SISTEM PELAPORAN DAN PENGOLAHAN DATA PAJAK BUMI DAN BANGUNAN (PBB) KABUPATEN JOMBANG BERBASIS *WEB* MENGGUNAKAN *FRAMEWORK CODEIGNITER*

SKRIPSI

LABORATORIUM REKAYASA PERANGKAT LUNAK

Diajukan untuk Memenuhi Persyaratan

Memperoleh Gelar Sarjana Komputer

Disusun Oleh :

VINISHYA AMIN

105060807111123

PROGRAM STUDI INFORMATIKA / ILMU KOMPUTER

PROGRAM TEKNOLOGI INFORMASI DAN ILMU KOMPUTER

UNIVERSITAS BRAWIJAYA

2014

LEMBAR PERSETUJUAN

SISTEM PELAPORAN DAN PENGOLAHAN DATA PAJAK BUMI DAN BANGUNAN (PBB) KABUPATEN JOMBANG BERBASIS *WEB* MENGGUNAKAN *FRAMEWORK CODEIGNITER*

SKRIPSI

Untuk Memenuhi Sebagian Persyaratan Mencapai Gelar Sarjana Komputer

Disusun Oleh :

VINISHYA AMIN

105060807111123

Telah diperiksa dan disetujui oleh:

Pembimbing I	Pembimbing II,
Ismiarta Aknuranda, S.T., M.Sc., Ph.D.	**Fajar Pradana, S.ST., M.Eng.**
NIK. 740719 06 1 1 0079	**NIK. 871121 16 1 1 0371**

LEMBAR PENGESAHAN

SISTEM PELAPORAN DAN PENGOLAHAN DATA PAJAK BUMI DAN BANGUNAN (PBB) KABUPATEN JOMBANG BERBASIS *WEB* MENGGUNAKAN *FRAMEWORK CODEIGNITER*

SKRIPSI

Untuk Memenuhi Sebagian Persyaratan Mencapai Gelar Sarjana Komputer

Disusun Oleh :

VINISHYA AMIN
105060807111123

Skripsi ini telah diuji dan dinyatakan lulus pada
tanggal 25 Juni 2014

Penguji I	Penguji II
Issa Arwani, S.Kom., M.Sc.	**Diah Priharsari, ST., MT.**
NIP. 19830922 201212 1 003	**-**

Penguji III

Aswin Suharsono, ST., MT.

NIK. 840919 06 1 1 0251

Mengetahui
Ketua Program Studi Informatika / Ilmu Komputer

Drs. Marji, M.T.
NIP. 19670801 199203 1 001

PERNYATAAN
ORISINALITAS SKRIPSI

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

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

Malang, 11 Juni 2014

Mahasiswa,

Vinishya Amin
NIM. 105060807111123

KATA PENGANTAR

Puji syukur penulis panjatkan kehadirat Allah SWT, karena berkat rahmat dan hidayah-Nya, penulis dapat menyelesaikan penyusunan skripsi ini. Tak lupa penulis haturkan shalawat serta salam kepada Rasulullah Nabi Muhammad SAW, keluarga, sahabat-sahabat, dan para pengikutnya, semoga rahmat Allah SWT selalu tercurah untuk kita semua. Amin.

Skripsi ini disusun dengan judul "**SISTEM PELAPORAN DAN PENGOLAHAN DATA PAJAK BUMI DAN BANGUNAN (PBB) KABUPATEN JOMBANG BERBASIS *WEB* MENGGUNAKAN *FRAMEWORK CODEIGNITER***". Penulisan skripsi ini disusun untuk memenuhi sebagian syarat menjadi Sarjana Komputer. Dalam penyusunan, skripsi ini telah banyak mendapat bantuan dari berbagai pihak. Ucapan terima kasih penulis sampaikan kepada:

