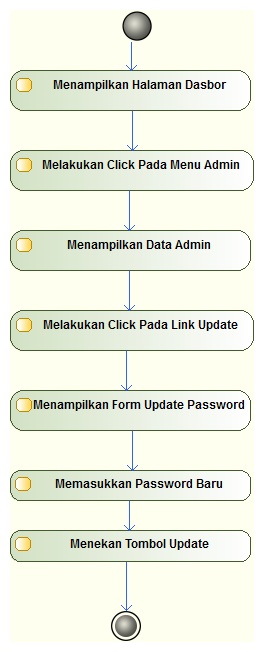
Aktivitas ini mengharuskan aktor untuk *login* ke dalam aplikasi. Aplikasi menyediakan fasilitas hapus data surat masuk dan surat keluar yang ada. Berikut adalah gambar 4.17 yang menunjukkan diagram aktivitasnya.



Gambar 4.17 Diagram Aktivitas Untuk Menghapus Surat Masuk dan Surat Keluar

1. Diagram Aktivitas Mengubah *Password* Administrator

Aktivitas ini mengharuskan aktor untuk *login* ke dalam aplikasi. Aplikasi menyediakan fasilitas mengubah *password* administrator. Berikut adalah gambar 4.18 yang menunjukkan diagram aktivitasnya.



Gambar 4.18 Diagram Aktivitas Untuk Mengubah *Password* Administrator

## 4.3 *Class Analysis*

Hasil identifikasi *class analysis* dijelaskan pada tabel 4.11 dan tanggung jawab serta atribut *class* dijelaskan pada tabel 4.12 dibawah ini:

Tabel 4.11 Identifikasi *Class Analysis*

| No. | Nama *Class Analysis* | Deskripsi |
| --- | --- | --- |
| 1. | administrator | *Entity* |
| 2. | suratmasuk | *Entity* |
| 3. | suratkeluar | *Entity* |
| 4. | login\_form | *Boundary* |
| 5. | index | *Boundary* |
| 6. | suratk\_form | *Boundary* |
| 7. | suratm\_form | *Boundary* |
| 8. | suratk\_table | *Boundary* |
| 9. | suratm\_table | *Boundary* |
| 10. | administrator\_view | *Boundary* |
| 11. | login\_controller | *Boundary* |
| 12. | form\_controller | *Boundary* |
| 13. | table\_controller | *Boundary* |

Tabel 4.12 Tanggung jawab dan Atribut *Class*

| No. | Nama *Class Analysis* | Deskripsi | DaftarAtribut |
| --- | --- | --- | --- |
| 1. | administrator | Menyimpan dan mengambil data administrator dari basis data | 1. Idadministrator  *2. username*  3*. password*  4. nama |
| 2. | suratmasuk | Menyimpan dan mengambil data surat masuk dari basis data | 1. idsuratm  2. tanggal  3. nomasuk  4. tanggalsurat  5. nosurat  6. dari  7. perihal  8. disposisi  9*. file* |
| 3. | suratkeluar | Menyimpan dan mengambil data surat keluar | 1. idsuratk  2. nokeluar  3. tanggal  4. perihal  5. atassurat  6. tujuan  7. unitkerja  8. *file* |
| 4. | login\_form | Menampilkan formulir isian untuk kepeluan masuk ke dalam aplikasi |  |
| 5. | index | Menampilkan halaman awal dari aplikasi |  |
| 6. | suratk\_form | Menampilkan formulir untuk memasukkan data surat keluar |  |
| 7. | suratm\_form | Menampilkan formulir untuk memasukkan data surat masuk |  |
| 8. | suratk\_table | Menampilkan semua data surat keluar yang telah dimasukkan. |  |
| 9. | suratm\_table | Menampilkan semua data surat masuk yang telah dimasukkan. |  |
| 10. | administrator\_view | Menampilkan *username* dan *password* administrator |  |
| 11. | *login*\_controller | Mengelola proses *login* pada aplikasi |  |
| 12. | form\_controller | Mengelola proses masukkan dan *update* data surat masuk, surat keluar, dan administrator ke dalam basis data |  |
| 13. | table\_controller | Mengelola proses pengambilan data dari basis data dan menerima *request* dari aktor |  |

## 4.4 Perancangan

### 4.4.1 Realisasi *Use Case* Tahap Perancangan

Realisasi *use case* tahap perancangan dapat digambarkan dengan menggunakan *class diagram*. Berikut realisasi *use case* tahap perancangan untuk setiap *use case*:

1. *Login*

Tabel 4.13 merupakan hasil identifikasi *class* perancangan dari *class analysis* pada *use case* Login.

Tabel 4.13 Identifikasi *Class* Perancangan *Use Case* *Login*

|  |  |  |
| --- | --- | --- |
| *Stereotype* | *Class Analysis* | *Class* Perancangan |
| *Boundary* | login\_form | LoginFormUI |
| *Control* | login\_controller | LoginController |
| *Entity* | administrator | administrator |

1. Memasukkan Data Surat Masuk dan Surat Keluar

Tabel 4.14 merupakan hasil identifikasi class perancangan dari *class analysis* pada *use case* memasukkan data surat masuk dan surat keluar.

Tabel 4.14 Identifikasi *Class* Perancangan *Use Case* memasukkan data surat masuk dan surat keluar

| *Stereotype* | *Class Analysis* | *Class* Perancangan |
| --- | --- | --- |
| *Boundary* | suratk\_form | SuratkFormUI |
|  | suratm\_form | SuratmFormUI |
| *Control* | form\_controller | FormController |
| *Entity* | suratkeluar | suratkeluar |
|  | suratmasuk | suratmasuk |

1. Menampilkan *File* Surat Masuk dan Surat Keluar

Tabel 4.15 merupakan hasil identifikasi *class* perancangan dari *class analysis* pada *use case* menampilkan *file* surat masuk dan surat keluar.

Tabel 4.15 Identifikasi *Class* Perancangan *Use Case* menampilkan *file* surat masuk dan surat keluar

|  |  |  |
| --- | --- | --- |
| *Stereotype* | *Class Analysis* | *Class* Perancangan |
| *Boundary* | suratk\_table | SuratkTableUI |
|  | suratm\_table | SuratmTableUI |
| *Control* | table\_controller | TableController |
| *Entity* | suratkeluar | suratkeluar |
|  | suratmasuk | suratmasuk |

1. Mengubah Data Surat Masuk dan Surat Keluar

Tabel 4.16 merupakan hasil identifikasi *class* perancangan dari *class analysis* pada *use case* mengubah data surat masuk dan surat keluar.

Tabel 4.16 Identifikasi *Class* Perancangan *Use Case* mengubah data surat masuk dan surat keluar

|  |  |  |
| --- | --- | --- |
| *Stereotype* | *Class Analysis* | Class Perancangan |
| *Boundary* | suratk\_form | SuratkFormUI |
|  | suratm\_form | SuratmFormUI |
| *Control* | form\_controller | FormController |
| *Entity* | suratkeluar | suratkeluar |
|  | suratmasuk | suratmasuk |

1. Menghapus Data Surat Masuk dan Surat Keluar

Tabel 4.17 merupakan hasil identifikasi *class* perancangan dari *class analysis* pada *use case* memasukkan data surat masuk dan surat keluar.

Tabel 4. 17 Identifikasi *Class* Perancangan *Use Case* menghapus data surat masuk dan surat keluar

| *Stereotype* | *Class Analysis* | *Class* Perancangan |
| --- | --- | --- |
| *Boundary* | suratk\_table | SuratkTableUI |
|  | suratm\_table | SuratmTableUI |
| *Control* | table\_controller | TableController |
| *Entity* | suratkeluar | suratkeluar |
|  | suratmasuk | suratmasuk |
|  | suratmasuk | suratmasuk |

1. Mengubah *Password* Administrator

Tabel 4.18 merupakan hasil identifikasi *class* perancangan dari *class analysis* pada *use case* mengubah *password* administrator.

Tabel 4.18 Identifikasi *Class* Perancangan *Use Case* mengubah *password* administrator

| *Stereotype* | *Class Analysis* | *Class* Perancangan |
| --- | --- | --- |
| *Boundary* | administrator\_view | AdminViewUI |
| *Control* | table\_controller | TableController |
| *Entity* | administrator | administrator |

### 4.4.2 Perancangan *Database*

Perancangan *database* dari Aplikasi Dokumentasi Persuratan. Hasil *mapping* dari *class diagram entity* ke skema basis data relasional dapat dilihat pada tabel – tabel berikut

1. Tabel Administrator

Nama Tabel : administrator

Deskripsi : menyimpan data administrator

*Primary key* : Idadministrator

Tabel 4.19 Daftar Tabel Administrator

| *Id Field* | Deskripsi | Tipe | Keterangan |
| --- | --- | --- | --- |
| Idadministrator | Id Administrator | Int(2) | *Primary key* |
| *username* | Alamat Email Administrator | varchar(50) |  |
| *password* | Kata Sandi | varchar(50) |  |
| nama | Nama Administrator | varchar (50) |  |

1. Tabel Surat Masuk

Nama Tabel : suratmasuk

Deskripsi : menyimpan data suratmasuk

*Primary key* : idsuratm

Tabel 4.20 Daftar Tabel suratmasuk

| *Id Field* | Deskripsi | Tipe | Keterangan |
| --- | --- | --- | --- |
| idsuratm | Id surat masuk | Int(11) | *Primary key* |
| Tanggal | Tanggal surat diterima oleh BBPBAP Jepara | Date | *Default* hari dimasukkannya data surat ke dalam aplikasi |
| Nomasuk | Nomor masuk | varchar(20) |  |
| Tanggalsurat | Tanggal yang tertera pada surat | Date |  |
| Nosurat | Nomor surat | varchar(30) |  |
| Dari | Nama organisasi/ individu yang mengirimkan surat | varchar(150) |  |
| Perihal | Atas dasar apa surat tersebut dikirim | varchar(150) |  |
| Disposisi | Status disposisi surat, seperti telah dilaksanakan, belum selesai, sudah diterima | varchar(18) |  |
| File | Nama *file* surat yang diunggah | varchar(50) |  |