1. Bapak Aminudin, S.Pd., Ibu Nurul Ayyun, Aulia Putri Nurdiani, S.Farm,Apt., Heri Widodo, Amd., M. Enggar Syahrudin, dan M. Khalif Faturrahman selaku Ayah, Bunda, Kakak dan Keponakan tercinta atas segenap do'a, dukungan dan kasih sayang yang telah diberikan.
2. Bapak Ismiarta Aknuranda, ST., M.Sc., Ph.D. dan Bapak Fajar Pradana, S.ST., M.Eng. selaku dosen pembimbing yang telah membimbing penulis dalam pengerjaan dan penulisan skripsi ini.
3. Bapak Drs. Marji, MT. selaku Ketua Program Studi Informatika/Ilmu Komputer Program Teknologi Informasi dan Ilmu Komputer.
4. Bapak Suprapto, S.T., M.T. selaku dosen penasihat akademik penulis yang telah memberikan pengarahan selama penulis menempuh pendidikan di Program Teknologi Informasi dan Ilmu Komputer Universitas Brawijaya.
5. Bapak Denny Sagita Rusdianto, S.Kom., M.Kom. selaku Kepala Laboratorium Rekayasa Perangkat Lunak Program Teknologi Informasi dan Ilmu Komputer.
6. Segenap dosen Program Teknologi Informasi dan Ilmu Komputer Universitas Brawijaya atas segenap ilmu pengetahuan dan perhatian yang diberikan.

7. Segenap staff dan pegawai Program Teknologi Informasi dan Ilmu Komputer Universitas Brawijaya atas segala bantuan yang bersifat administratif.
8. Adien Faishol Habib, S.Kom. atas kasih sayang, perhatian, dan dukungan yang luar biasa kepada penulis dan tetap berada di samping penulis sampai saat ini serta bantuannya dalam menyelesaikan skripsi.
9. Intan Rahmawati dan Eko Aprilia atas bantuan dan persahabatan yang tak ternilai mulai awal perkuliahan sampai sekarang.
10. Andrian Riza Hermawan, Arif Aulia, Mafturiza Yumida, Yulis Raga, Deny Irfan, Galih Julianto Purnomo, Candra Irwansyah, S.Kom., dan Mochtar Setya Putra, S.Kom. atas bantuannya dalam kelengkapan pembuatan sistem.
11. Keluarga Besar Kos Kertosentono 109 atas dukungan selama penulis mengerjakan skripsi serta hiburan yang diberikan.
12. Teman-teman Program Studi Informatika/Ilmu Komputer angkatan 2010 tercinta yang selalu memberikan semangat, dukungan, dan bantuan pikiran.
16. Himpunan Mahasiswa Teknik Informatika khususnya Lembaga Himpunan Mahasiswa Informatika periode 2012-2013 atas pengalaman organisasi dan politik yang luar biasa.
17. Serta semua pihak yang telah membantu dan memberikan pengalaman berharga bagi penulis selama penulis menjalani masa perkuliahan.

Akhirnya atas segala bantuan semua pihak semoga mendapat balasan yang setimpal dari Allah SWT. Harapan penulis semoga skripsi ini dapat memberikan manfaat pada semua pihak, terutama kepada penulis dan para pengembang yang nantinya akan mengembangkan penelitian dari penulis.

Malang, 11 Juni 2014

Penulis

ABSTRAK

Vinishya Amin.2014. : Sistem Pelaporan Dan Pengolahan Data Pajak Bumi Dan Bangunan (PBB) Kabupaten Jombang Berbasis *Web* Menggunakan *Framework CodeIgniter*. Skripsi Program Studi Informatika/Ilmu Komputer, Program Teknologi Informasi dan Ilmu Komputer, Universitas Brawijaya. Dosen Pembimbing : Ismiarta Aknuranda, S.T., M.Sc., Ph.D. dan Fajar Pradana, S.ST., M.Eng.

Pajak Bumi dan Bangunan (PBB) digunakan sebagai hasil Pendapatan Asli Daerah Sendiri. PBB dapat dibayarkan melalui instansi yang ditunjuk atau Bank. Hal ini menyebabkan data pembayaran menjadi terpencar-pencar sesuai tempat pembayaran PBB yang dilakukan oleh wajib pajak. Kemudian, setiap instansi yang ditunjuk dan Bank perlu melaporkan data pembayaran ini secara manual ke Dinas Pendapatan Daerah untuk penggabungan dan pengolahan data lebih lanjut. Hal ini terjadi di Kabupaten Jombang. Oleh karena itu dirancang sebuah sistem pelaporan dan pengolahan data pada skripsi ini untuk membantu dalam memusatkan pengolahan data pembayaran PBB. Metode pengembangan sistem pelaporan dan pengolahan data ini menggunakan model *reuse-oriented software engineering.* Sistem ini diuji menggunakan pengujian validasi dan kompatibilitas *browser*. Dari hasil pengujian validasi, seluruh spesifikasi persyaratan sistem telah dipenuhi atau dengan kata lain sistem dapat menampilkan seluruh fungsi sesuai dengan pemodelan persyaratan. Sedangkan hasil dari pengujian kompatibilitas *browser* yang dilakukan pada Firefox *browser*, Chrome *browser* dan *Internet Explorer browser*, seluruh *browser* dapat menampilkan seluruh antar muka sesuai dengan perancangan antar muka pengguna. Pada Firefox *browser* dan Chrome *browser* juga dapat menampilkan antar muka secara lengkap. Tetapi pada *Internet Explorer browser* tidak dapat menampilkan antar secara lengkap karena dalam kotak *login* tidak dapat menampilkan penjelasan (*placeholder*). Oleh karena itu, diperlukan pengembangan sistem agar sistem dapat diakses di *browser* mana saja.