1. Tabel Surat Keluar

Nama Tabel : suratkeluar

Deskripsi : menyimpan data suratkeluar

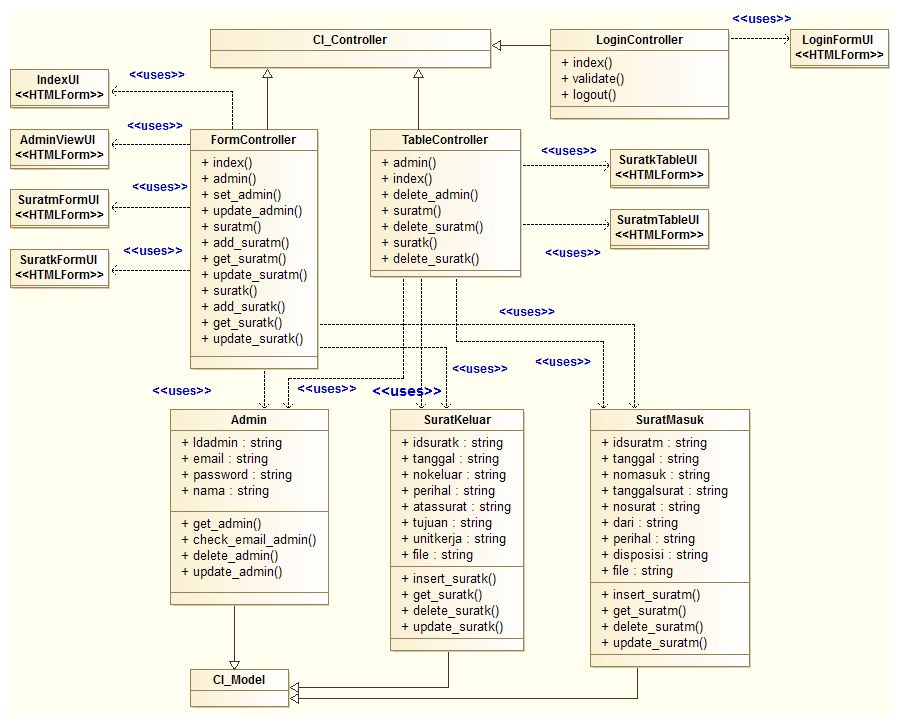
*Primary key* : idsuratk

Tabel 4.21 Daftar Tabel suratkeluar

| *Id Field* | Deskripsi | Tipe | Keterangan |
| --- | --- | --- | --- |
| idsuratk | Id surat keluar | Int(11) | *Primary key* |
| tanggal | Tanggal surat dikirim oleh BBPBAP Jepara | varchar(50) | *Default* hari dimasukkannya data surat ke dalam aplikasi |
| nokeluar | Kata Sandi | Date |  |
| perihal | Atas dasar apa surat tersebut dikirim | varchar (150) |  |
| atassurat | Menjawab surat berasal dari mana | varchar (50) |  |
| tujuan | Surat Organisasi/ individu yang dituju | varchar (150) |  |
| unitkerja | Unit kerja yang dituju | varchar (150) |  |
| *file* | Nama *file* yang diunggah | varchar (50) |  |

## 4.5 Pemodelan *Class Diagram*

Permodelan *class diagram* dapat dilihat pada gambar 4.19 dan tertera pada lampiran.



Gambar 4.19 *Class Diagram* Aplikasi Dokumentasi Persuratan

# BAB V IMPLEMENTASI DAN ANALISIS HASIL

Bab ini akan menjelaskan tentang pembahasan yang meliputi implementasi dan analisis hasil dari Aplikasi Dokumentasi Persuratan Berbasis *Web* Pada BBPBAP Jepara Jawa Tengah.

## 5.1 Spesifikasi Perangkat

Spesifikasi perangkat keras yang digunakan dalam membangun Aplikasi Dokumentasi Persuratan Berbasis *Web* Pada BBPBAP Jepara Jawa Tengah adalah sebagai berikut:

Notebook (Laptop) dengan spesifikasi:

1. *Processor* : Intel Core i3 2370M – 2.4 Ghz
2. *Memory* : 6 GB RAM
3. *Graphic* : Nvidia 610M
4. Papan Ketik (*Keyboard*)
5. Papan Sentuh (*Touchpad*)

Sedangkan perangkat lunak yang digunakan untuk membangun Aplikasi Dokumentasi Persuratan Berbasis *Web* Pada BBPBAP Jepara Jawa Tengah adalah sebagai berikut:

1. *Operating System* : Windows7 Proffesional 64 bit
2. *Web* *Server* : Apache 2.4.7
3. Bahasa Pemrograman : PHP Versi 5.5.8
4. *Database Management System* : MySQL 5.6.15
5. *Tools* : Notepad++
6. *Web* *Browser*  : Opera, Google Chrome, Mozilla Firefox

## 5.2 Implementasi Basis Data

Implementasi basis data merupakan transformasi desain data yang dihasilkan dari proses proses perancangan data menjadi suatu *database*. Dalam Aplikasi Dokumentasi Persuratan Berbasis *Web* Pada BBPBAP Jepara Jawa Tengah ini, *database* yang dibangun terdiri dari tiga tabel. Adapun *query* untuk membuat basis data dalam Aplikasi Dokumentasi Persuratan Berbasis *Web* Pada BBPBAP Jepara Jawa Tengah adalah sebagai berikut.

CREATE DATABASE persuratan;

Tabel-tabel dalam basis data yang dibuat dalam sistem manajemen basis data *MySQL* dengan nama ‘persuratan’ adalah sebagai berikut.

1. Tabel admin

Tabel admin merupakan tabel di dalam basis data yang berisikan informasi-informasi mengenai admin untuk keperluan login yang ada pada Balai Besar Perikanan Budidaya Air Payau (BBPBAP) Jepara. Tabel admin digunakan untuk menyimpan data admin. Adapun *query* untuk membuat tabel admin pada Aplikasi Dokumentasi Persuratan Berbasis *Web* Pada BBPBAP Jepara Jawa Tengah adalah sebagai berikut.

CREATE TABLE admin (

Idadmin int(2) NOT NULL AUTO INCREMENT,

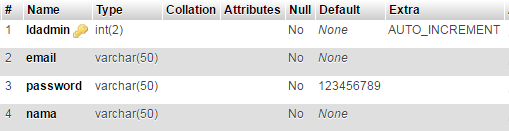
email varchar(50) NOT NULL,

password varchar(50) NOT NULL DEFAULT 123456789,

nama varchar(50) NOT NULL,

PRIMARY KEY (‘Idadmin’)

)



Gambar 5. 1 Implementasi Tabel admin

1. Tabel suratmasuk

Tabel suratmasuk merupakan tabel di dalam basis data yang berisikan informasi-informasi mengenai suratmasuk yang ada pada Balai Besar Perikanan Budidaya Air Payau (BBPBAP) Jepara. Tabel suratmasuk digunakan untuk menyimpan data suratmasuk. Adapun *query* untuk membuat tabel suratmasuk pada Aplikasi Dokumentasi Persuratan Berbasis *Web* Pada BBPBAP Jepara Jawa Tengah adalah sebagai berikut.

CREATE TABLE suratmasuk (

Idsuratm int(4) NOT NULL AUTO INCREMENT,

tanggal date NOT NULL,

nomasuk varchar(20) NOT NULL,

tanggalsurat date NOT NULL,

nosurat varchar(30) NOT NULL,

dari varchar(150) NOT NULL,

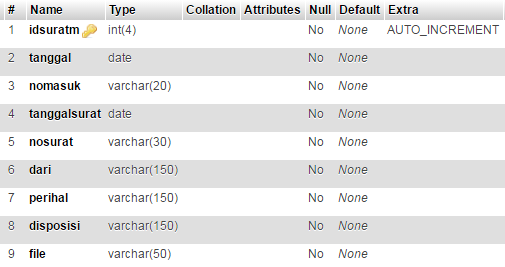
perihal varchar(150) NOT NULL,

disposisi varchar(150) NOT NULL,

file varchar(50) NOT NULL,

PRIMARY KEY (‘Idsuratm’)

)



Gambar 5.2 Implementasi Tabel suratmasuk

1. Tabel suratkeluar

Tabel suratkeluar merupakan tabel di dalam basis data yang berisikan informasi-informasi mengenai surat keluar yang ada pada Balai Besar Perikanan Budidaya Air Payau (BBPBAP) Jepara. Tabel suratkeluar digunakan untuk menyimpan data surat keluar. Adapun *query* untuk membuat tabel suratkeluar pada Aplikasi Dokumentasi Persuratan Berbasis *Web* Pada BBPBAP Jepara Jawa Tengah adalah sebagai berikut.

CREATE TABLE suratkeluar (

Idsuratk int(4) NOT NULL AUTO INCREMENT,

tanggal date NOT NULL,

nokeluar varchar(20) NOT NULL,

perihal varchar(150) NOT NULL,

atassurat varchar(50) NOT NULL,

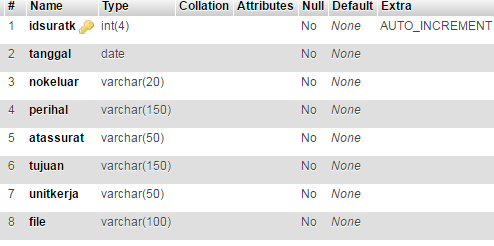
tujuan varchar(150) NOT NULL,

unitkerja varchar(50) NOT NULL,

file varchar(50) NOT NULL,

PRIMARY KEY (‘Idsuratm’)

)



Gambar 5.3 Implementasi Tabel suratkeluar

## 5.3 Implementasi *Activity Diagram*

Implementasi *activity diagram* pada Aplikasi Dokumentasi Persuratan Berbasis *Web* Pada BBPBAP Jepara Jawa Tengah merupakan realisasi dari Diagram aktivitasyang telah dibuat sebelumnya. Bahasa yang digunakan pada implementasi ini adalah bahasa pemrograman PHP yang ditulis sebagai runtutan fungsi-fungsi. Berikut adalah implementasi *activity diagram*:

1. Kode Program Melakukan *Login*

*Login* dilakukan oleh administrator maupun *user* yang memiliki *username* dan *password*. Jika cocok, maka administratordapat masuk ke dalam aplikasi.

function login($data=array()){

$email = $this->test\_input($data['email']);

$pass = $this->test\_input($data['password']);

$email = mysqli\_real\_escape\_string($this->get\_mysqli(),$email);

$pass = mysqli\_real\_escape\_string($this->get\_mysqli(),$pass);

$query = 'SELECT \* FROM admin WHERE email="'.$email.'" AND password="'.$pass.'"';

$result = $this->db->query($query);

if($result->num\_rows() > 0){

foreach($result->result\_array() as $row){

$data[] = $row;

}

return $data;

}

}

function get\_mysqli(){

$db = (array)get\_instance()->db;

return mysqli\_connect('localhost',$db['username'],$db['password'],

$db['database']);

}

1. Kode Program Memasukkan Data Surat Masuk dan Surat Keluar

Berikut adalah kode program yang ada pada suratm\_model dan suratk\_model secara berturut-turut.

//Surat Masuk

function insert\_suratm($data=array()){

$tanggal = $this->test\_input($data['tanggal']);

$nomasuk = $this->test\_input($data['nomasuk']);

$tanggalsurat = $this->test\_input($data['tanggalsurat']);

$nosurat = $this->test\_input($data['nosurat']);

$dari = $this->test\_input($data['dari']);

$perihal = $this->test\_input($data['perihal']);

$disposisi = $this->test\_input($data['disposisi']);

$file = $this->test\_input($data['file']);

$tanggal = mysqli\_real\_escape\_string($this->get\_mysqli(),$tanggal);

$nomasuk = mysqli\_real\_escape\_string($this->get\_mysqli(),$nomasuk);

$tanggalsurat = mysqli\_real\_escape\_string($this->get\_mysqli(),$tanggalsurat);

$nosurat = mysqli\_real\_escape\_string($this->get\_mysqli(),$nosurat);

$dari = mysqli\_real\_escape\_string($this->get\_mysqli(),$dari);

$perihal = mysqli\_real\_escape\_string($this->get\_mysqli(),$perihal);

$disposisi = mysqli\_real\_escape\_string($this->get\_mysqli(),$disposisi);

$file = mysqli\_real\_escape\_string($this->get\_mysqli(),$file);

$query = 'INSERT INTO suratmasuk (tanggal,nomasuk,tanggalsurat,nosurat,dari,perihal,disposisi,file)

VALUES ("'.$tanggal.'","'.$nomasuk.'",

"'.$tanggalsurat.'","'.$nosurat.'","'.$dari.'","'.$perihal.'","'.$disposisi.'","'.$file.'")';

$result = $this->db->query($query);

}

//Surat Keluar

function insert\_suratk($data=array()){

$tanggal = $this->test\_input($data['tanggal']);

$nokeluar = $this->test\_input($data['nokeluar']);

$tujuan = $this->test\_input($data['tujuan']);

$atassurat = $this->test\_input($data['atassurat']);

$perihal = $this->test\_input($data['perihal']);

$unitkerja = $this->test\_input($data['unitkerja']);

$file = $this->test\_input($data['file']);

$tanggal = mysqli\_real\_escape\_string($this->get\_mysqli(),$tanggal);

$nokeluar = mysqli\_real\_escape\_string($this->get\_mysqli(),$nokeluar);

$tujuan = mysqli\_real\_escape\_string($this->get\_mysqli(),$tujuan);

$atassurat = mysqli\_real\_escape\_string($this->get\_mysqli(),$atassurat);

$perihal = mysqli\_real\_escape\_string($this->get\_mysqli(),$perihal);

$unitkerja = mysqli\_real\_escape\_string($this->get\_mysqli(),$unitkerja);

$file = mysqli\_real\_escape\_string($this->get\_mysqli(),$file);

$query = 'INSERT INTO suratkeluar (tanggal,nokeluar,perihal,atassurat,tujuan,unitkerja,file)

VALUES

("'.$tanggal.'","'.$nokeluar.'","'.$perihal.'","'.$atassurat.'","'.$tujuan.'","'.$unitkerja.'","'.$file.'")';

$result = $this->db->query($query);

}

1. Kode Program untuk Menampilkan *File* Surat Masuk dan Surat Keluar

Berikut adalah kode program yang ada pada suratm\_tabel dan suratk\_tabel secara berturut-turut yang digunakan untuk menampilkan *file* surat.

foreach($suratm as $value){

echo '<tr>';

echo '<td style="text-align:center;">'.anchor('uploads/surat\_masuk/'.$value['file'].'','Lihat',array('target' => '\_blank')).'</td>';

foreach($suratk as $value){

echo '<tr>';

echo '<td style="text-align:center;">'.anchor('uploads/surat\_keluar/'.$value['file'].'','Lihat',array('target' => '\_blank')).'</td>';

1. Kode Program untuk Mengubah Data Surat Masuk dan Surat Keluar

Berikut adalah kode program yang ada pada suratm\_model dan suratk\_model secara berturut-turut yang digunakan untuk mengubah data surat masuk dan surat keluar.

function update\_suratm($data=array()){

$idsuratm = $this->test\_input($data['idsuratm']);

$tanggal = $this->test\_input($data['tanggal']);

$nomasuk = $this->test\_input($data['nomasuk']);

$tanggalsurat = $this->test\_input($data['tanggalsurat']);

$nosurat = $this->test\_input($data['nosurat']);

$dari = $this->test\_input($data['dari']);

$disposisi = $this->test\_input($data['disposisi']);

$file = $this->test\_input($data['file']);

$idsuratm = mysqli\_real\_escape\_string($this->get\_mysqli(),$idsuratm);

$tanggal = mysqli\_real\_escape\_string($this->get\_mysqli(),$tanggal);

$nomasuk = mysqli\_real\_escape\_string($this->get\_mysqli(),$nomasuk);

$tanggalsurat = mysqli\_real\_escape\_string($this->get\_mysqli(),$tanggalsurat);

$nosurat = mysqli\_real\_escape\_string($this->get\_mysqli(),$nosurat);

$dari = mysqli\_real\_escape\_string($this->get\_mysqli(),$dari);

$disposisi = mysqli\_real\_escape\_string($this->get\_mysqli(),$disposisi);

$file = mysqli\_real\_escape\_string($this->get\_mysqli(),$file);

$query = 'UPDATE suratmasuk SET tanggal="'.$tanggal.'",nomasuk="'.$nomasuk.'",

tanggalsurat="'.$tanggalsurat.'",nosurat="'.$nosurat.'",

dari="'.$dari.'",disposisi="'.$disposisi.'" ,file="'.$file.'"

WHERE idsuratm="'.$idsuratm.'"';

$result = $this->db->query($query);

}

$file = mysqli\_real\_escape\_string($this->get\_mysqli(),$file);

$query = 'UPDATE suratmasuk SET tanggal="'.$tanggal.'",nomasuk="'.$nomasuk.'",

tanggalsurat="'.$tanggalsurat.'",nosurat="'.$nosurat.'",

dari="'.$dari.'",disposisi="'.$disposisi.'" ,file="'.$file.'"

WHERE idsuratm="'.$idsuratm.'"';

$result = $this->db->query($query);

}

function update\_suratk($data=array()){

$idsuratk = $this->test\_input($data['idsuratk']);

$tanggal = $this->test\_input($data['tanggal']);

$nokeluar = $this->test\_input($data['nokeluar']);

$perihal = $this->test\_input($data['perihal']);

$atassurat = $this->test\_input($data['atassurat']);

$tujuan = $this->test\_input($data['tujuan']);

$unitkerja = $this->test\_input($data['unitkerja']);

$file = $this->test\_input($data['file']);

$idsuratk = mysqli\_real\_escape\_string($this->get\_mysqli(),$idsuratk );

$tanggal = mysqli\_real\_escape\_string($this->get\_mysqli(),$tanggal);

$nokeluar = mysqli\_real\_escape\_string($this->get\_mysqli(),$nokeluar);

$perihal = mysqli\_real\_escape\_string($this->get\_mysqli(),$perihal);

$atassurat = mysqli\_real\_escape\_string($this->get\_mysqli(),$atassurat);

$tujuan = mysqli\_real\_escape\_string($this->get\_mysqli(),$tujuan);

$unitkerja = mysqli\_real\_escape\_string($this->get\_mysqli(),$unitkerja);

$file = mysqli\_real\_escape\_string($this->get\_mysqli(),$file);

$query = 'UPDATE suratkeluar SET tanggal="'.$tanggal.'",nokeluar="'.$nokeluar.'",atassurat="'.$atassurat.'",

perihal="'.$perihal.'",tujuan="'.$tujuan.'",unitkerja="'.$unitkerja.'"

WHERE idsuratk="'.$idsuratk.'"';

$result = $this->db->query($query);

}

1. Kode Program untuk Menghapus Data Surat Masuk dan Surat Keluar

Berikut adalah kode program yang ada pada suratm\_model dan suratk\_model secara berturut-turut yang digunakan untuk mengubah data surat masuk dan surat keluar.

function delete\_suratm($id){

$this->load->view('suratm\_tabel');

$query = 'DELETE FROM suratmasuk WHERE idsuratm="'.$id.'"';

$query2 = 'ALTER TABLE suratmasuk AUTO\_INCREMENT=1';

$result = $this->db->query($query);

$result2 = $this->db->query($query2);

}

function delete\_suratk($id){

$this->load->view('suratk\_tabel');

$query = 'DELETE FROM suratkeluar WHERE idsuratk="'.$id.'"';

$query2 = 'ALTER TABLE suratkeluar AUTO\_INCREMENT=1';

$result2 = $this->db->query($query2);

$result = $this->db->query($query);

}

1. Kode Program untuk Mengubah *Password* Admin

Berikut adalah kode program yang ada pada suratm\_model dan suratk\_model secara berturut-turut yang digunakan untuk mengubah data surat masuk dan surat keluar.

function update\_admin($data=array()){

$password = $this->test\_input($data['admin\_pass1']);

$password = mysqli\_real\_escape\_string($this->get\_mysqli(),$password);

$query = 'UPDATE admin SET password="'.$password.'" WHERE Idadmin="'.$data['init\_id'].'"';

$result = $this->db->query($query);