Kata Kunci: PBB, sistem pelaporan dan pengolahan data, *reuse-oriented software engineering*

ABSTRACT

Vinishya Amin.2014.: Reporting System And Data Processing The Land And Building Tax In Jombang Based On Web Applying Codeigniter Framework. Skripsi Program Studi Informatika/Ilmu Komputer, Program Teknologi Informasi dan Ilmu Komputer, Universitas Brawijaya. Supervisors : Ismiarta Aknuranda, S.T., M.Sc., Ph.D. and Fajar Pradana, S.ST., M.Eng.

The Land and Buildings tax (PBB) used as the results of regional income. The PBB can be paid through a designated agency and Bank. Which is mean the payment data spreads over depending on where the Land and Building tax is paid so that. Then, designated agency and bank should report this payment data in manual of Regional Revenue Office to merger and process the data further. These, occurred in Jombang. Thus designed a processing the tax payer data and the tax obligate data system in this research for help centralize the obligate data payment PBB. The development method of reporting system uses a reuse-oriented software engineering model. The system was tested using validation and browser compatibility testing. The testing result of validation testing that all of requirements specification system completed or system can display all functional system suitable with requirements specification. While the testing result of browser compatibility at the browser Firefox, browser Chrome and browser Internet Explorer, all browser can display interface each page so suitable with design of user interface. At Firefox browser and Chrome browser can display completed interface too. But at the Internet Explorer browser display incompleted interface because in login box cannot display explanation (placeholder). Thus required development of a system so that the system can be accessed anywhere in the browser.

Keywords: *PBB, processing the tax payer data and the tax obligate data system, reuse-oriented software engineer*

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I
PENDAHULUAN

1.1 Latar Belakang

Pemerintah Daerah dalam melaksanakan pemerintahan dan pembangunan membutuhkan dana cukup besar. Dana tersebut selain diperoleh dari Pusat juga dari hasil Pendapatan Asli Daerah Sendiri yang diatur dalam peraturan perundang-undangan. Hal ini sejalan dengan upaya menegakkan kemandirian pembiayaan pembangunan salah satunya melalui penerimaan perpajakan. Sumber penerimaan yang sangat potensial bagi daerah sebagai pajak langsung salah satunya adalah Pajak Bumi dan Bangunan [GTS - 13].

Pajak Bumi dan Bangunan (PBB) merupakan Pajak Negara yang dikenakan terhadap bumi dan atau bangunan berdasarkan Undang-undang nomor 12 Tahun 1985 tentang Pajak Bumi dan Bangunan sebagaimana telah diubah dengan Undang-Undang nomor 12 Tahun 1994 [ANY - 09].

Dari pengamatan penulis terdapat beberapa masalah yang dihadapi pemerintah dalam pengelolaan PBB, salah satunya adalah kurang efisiennya penanganan sistem pelaporan data pembayaran PBB seperti yang terjadi di Kabupaten Jombang. Wajib pajak dapat membayar pajak di beberapa tempat, seperti Kelurahan/Kecamatan, Kantor Dinas Pendapatan Pengelolaan Keuangan Dan Aset Daerah (DPPKAD), dan Bank Jatim. Tetapi hal tersebut menyebabkan data pembayaran PBB menjadi terpencar sesuai dengan tempat pembayaran yang dilakukan. Hal ini menyebabkan dibutuhkannya pelaporan data pembayaran wajib pajak dari pihak Kelurahan/Kecamatan dan Bank ke pihak DPPKAD setiap minggunya secara manual. Sehingga pihak DPPKAD harus mencocokan data dari Kelurahan/Kecamatan dengan data di Bank secara manual juga. Jadi untuk memudahkan dalam penanganan sistem pelaporan pajak maka sistem ini perlu dipusatkan dalam pengolahan data wajib pajak yang diopersaikan secara *online*.