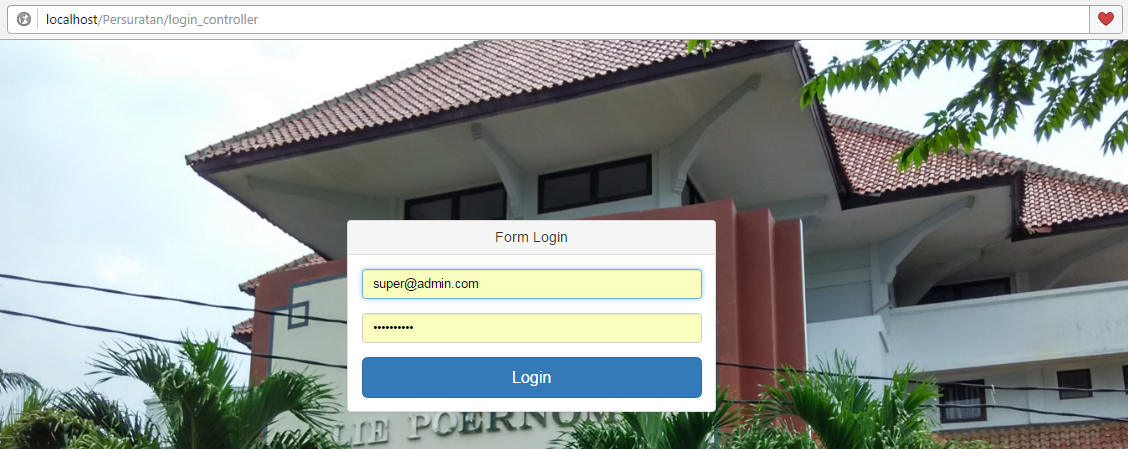
}

## 5.4 Implementasi Antarmuka

Aplikasi Dokumentasi Persuratan Berbasis *Web* Pada BBPBAP Jepara Jawa Tengah adalah sebuah aplikasi yang mempermudah administratoruntuk mengelola dan mendokumentasikan persuratan. Maka dari itu, aplikasi ini harus memiliki tampilan yang nyaman dan sesuai dengan karakteristik administrator. Implementasi antarmuka Aplikasi Dokumentasi Persuratan Berbasis *Web* Pada BBPBAP Jepara Jawa Tengah dapat dilihat pada Gambar 5.4 sampai dengan Gambar 5.19 berikut ini.

1. Halaman *Login*

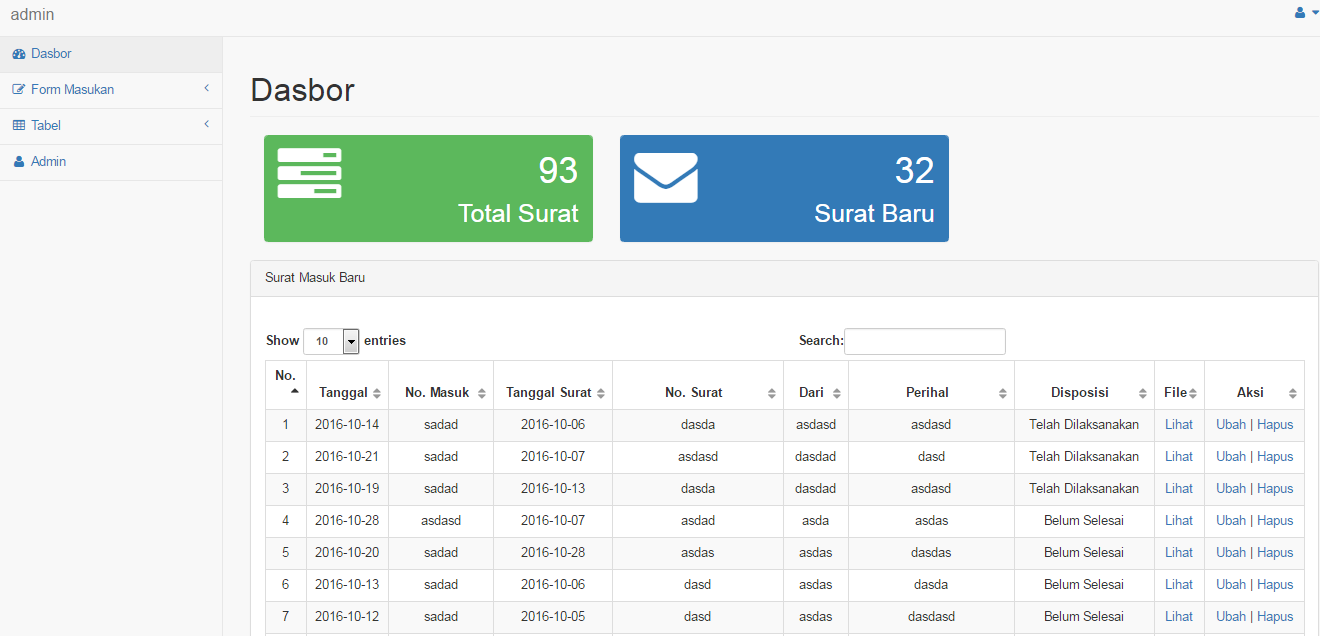
Saat pertama kali administrator mengakses aplikasi maka akan tampil halaman seperti berikut.



Gambar 5.4 Antarmuka Halaman *Login*

1. Halaman Dasbor

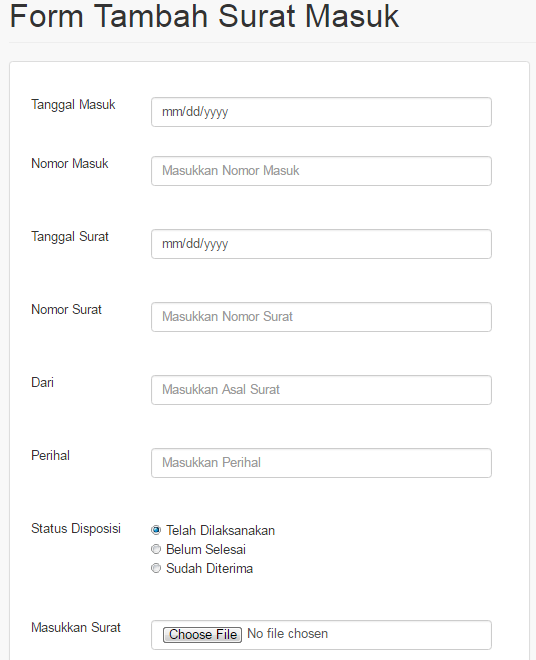
Halaman ini adalah halaman utama pada aplikasi. Halaman ini menampilkan informasi total surat yang ada dan total surat yang dimasukkan dalam seminggu terakhir.



Gambar 5.5 Halaman Dasbor

1. Halaman Form Masukan Surat Masuk

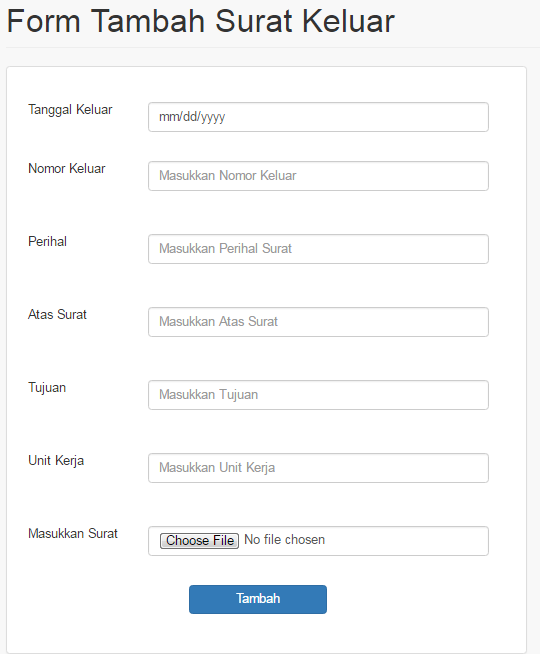
Halaman ini adalah halaman yang menampilkan form isian surat masuk. Pada halaman ini juga *file* surat masuk diunggah dan akan disimpan di dalam *folder* yang telah ditentukan sebelumnya pada komputer.



Gambar 5.6 Halaman Form Masukan Surat Masuk

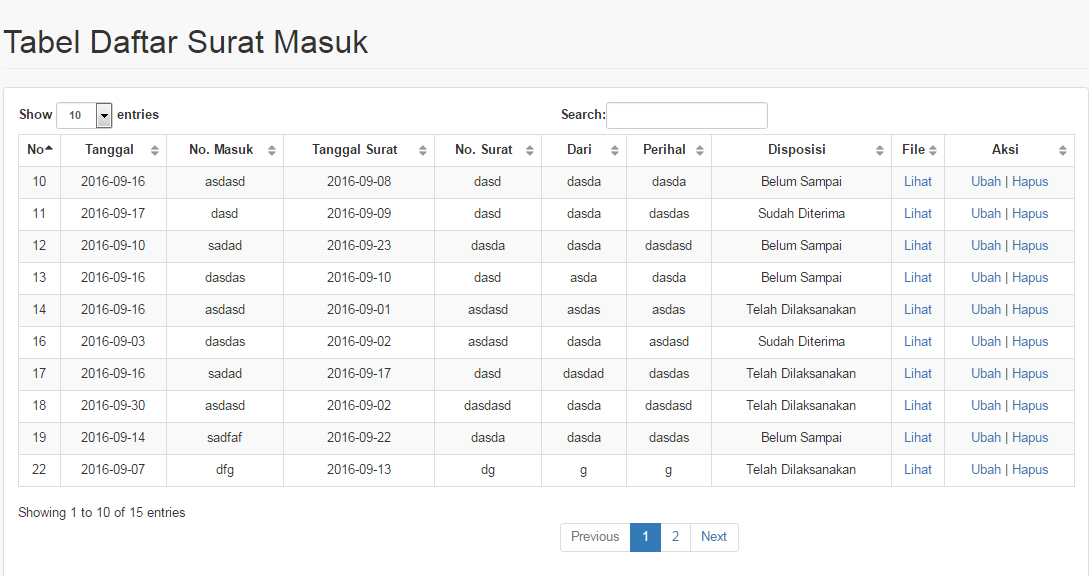
1. Halaman Form Masukan Surat Keluar

Halaman ini adalah halaman yang menampilkan form isian surat keluar. Pada halaman ini juga *file* surat masuk diunggah dan akan disimpan di dalam *folder* yang telah ditentukan sebelumnya pada komputer.



Gambar 5.7 Halaman Form Masukan Surat Keluar

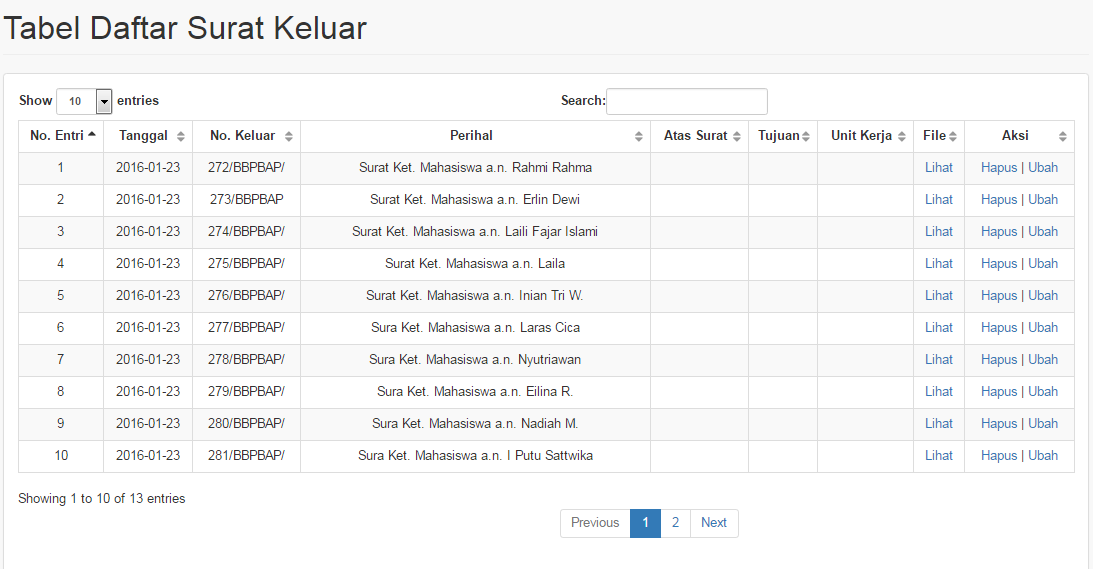
1. Halaman Tabel Daftar Surat Masuk

Halaman ini adalah halaman yang menampilkan seluruh daftar surat masuk. Pada halaman ini juga terdapat link “Lihat” untuk menampilkan *file* surat, “Ubah” untuk mengubah data surat, dan “Hapus” untuk menghapus data surat yang dipilih.

Gambar 5.8 Halaman Tabel Daftar Surat Masuk

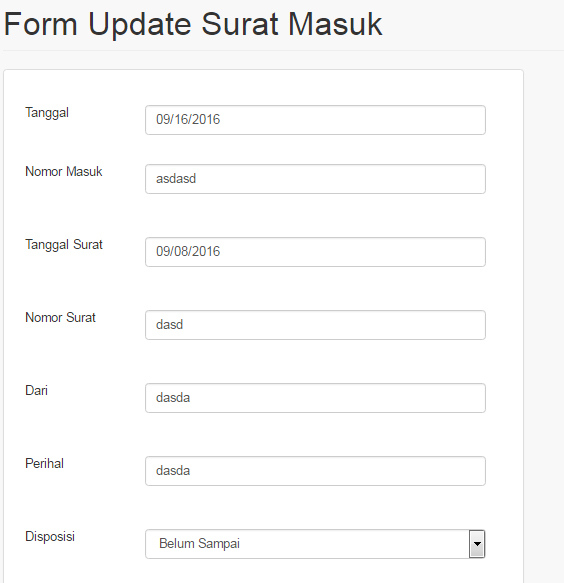
1. Halaman Tabel Daftar Surat Keluar

Halaman ini adalah halaman yang menampilkan seluruh daftar surat keluar. Pada halaman ini juga terdapat link “Lihat” untuk menampilkan *file* surat, “Ubah” untuk mengubah data surat, dan “Hapus” untuk menghapus data surat yang dipilih.



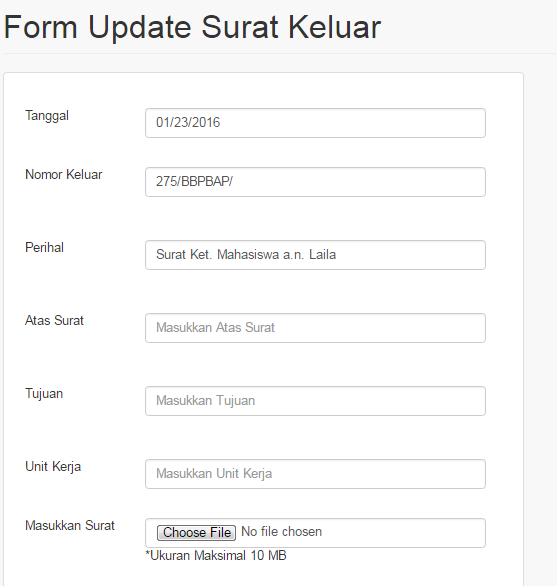
Gambar 5.9 Tabel Surat Keluar

1. Halaman Form *Update* Surat Masuk

Halaman ini adalah halaman yang menampilkan form isian untuk mengubah data surat masuk. Pada halaman ini, data yang sebelumnya disimpan ditampilkan sesuai dengan *field*-nya.

Gambar 5.10 *Update* Surat Masuk

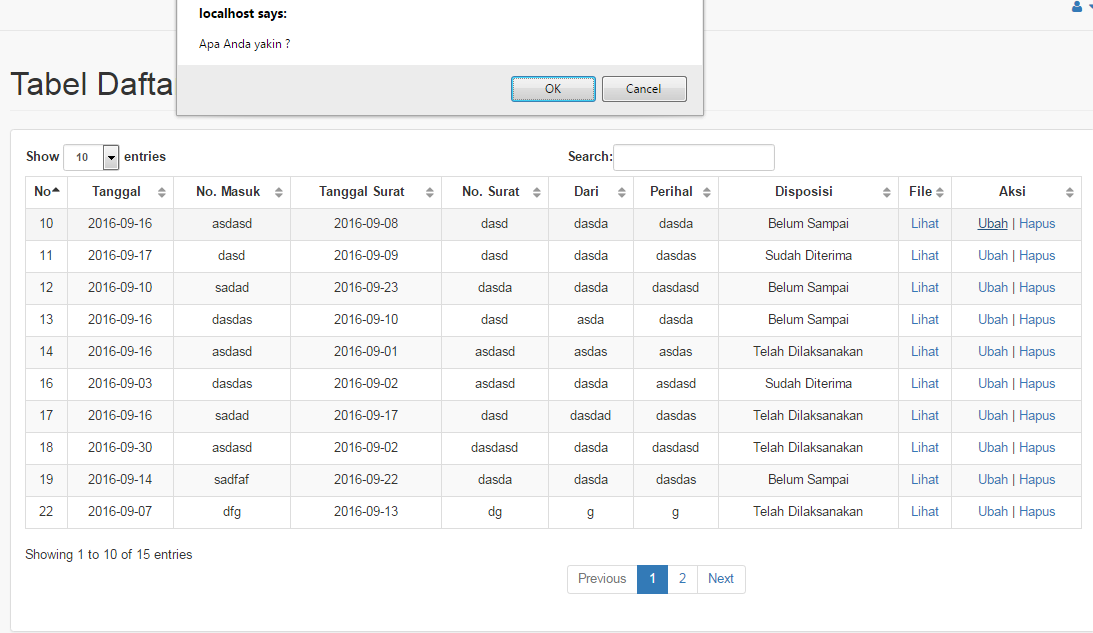
1. Halaman Form Update Surat Keluar

Halaman ini adalah halaman yang menampilkan form isian untuk mengubah data surat keluar. Pada halaman ini, data yang sebelumnya disimpan ditampilkan sesuai dengan *field*-nya.

Gambar 5. 11 *Update* Surat Keluar

1. Halaman Form Tabel Daftar Surat Masuk dengan *Alert* untuk “Ubah” dan “Hapus”

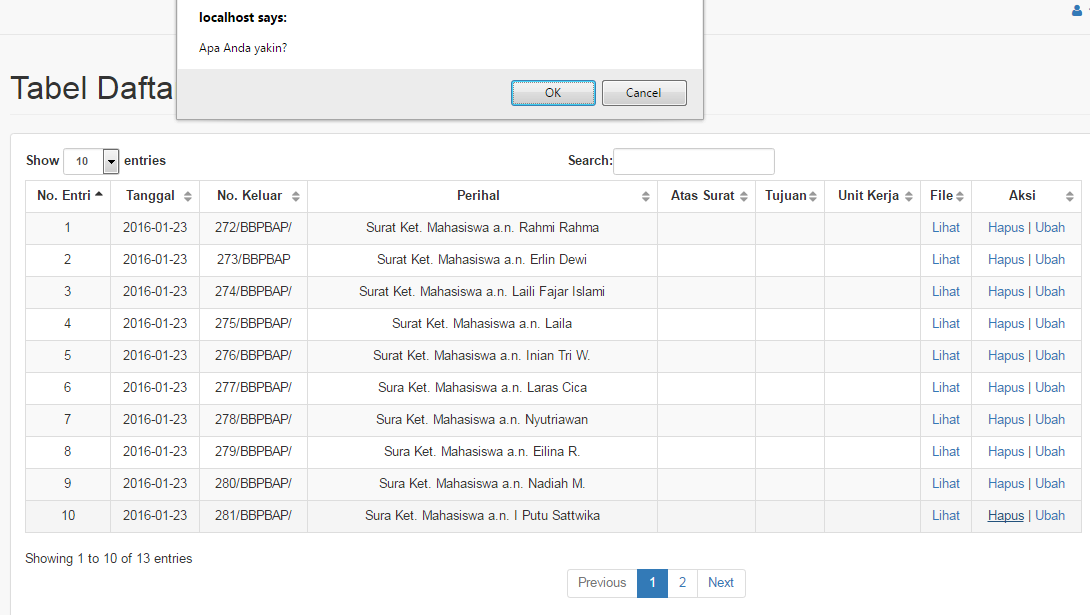
Halaman ini adalah halaman yang menampilkan seluruh data surat masuk dengan menampilkan *alert* untuk ubah data pada surat yang dipilih.