Pelaporan data pajak secara *online* dapat dibuat menggunakan *framework CodeIgniter* (CI). CI merupakan aplikasi *open source* yang berupa *framework* dengan model MVC (*Model, View, Controller*) untuk membangun aplikasi berbasis *web* dinamis dengan menggunakan PHP. CI memudahkan pengembang

aplikasi untuk membuat aplikasi *web* dengan cepat dan mudah dibandingkan dengan membuatnya dari awal.

Berdasarkan penjelasan informasi di atas, penulis mengambil judul skripsi *"Sistem Pelaporan Dan Pengolahan Data Pajak Bumi Dan Bangunan Kabupaten Jombang Berbasis Web Menggunakan Framework CodeIgniter"*. Pelaporan dan pengolahan data PBB dapat dilakukan melalui sistem Olah Data Pajak (ODP). ODP merupakan sebuah aplikasi yang diharapkan nantinya dapat membantu pihak Kelurahan/Kecamatan, Bank, dan DPPKAD dalam menangani pelaporan pembayaran PBB dan pengolahan data PBB dengan efisien. Dalam menangani pelaporan pembayaran PBB, selain dilakukan oleh sistem ODP, terdapat juga Sistem Virtual Bank yang merupakan simulasi sebuah sistem Bank yang sebenarnya. Sistem virtual Bank ini berfungsi untuk menangani validasi pembayaran PBB yang dilakukan oleh petugas Bank. Oleh karena itu, petugas Bank tidak perlu lagi melakukan pelaporan pembayaran PBB secara manual ke pihak DPPKAD untuk mencocokkan data pembayaran PBB yang ada di Bank dengan data pembayaran PBB dari Kelurahan, Kecamatan, dan DPPKAD.

1.2 Rumusan Masalah

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

1. Bagaimana memodelkan persyaratan sistem ODP untuk membantu DPPKAD, Kelurahan/Kecamatan, dan Bank dalam pelaporan dan pengolahan data PBB?
2. Bagaimana merancang dan mengimplementasikan sistem ODP menggunakan *framework CodeIgniter*?
3. Bagaimana menguji sistem ODP?

1.3 Batasan Masalah

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

1. *Framework* yang akan digunakan dalam sistem ODP adalah *CodeIgniter.*

2. Pengajuan penanganan PBB, seperti permohonan pengurangan pembayaran PBB, permohonan mutasi/pembetulan objek/subjek PBB, keberatan atas pengenaan PBB, dan permohonan salinan SPPT hanya sampai batas mengirim dan tidak dilakukan pemrosesan data lebih lanjut.
3. Data wajib pajak yang digunakan dalam studi kasus ODP berjumlah 20 orang yang diperoleh dari warga Kelurahan Kepanjen Jombang.
4. Sistem yang dibuat tidak memuat perhitungan pajak jadi data yang digunakan hanya data pajak yang sudah ada.
5. Sistem yang dibuat hanya prototipe dan belum bisa diimplementasikan sesungguhnya karena masih perlu pengembangan sistem lebih lanjut.

1.4 Tujuan

Adapun tujuan dari perancangan skripsi ini adalah sebagai berikut:

1. Memodelkan persyaratan sistem ODP untuk membantu DPPKAD, Kelurahan/Kecamatan, dan Bank dalam pelaporan dan pengolahan data PBB.
2. Merancang dan mengimplementasikan sistem ODP menggunakan *framework CodeIgniter*.
3. Melakukan pengujian yang dilakukan terhadap sistem ODP.

1.5 Manfaat

Penulisan skripsi ini diharapkan dapat memberikan manfaat yang baik dan berguna bagi penulis, pembaca, dan instansi. Adapun manfaat yang diharapkan adalah sebagai berikut:

- Bagi Penulis
    1. Mendapatkan pengetahuan dan wawasan terkait pemodelan persyaratan untuk pengembangan sistem.
    2. Sebagai sarana pengembangan pengetahuan dan wawasan terkait perancangan sistem berbasis *web* yang telah didapat dibangku perkuliahan.

- Bagi Pembaca

  Diharapkan dapat memberikan informasi dan menambah wawasan seputar PBB kepada pembaca apabila sistem nantinya dapat terealisasikan.

- Bagi Instansi

  Untuk merekomendasi salah satu solusi dalam membantu kinerja DPPKAD, Kelurahan/Kecamatan, dan Bank dalam pelaporan dan pengolahan data PBB di Kabupaten Jombang.