Gambar 5.12 *Alert* Muncul Ketika *Link* “Ubah” Atau “Hapus” Diklik

1. Halaman Form Tabel Daftar Surat Keluar dengan *Alert* untuk “Ubah” dan “Hapus”

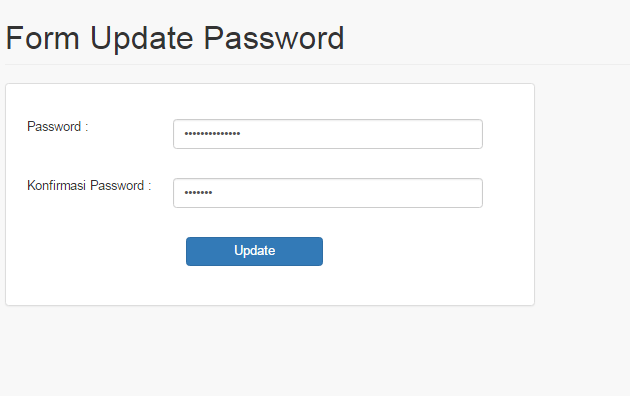
Halaman ini adalah halaman yang menampilkan seluruh data surat keluar dengan menampilkan *alert* untuk ubah data pada surat yang dipilih.



Gambar 5.13 *Alert* Muncul Ketika *Link* “Ubah” atau “Hapus” Diklik

1. Halaman Form *Update Password* Administrator

Halaman ini adalah halaman untuk mengubah *password* administrator.



Gambar 5.14 Halaman Form *Update* *password* administrator

## 5.4 Pengujian

Pengujian Aplikasi Dokumentasi Persuratan Berbasis *Web* Pada BBPBAP Jepara Jawa Tengah menggunakan metode *black box*, yaitu melakukan pengujian fungsionalitas dari perangkat lunak untuk menemukan kesalahan pada persyaratan fungsional dengan mengabaikan mekanisme internal atau komponen dari program tersebut. Pengujian *Black box* diterima jika fitur-fitur dari perangkat lunak telah memenuhi kebutuhan aplikasi.

1. **Persiapan Prosedural**

Langkah-langkah prosedur untuk melakukan pengujian pada Aplikasi Dokumentasi Persuratan Berbasis Web Pada BBPBAP Jepara Jawa Tengah adalah sebagai berikut.

1. Menyiapkan *software* yang ingin diuji serta *hardware* yang mendukung dalam proses pengujian.
2. Menetapkan tujuan yang hendak dicapai dalam setiap pengujian yang dilakukan yang telah terdapat pada skenario rencana pengujian.
3. Menentukan kategori keberhasilan dalam pengujian.
4. Membuat kasus uji dan hasil uji.
5. Menemukan cacat yang ada dalam perangkat lunak.
6. Membuat perbaikan terhadap yang ada.
7. Menentukan diterima atau tidaknya alur tes.
8. Membuat evaluasi pengujian.
9. **Rencana Pengujian**

Daftar untuk rencana pengujian Aplikasi Dokumentasi Persuratan Berbasis *Web* Pada BBPBAP Jepara Jawa Tengah dapat dilihat pada tabel 5.1.

Tabel 5.1 Tabel Rencana Pengujian

| No | Fungsional | Pengujian | Jenis Pengujian | Identifikasi |
| --- | --- | --- | --- | --- |
| 1. | SRS-ADP-F-01 | Antarmuka *Login* | *Black box* | U-1-001 hingga U-1-003 |
| 2. | SRS-ADP-F-02 | Antarmuka Tambah Data Surat Masuk dan Surat Keluar | *Black box* | U-2-001 hingga U-2-003 |
| 3. | SRS-ADP-F-03 | Antarmuka Menampilkan File Surat Masuk dan Surat Keluar | *Black box* | U-3-001 |
| 4. | SRS-ADP-F-04 | Antarmuka Ubah Data Surat Masuk dan Surat Keluar | *Black box* | U-4-001 |
| 5. | SRS-ADP-F-05 | Antarmuka Hapus Data Surat Masuk dan Surat Keluar | *Black box* | U-5-001 |
| 6. | SRS-ADP-F-06 | Antarmuka Ubah *Password* Administrator | *Black box* | U-6-001 |

1. **Deskripsi dan Hasil Uji**

Deskripsi dan Hasil Uji dapat dilihat pada Lampiran 1.

1. **Analisis Hasil Pengujian**

Aplikasi Dokumentasi Persuratan Berbasis *Web* Pada BBPBAP Jepara Jawa Tengah telah dilakukan sejumlah pengujian untuk menguji kelayakan aplikasi. Hasil pengujian dari Aplikasi Dokumentasi Persuratan Berbasis *Web* Pada BBPBAP Jepara Jawa Tengah telah diterima, sehingga dapat disimpulkan bahwa aplikasi ini telah memenuhi semua kebutuhan fungsional yang telah disepakati sebelumnya.

# BAB VI PENUTUP

**6.1 Kesimpulan**

Kesimpulan dari laporan praktek kerja lapangan Aplikasi Dokumentasi Persuratan Berbasis Web Pada BBPBAP Jepara Jawa Tengah ini adalah sebagai berikut:

1. Aplikasi Dokumentasi Persuratan Pada BBPBAP Jepara Jawa Tengah adalah suatu sistem yang digunakan untuk mendokumentasikan persuratan pada BBPBAP Jepara Jawa Tengah
2. Aplikasi ini membantu administrator persuratan yang sudah berusia lanjut
3. Menu yang ada dalam aplikasi antara lain memasukkan, melihat, menghapus dan mengunggah data – data surat serta mengubah *password* administrator.
4. Berdasarkan proses pengujian dapat diketahui bahwa sistem telah mempunyai fungsi yang sesuai dengan SRS.

**6.2 Saran**

Saran-saran untuk pengembangan Aplikasi Dokumentasi Persuratan Berbasis *Web* Pada BBPBAP Jepara Jawa Tengah adalah sebagai berikut:

1. Aplikasi Dokumentasi Persuratan Berbasis *Web* Pada BBPBAP Jepara Jawa Tengah dapat dikembangkan lagi untuk mengirim surat ke pimpinan via *email*.
2. Data *user* dalam aplikasi ini dapat dilengkapi lagi dengan menambahkan foto administrator.

# DAFTAR PUSTAKA

*Admin.* 2013. *Sejarah BBPBAP.* Dalam http://bbpbapjepara.djpb.kkp.go.id/index.php/article/details/sejarah-bbpbap-jepara diakses tanggal 28 Desember 2015.

*Admin*. 2013. *Struktur Organisasi*. Dalam <http://bbpbapjepara.djpb.kkp.go.id/index.php/article/details/struktur-organisasi> diakses tanggal 29 Desember 2015

*Admin*. 2013. *Visi dan Misi BBPBAP.* Dalam http://bbpbapjepara.djpb.kkp.go.id/index.php/article/details/tugas-pokok-dan-fungsi-visi-misi-bbpbap-jepara diakses tanggal 28 Desember 2015

Evadeva. 2007. “Desain Perangkat Lunak”, (Online), (http://evadeva. blogspot.com/2007/06/desain-perangkat-lunak.html, diakses pada 8 September 2016).

Anhar, 2010. *Panduan Menguasai PHP & MySQL secara Otodidak.* Jakarta: MediaKita.

Aziz, F., 2005. *Object Oriented Programming dengan PHP5.* Jakarta: PT. Elex Media Komputindo.

Daqiqil, I., 2011. Framework Codeigniter. Dalam: *Sebuah Panduan dan Best Practice.* Pekanbaru: s.n.

Fowler, M., 2003. *UML Destilled: A Brief Guide to the Standard Object Modeling Language.* 3rd penyunt. s.l.:Addison Wesley.

Triandini, Evi.2012.*Step by Step Desain Proyek Menggunakan UML.* Yogyakarta:Andi

Fathansyah. 2007. *Buku Teks Komputer Basis Data*. Bandung: Informatika.

Nugroho, A., 2005. *Rational Rose untuk Pemodelan Berorientasi Objek.* Bandung: Informatika.

Pressman, R. S., 2001. *Software Engineering: A Practitioner's Approach.* Fifth Edition penyunt. New York: McGraw-Hill Companies, Inc..

Rosa, A. S. & Shalahuddin, M., 2013. *Rekayasa Perangkat Lunak Terstruktur dan Berorientasi Objek.* edisi pertama penyunt. bandung: Informatika Bandung.

Sommerville, I., 2003. *Software Engineering. Ed ke-6.* Jakarta: Erlangga.

# LAMPIRAN

Lampiran 1.

**Deskripsi dan Hasil Uji**

**Deskripsi dan Hasil Uji**

1. Pengujian *Login*

Kode SRS : SRS-ADP-F-01

Deskripsi : Pengujian terhadap antarmuka melakukan *login*

Tabel L1.1 Tabel Pengujian *Login*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Identifikasi | Deskripsi | Prosedur Pengujian | Keluaran yang Diharapkan | Evaluasi Hasil | Kesimpulan |
| U-1-001 | Menguji dengan mengisi form secara benar | Administrator memasukkan *username* dan *password* dengan benar dan menekan tombol *login* | Aplikasi menampilkan halaman menu utama | Dapat masuk kedalam sistem dan menampilkan halaman menu utama | Diterima |
| U-1-002 | Menguji dengan mengisi form secara salah | Administrator memasukkan *username* dan *password* dengan salah dan menekan tombol *login* | Aplikasi menampilkan pesan salah dan menampilkan kembali antarmuka *login* | Menampilkan pesan kesalahan dan kembali ke halaman *login* | Diterima |
| U-1-003 | Menguji dengan tidak mengisi form dan langsung menekan tombol *login* | Administrator mengosongkan field dan menekan tombol *login* | Aplikasi menampilkan pesan salah dan menampilkan kembali antarmuka *login* | Menampilkan pesan kesalahan dan kembali ke halaman *login* | Diterima |

1. Pengujian Tambah Data Surat Masuk dan Surat Keluar

Kode SRS : SRS-ADP-F-02

Deskripsi : Pengujian terhadap antarmuka tambah data Surat Masuk dan Surat Keluar

Tabel L1.2 Tabel Pengujian Tambah Data Surat Masuk dan Surat Keluar

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Identifikasi | Deskripsi | Prosedur Pengujian | Keluaran yang Diharapkan | Evaluasi Hasil | Kesimpulan |
| U-2-001 | Menguji dengan mengisi form secara benar dan sesuai dengan formatnya | Administrator memasukkan data surat masuk atau surat keluar sesuai dengan contoh kemudian menekan tombol tambah | Aplikasi menampilkan pesan data berhasil dimasukkan | Dapat menampilkan pesan data berhasil menambahkan data surat masuk dan surat keluar dan disimpan di dalam database | Diterima |
| U-2-002 | Menguji dengan mengisi form secara salah dan sesuai dengan formatnya | Administrator memasukkan data surat masuk atau surat keluar tidak sesuai dengan contoh kemudian menekan tombol tambah | Aplikasi menampilkan pesan kesalahan dan meminta Administrator melakukan pengecekan terhadap data yang dimasukkan | Dapat menampilkan pesan error dan data surat masuk atau surat keluar tidak ada di database | Diterima |
| U-2-003 | Menguji dengan mengosongkan form | Administrator mengosongkan *field* dan menekan tombol tambah | Aplikasi menampilkan meminta *field*-nya diisi | Dapat menampilkan pesan error dan data surat masuk atau surat keluar tidak ada di database | Diterima |

1. Pengujian Menampilkan *File* Surat Masuk dan Surat Keluar

Kode SRS : SRS-ADP-F-03

Deskripsi : Pengujian terhadap antarmuka Menampilkan *File* Surat Masuk dan Surat Keluar

Tabel L1.3 Tabel Menampilkan *File* Surat Masuk dan Surat Keluar

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Identifikasi | Deskripsi | Prosedur Pengujian | Keluaran yang Diharapkan | Evaluasi Hasil | Kesimpulan |
| U-3-001 | Menguji dengan melakukan *click link* “Lihat” | Administrator memilih salah satudata surat dan melakukan *click link* “Lihat”, lalu amati apakah file yang ditampilkan sudah benar atau tidak. | *File* yang ditampilkan sudah benar | Dapat menampilkan *file* yang benar | Diterima |

1. Pengujian Mengubah Data Surat Masuk dan Surat Keluar

Kode SRS : SRS-ADP-F-04

Deskripsi : Pengujian terhadap antarmuka Mengubah Data Surat Masuk dan Surat Keluar

Tabel L1.4 Tabel Pengujian Mengubah Data Surat Masuk dan Surat Keluar

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Identifikasi | Deskripsi | Prosedur Pengujian | Keluaran yang Diharapkan | Evaluasi Hasil | Kesimpulan |
| U-4-001 | Menguji dengan melakukan *click link* “Ubah” | Administrator memilih salah satudata surat dan melakukan *click link* “Ubah”, lalu mengisi/ atau mengubah data pada satu/ sebagian/ semua field yang ada pada form. | Aplikasi mengubah data surat masuk atau surat keluar | Dapat mengubah data surat masuk atau surat keluar dan disimpan di dalam *database* | Diterima |

1. Pengujian Hapus Data Surat Masuk dan Surat Keluar

Kode SRS : SRS-ADP-F-05

Deskripsi : Pengujian terhadap antarmuka hapus data surat masuk dan surat keluar

Tabel L1.5 Tabel Hapus Data Surat Masuk dan Surat Keluar

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Identifikasi | Deskripsi | Prosedur Pengujian | Keluaran yang Diharapkan | Evaluasi Hasil | Kesimpulan |
| U-5-001 | Menguji dengan melakukan *click link* “Hapus” | Administrator memilih salah satudata surat dan melakukan *click link* “Hapus”. | Aplikasi menghapus data yang dipilih | Dapat menghapus satu data yang dipilih | Diterima |

1. Pengujian Mengubah *Password* Admin

Kode SRS : SRS-ADP-F-07

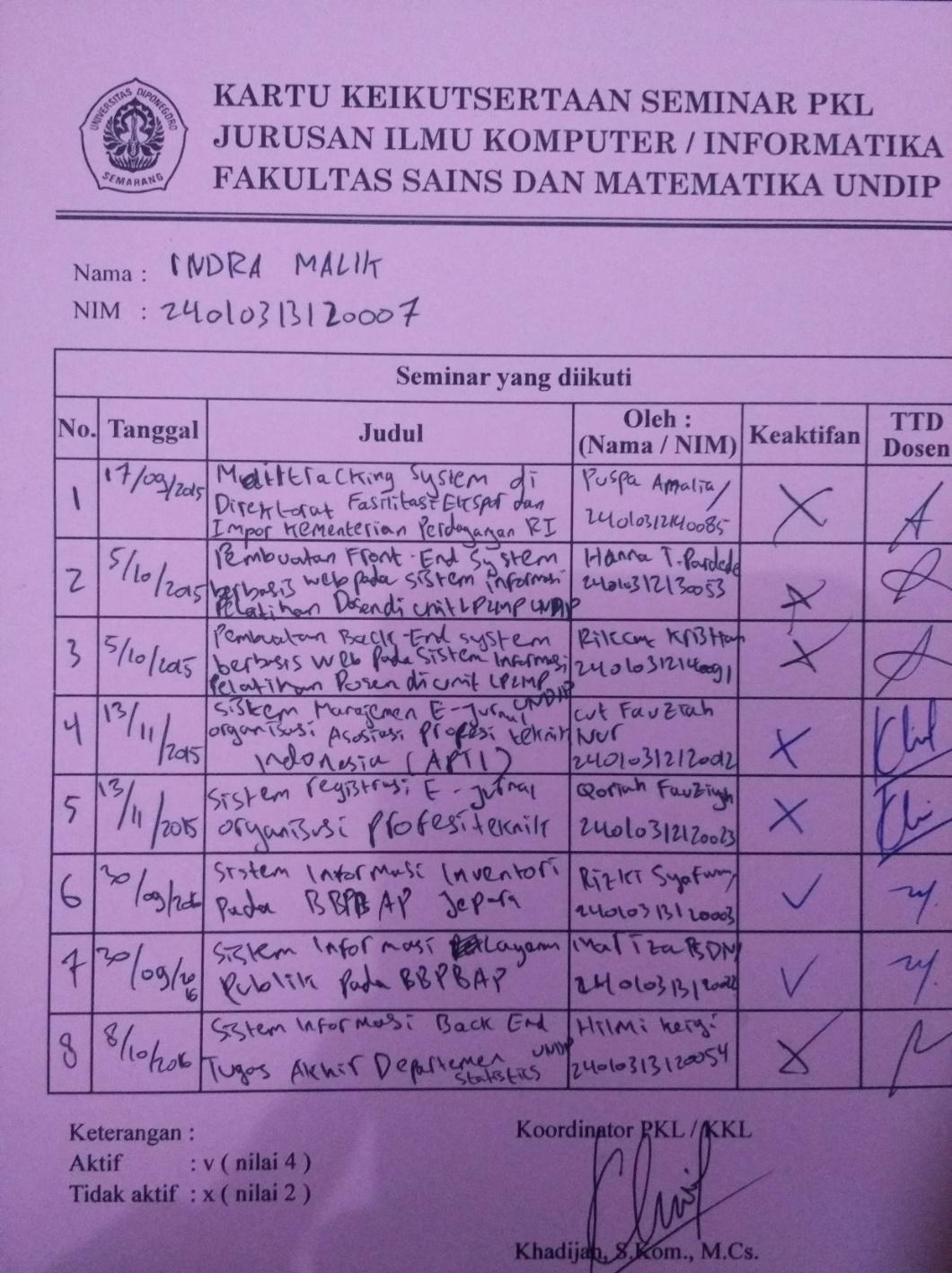
Deskripsi : Pengujian terhadap antarmuka Mengubah *Password* Admin

Tabel L1.6 Tabel Pengujian Mengubah *Password* Admin

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Identifikasi | Deskripsi | Prosedur Pengujian | Keluaran yang Diharapkan | Evaluasi Hasil | Kesimpulan |
| U-7-001 | Menguji dengan mengisi *form update password* secara benar | Administrator memasukkan *password* baru dan menekan tombol perbarui | *Password* admin telah diubah | Dapat mengubah *password* admin dan menampilkan pesan berhasil | Diterima |

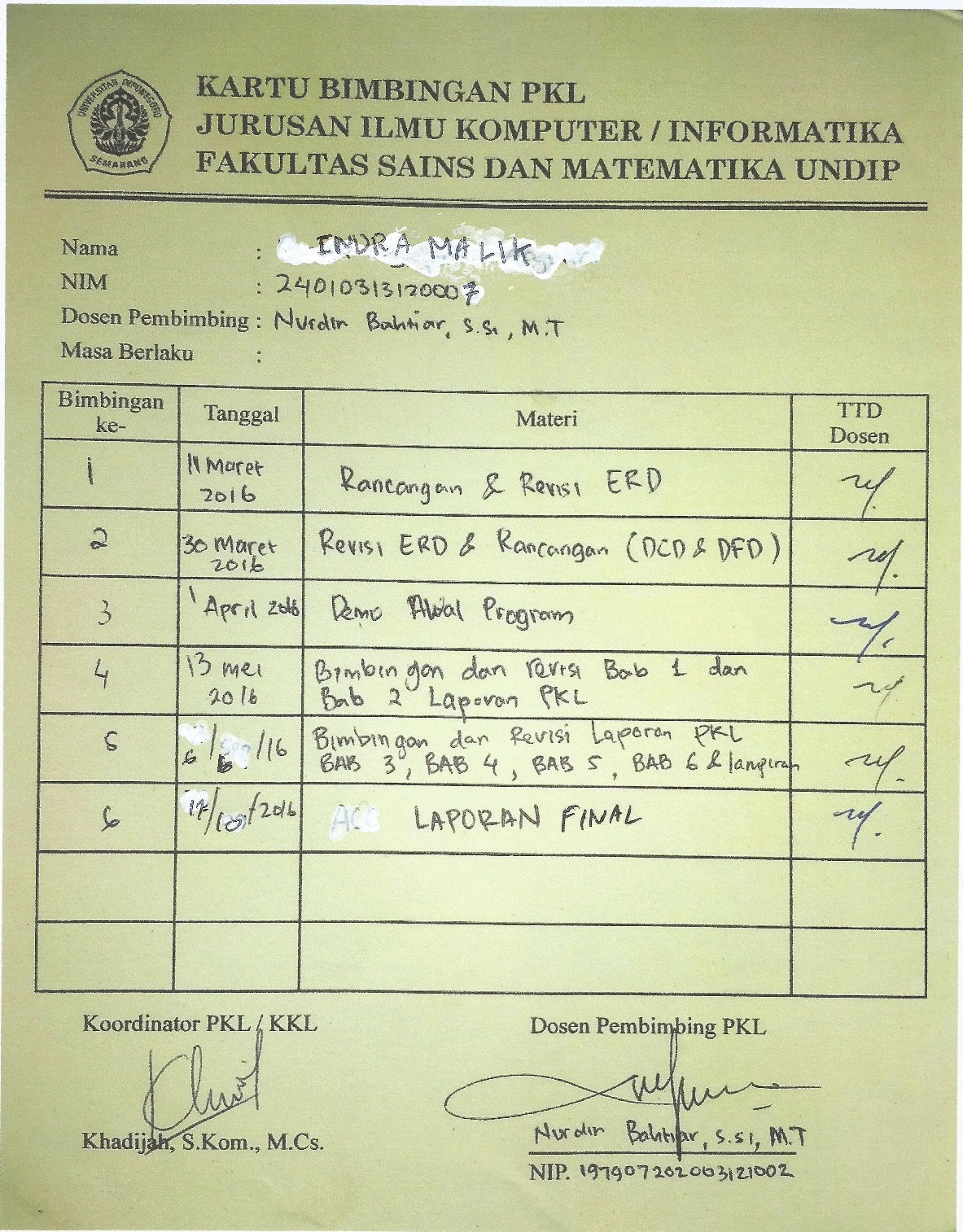
Lampiran 2.