1.6 Sistematika Penulisan

Sistematika penulisan dalam skripsi ini adalah sebagai berikut:

BAB I Pendahuluan

Membahas latar belakang, rumusan masalah, tujuan, batasan masalah, manfaat, dan sistematika penulisan.

BAB II Kajian Pustaka

Menjelaskan dasar-dasar teori dan referensi yang mendasari dalam perancangan dan pembuatan sistem ODP dan Sistem Virtual Bank.

BAB III Metode Penelitian

Menjelaskan metode yang digunakan dalam penelitian yang terdiri dari studi literatur, pemodelan persyaratan, analisa komponen, perancangan dengan komponen, implementasi sistem, pengujian dan analisis, serta penulisan laporan.

BAB IV Identifikasi Persyaratan dan Perancangan

Membahas pemodelan persyaratan, analisa komponen dan perancangan dengan komponen sistem ODP dan Sistem Virtual Bank.

BAB V Implementasi

Menjelaskan implementasi dari sistem ODP dan Sistem Virtual Bank sesuai dengan perancangan sistem yang telah dibuat.

BAB VI Pengujian dan Analisis

Menjelaskan proses dan analisa hasil pengujian terhadap sistem yang telah dibuat.

BAB VII Penutup

Memuat kesimpulan yang diperoleh dari pembuatan dan pengujian sistem dalam skripsi ini serta saran-saran untuk pengembangan lebih lanjut.

BAB II
KAJIAN PUSTAKA

Pada bab ini berisi pembahasan tentang teori dasar yang digunakan sebagai literatur penunjang dalam pembuatan sistem. Teori dasar yang akan dibahas pada bab ini yaitu penjelasan tentang konsep dasar Pajak Bumi dan Bangunan (PBB), *software process model*, *Unified Modelling Language* (UML), pengujian perangkat lunak, *framework* CI, *Service Oriented Architecture* (SOA), *web service*, *Web Service Definition Language* (WSDL), *Simple Object Access Protocol* (SOAP), dan NuSOAP.

2.1 Pajak Bumi dan Bangunan (PBB)

PBB adalah Pajak Negara yang dikenakan terhadap bumi dan atau bangunan berdasarkan Undang-undang nomor 12 Tahun 1985 tentang Pajak Bumi dan Bangunan sebagaimana telah diubah dengan Undang-Undang nomor 12 Tahun 1994. PBB adalah pajak yang bersifat kebendaan dalam arti besarnya pajak terutang ditentukan oleh keadaan objek yaitu bumi/tanah dan atau bangunan. Keadaan subjek (siapa yang membayar) tidak ikut menentukan besarnya pajak [ANY - 09].

Objek PBB meliputi bumi dan bangunan. Bumi merupakan permukaan bumi (tanah dan perairan) dan tubuh bumi yang ada di pedalaman serta laut wilayah. Contoh objek pajak bumi adalah sawah, ladang, kebun, tanah, pekarangan, tambang, dan lain-lain. Sedangkan bangunan adalah konstruksi teknik yang ditanam atau diletakkan secara tetap pada tanah dan atau perairan. Contoh objek bangunan adalah rumah tempat tinggal, bangunan tempat usaha, gedung bertingkat, pusat perbelanjaan, jalan tol, kolam renang, dan lain-lain. Ada beberapa objek pajak yang tidak dikenakan PBB, seperti [ANY - 09]:

1. Digunakan semata-mata untuk melayani kepentingan umum dibidang ibadah, sosial, kesehatan, pendidikan dan kebudayaan nasional yang tidak dimaksudkan untuk memperoleh keuntungan, seperti masjid, gereja, rumah sakit pemerintah, sekolah, panti asuhan, candi, dan lain-lain.
2. Digunakan untuk kuburan, peninggalan purbakala atau yang sejenis dengan itu.

3. Merupakan hutan lindung, suaka alam, hutan wisata, taman nasional, tanah penggembalaan yang dikuasai oleh desa, tanah Negara yang belum dibebani suatu hak.
4. Digunakan oleh perwakilan diplomatik berdasarkan asas perlakuan timbal balik.
5. Digunakan oleh badan dan perwakilan organisasi internasional yang ditentukan oleh Menteri Keuangan.