**Kartu Keikutsertaan Seminar PKL**

****

Lampiran 3.

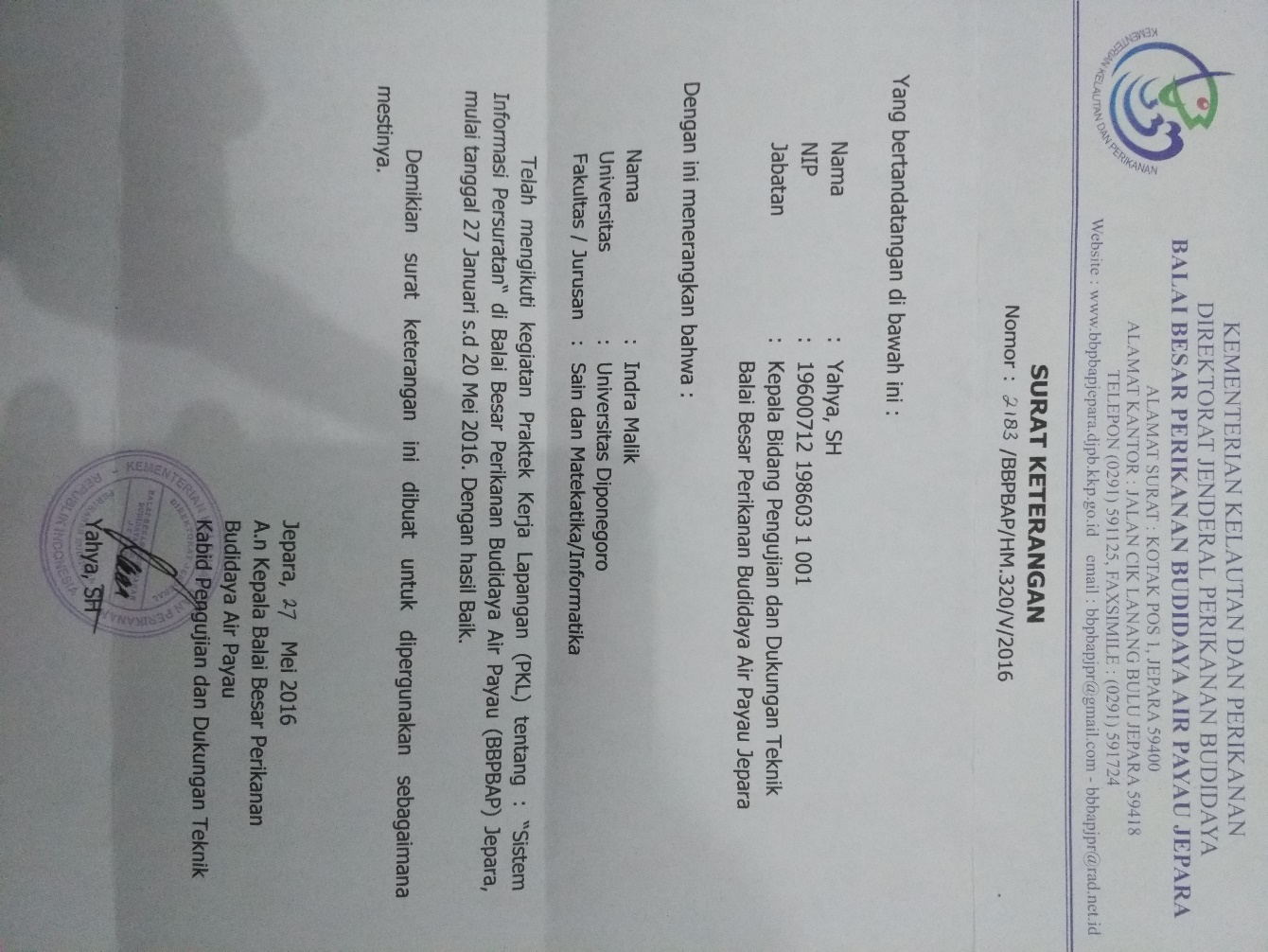
**Kartu Bimbingan PKL**



Indra Malik

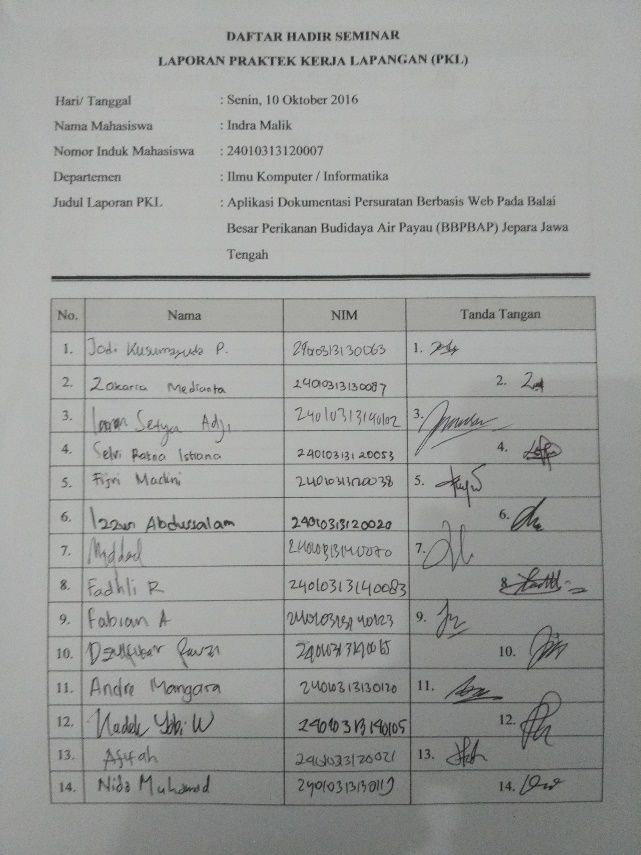
Lampiran 4.

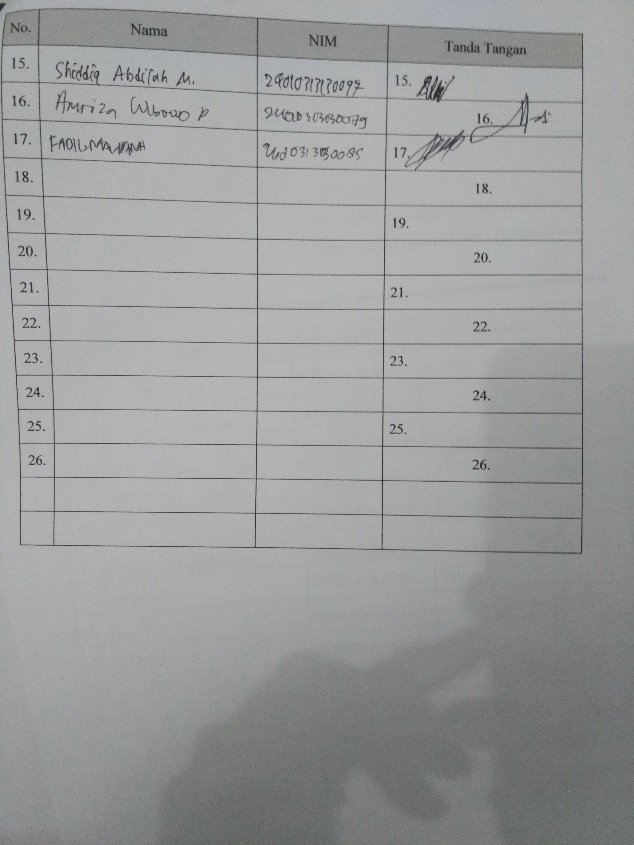
**Surat Keterangan Menyelesaikan PKL**

****

Lampiran 5.

**Daftar Hadir Peserta Seminar PKL**

****

****

Lampiran 6.

**Tanya Jawab Seminar PKL**

Nama : Jodi Kusumayuda

NIM : 30063

Untuk : Indra

Pertanyaan : ada disposisi pada surat masuk, apa maksudnya?

Jawaban : disposisi adalah instruksi dari atasan kepada bawahan untuk melakukan sesuatu

================================================================================================================================

Nama : Amriza Wibowo

NIM : 30079

Untuk : Semua

Pertanyaan : berbasis web atau ada mobile juga? Jika mobile, apakah sudah responsif untuk mobile?

Jawaban : Indra : berbasis web saja, resolusi disesuaikan dengan PC

================================================================================================================================

Nama : Shidiq

NIM : 30097

Untuk : Indra

Pertanyaan : Antisipasi untuk memudahkan pengguna berusia lanjut dalam menggunakan sistem?

Jawaban : Admin diajarkan oleh pengembang cara penggunaan aplikasi

================================================================================================================================

Nama : Dzulfikar Fauzi

NIM : 40065

Untuk : Indra

Pertanyaan : pengembangannya menggunakan framework atau tidak, dan alasannya

Jawaban : Indra : mencoba pendekatan baru, menggunakan framework codigniter karena mudah dipelajari dan banyak library nya

Nama : Fadhil

NIM : 30065

Untuk : Semua

Pertanyaan : Kendala selama PKL apa saja?

Jawaban : pengembang dan client sulit dalam berkomunikasi karena jarak yang jauh, dan perbedaan umur