Subjek pajak merupakan orang yang secara pribadi atau badan yang mempunyai suatu hak atas bumi dan memperoleh manfaat atas bumi dan memiliki bangunan dan menguasai atau memperoleh manfaat atas bangunan. Wajib pajak adalah subjek pajak yang dikenakan kewajiban membayar pajak. Subjek pajak harus mendaftarkan Objek Pajaknya ke Dinas Pendapatan Daerah (DISPENDA) dengan menggunakan formulir Surat Pemberitahuan Objek Pajak (SPOP) [ANY - 09].

Dasar pengenaan PBB adalah Nilai Jual Objek Pajak (NJOP). NJOP ditetapkan per wilayah berdasarkan keputusan Menteri Keuangan dengan mendengar pertimbangan Bupati/Walikota serta memperhatikan [ANY - 09]:

a. Harga rata-rata yang diperoleh dari transaksi jual beli yang terjadi.
b. Perbandingan harga dengan objek lain yang sejenis yang letaknya berdekatan dan fungsinya sama dan telah diketahui harga jualnya.
c. Nilai perolehan baru.
d. Penentuan Nilai Jual Objek Pajak pengganti.

2.2 *Software Process Model*

Model proses perangkat lunak merupakan penyederhanaan dari suatu proses perangkat lunak. Setiap model proses merupakan proses dari perspektif tertentu dan hanya menyediakan informasi parsial tentang proses tersebut. Sebuah model proses untuk rekayasa perangkat lunak dipilih berdasarkan sifat proyek dan aplikasi, metode dan alat-alat yang akan digunakan, dan kontrol *deliverable* yang diperlukan (SMI - 11). Beberapa model proses perangkat lunak yang sering digunakan para pengembang perangkat lunak adalah *waterfall model, incremental development,* dan *reuse-oriented software engineering.* Dalam sistem ODP

menggunakan model proses perangkat lunak *reuse-oriented* karena dalam pembuatan sistem ODP menggunakan *framework* CI.

2.2.1 *Reuse-Oriented Software Engineering*

Reuse-oriented software engineering adalah penggunaan kembali perangkat lunak yang telah ada. Hal ini sering terjadi ketika orang-orang yang bekerja pada proyek mengetahui desain atau kode yang mirip dengan apa yang dibutuhkan. Mereka mencari dan memodifikasi sesuai kebutuhan lalu menggabungkan ke dalam sistem. Pendekatan *reuse-oriented software engineering* mengandalkan komponen perangkat lunak yang dapat digunakan kembali dan mengintegrasikan kerangka untuk komposisi komponen sistem [SMI - 11].

Tahapan utama dari *reuse-oriented software engineering* secara langsung mencerminkan dasar perancangan sistem pada skripsi ini adalah [SMI - 11]:

1. Analisis komponen

   Dari spesifikasi kebutuhan yang telah ditentukan, dicari komponen yang dapat mengimplementasi spesifikasi tersebut. Komponen yang sesuai kebutuhan dapat digunakan dalam pengembangan sistem sedangkan yang tidak sesuai dengan kebutuhan biasanya akan mengalami tahap modifikasi.
2. Tahap Modifikasi

   Selama tahap ini kebutuhan dianalisis menggunakan informasi tentang komponen yang telah ditentukan. Apabila tidak sesuai dengan kebutuhan maka dimodifikasi untuk mencerminkan komponen yang tersedia dengan tujuan untuk memaksimalkan sistem. Ketika satu kondisi tidak memungkinkan untuk memodifikasi kebutuhan maka kegiatan analisis komponen dapat dilakukan kembali untuk mencari solusi alternatif.
3. Desain sistem dengan *reuse*

   Tahap ini merupakan proses merancang *framework* pada sistem atau menggunakan *framework* yang tersedia. Para desainer mengambil komponen untuk digunakan dan mengatur *framework* untuk memenuhi kebutuhan.
4. Pengembangan dan integrasi perangkat lunak

   Perangkat lunak yang tidak dapat diperoleh secara eksternal maka akan dikembangkan dan komponen diintegrasikan untuk menciptakan sistem baru.

2.3 *Unified Modelling Language* (UML)

Unified Modelling Language (UML) adalah sebuah "bahasa" yang telah menjadi standar dalam industri untuk visualisasi, merancang dan mendokumentasikan sistem piranti lunak. UML menawarkan sebuah standar untuk merancang model sebuah sistem [DSR - 03].

Dengan menggunakan UML kita dapat membuat model untuk semua jenis aplikasi piranti lunak, dimana aplikasi tersebut dapat berjalan pada piranti keras, sistem operasi dan jaringan apapun, serta ditulis dalam bahasa pemrograman apapun. Seperti bahasa-bahasa lainnya, UML mendefinisikan notasi dan *syntax*/semantik. Notasi UML merupakan sekumpulan bentuk khusus untuk menggambarkan berbagai diagram piranti lunak. Setiap bentuk memiliki makna tertentu, dan UML *syntax* mendefinisikan bagaimana bentuk-bentuk tersebut dapat dikombinasikan. Notasi UML terutama diturunkan dari 3 notasi yang telah ada sebelumnya: Grady Booch OOD (*Object-Oriented Design*), Jim Rumbaugh OMT (*Object Modeling Technique*), dan Ivar Jacobson OOSE (*Object-Oriented Software Engineering*) [DSR - 03].

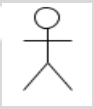
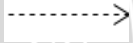
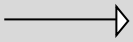
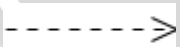
Adapun sejarah UML dimulai dari banyaknya metodologi pemodelan berorientasi objek yang telah bermunculan di dunia di era tahun 1990. Diantaranya metodologi booch, metodologi coad, metodologi OOSE, metodologi OMT, metodologi shlaer-mellor, metodologi wirfs-brock, dan sebagainya. Dikarenakan metodologi yang ada membawa notasi sendiri-sendiri yang mengakibatkan timbul masalah baru apabila ada sebuah kerjasama dengan *group*/perusahaan lain yang menggunakan metodologi yang berlainan sehingga pada tahun 1996 UML dijadikan standar bahasa pemodelan untuk aplikasi berorientasi objek [DSR - 03].

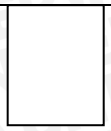
2.3.1 *Use Case Diagram*

Use case diagram menggambarkan fungsionalitas yang diharapkan dari sebuah sistem. Yang ditekankan adalah “apa” yang diperbuat sistem dan bukan “bagaimana”. Sebuah *use case* merepresentasikan sebuah interaksi antara aktor dengan sistem. *Use case* merupakan sebuah pekerjaan tertentu, misalnya *login* ke sistem, meng-*create* sebuah daftar belanja, dan sebagainya. Seorang/sebuah aktor adalah sebuah entitas manusia atau mesin yang berinteraksi dengan sistem untuk melakukan pekerjaan-pekerjaan tertentu [DSR - 03].

Sebuah *use case* dapat meng-*include* fungsionalitas *use case* lain sebagai bagian dari proses dalam dirinya. Secara umum diasumsikan bahwa *use case* yang di *include* akan dipanggil setiap kali *use case* yang meng-*include* dieksekusi secara normal. Sebuah *use case* dapat di-*include* oleh lebih dari satu *use case* lain, sehingga duplikasi fungsionalitas dapat dihindari dengan cara menarik keluar fungsionalitas yang *common*. Sebuah *use case* juga dapat meng-*extend use case* lain dengan *behaviour*-nya sendiri. Sementara hubungan generalisasi antar *use case* menunjukkan bahwa *use case* yang satu merupakan spesialisasi dari yang lain [DSR - 03].

Tabel 2.1 Keterangan Diagram Simbol *Use Case*

NO	GAMBAR	NAMA	KETERANGAN
1		Aktor / *Actor*	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat. Jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang. Biasanya dinyatakan menggunakan kata benda di awal frase nama aktor.
2		Generalisasi/ *Generalization*	Hubungan generalisasi dan spesialisasi (umum-khusus) antara dua buah *use case* dimana fungsi yang satu adalah fungsi yang lebih umum dari lainnya.
3		Menggunakan / *include / uses* <<include>>  <<uses>> 	Fungsinya atau sebagai syarat dijalankan *use case* ini. Include berarti *use case* yang ditambahkan atau dimasukkan akan selalu dipanggil sebelum atau saat *use case* tambahan dijalankan.
4		Ekstensi/ *Extend* <<extend>>	Relasi *use case* tambahan dari sebuah *use case* dimana *use case* yang ditambahkan dapat berdiri sendiri walau tanpa *use case* tambahan itu.
5		Asosiasi/ *Association*	Interaksi antara aktor dan *use case* yang berpartisipasi pada *use case*.
6		*System*	Menspesifikasikan paket yang menggambarkan sebuah sistem.

			
7		*Use Case*	Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor; biasanya dinyatakan dengan menggunakan kata kerja di awal frase nama *use case.*

Sumber: [RJB - 98]

2.3.2 Diagram Kelas

Diagram Kelas adalah suatu diagram yang memperlihatkan atau menampilkan struktur dari sebuah sistem yang akan menampilkan sistem kelas, atribut dan hubungan antara kelas. Objek diagram adalah suatu diagram yang berfungsi untuk mengatur atribut, objek dan hubungan antara *class*. Objek diagram juga dapat menampilkan struktur model sistem dalam waktu tertentu. *Class* dapat merupakan implementasi dari sebuah antar muka, yaitu *class* abstrak yang hanya memiliki metoda. Antar muka tidak dapat langsung diinstansiasikan, tetapi harus diimplementasikan dahulu menjadi sebuah *class.* Dengan demikian antar muka mendukung resolusi metoda pada saat *run-time* [DSR-03].

Class adalah sebuah spesifikasi yang jika diinstansiasi akan menghasilkan sebuah objek dan merupakan inti dari pengembangan dan desain berorientasi objek. *Class* menggambarkan keadaan (atribut/properti) suatu sistem, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut (metoda/fungsi) [DSR-03].

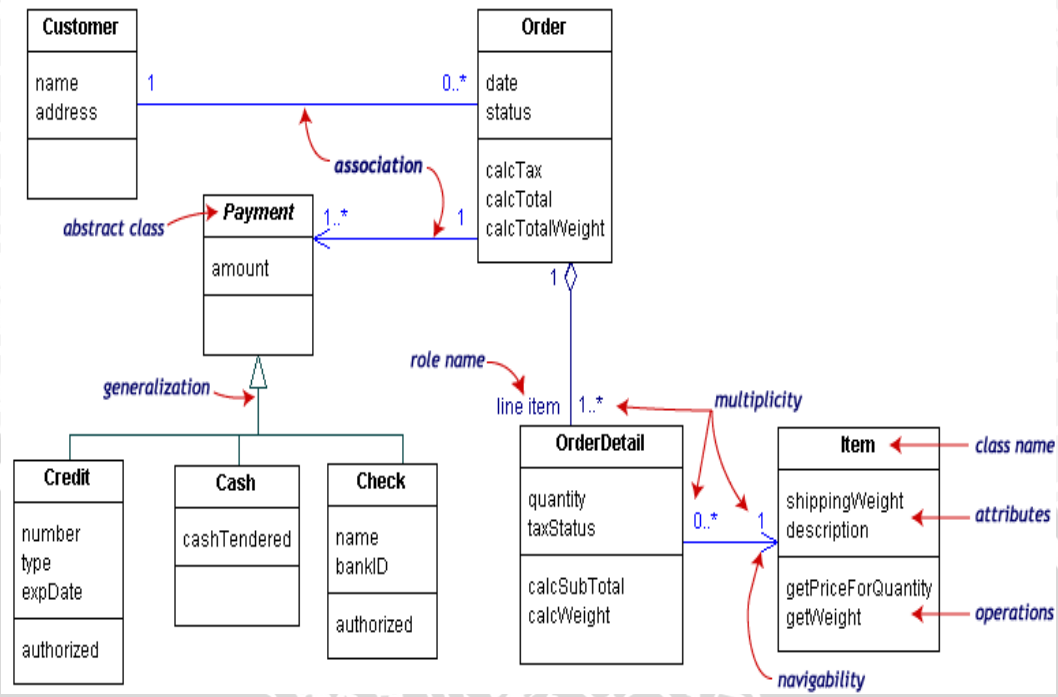
Class memiliki tiga area pokok :

1. Nama (dan *stereotype*)
2. Atribut
3. Metoda

Atribut dan metoda dapat memiliki salah satu sifat berikut :

1. *Private*, tidak dapat dipanggil dari luar *class* yang bersangkutan
2. *Protected*, hanya dapat dipanggil oleh *class* yang bersangkutan dan anak - anak yang mewarisinya

3. *Public*, dapat dipanggil oleh siapa saja



Gambar 2.1 Contoh *Class Diagram* [DSR - 03]

2.4 Pengujian Perangkat Lunak

Pengujian perangkat lunak dilakukan untuk mengetahui bahwa program apakah sudah dapat berjalan sesuai dengan kebutuhan pengguna. Ketika melakukan pengujian perangkat lunak, program harus dijalankan satu per satu setiap fungsi. Tahap selanjutnya, memeriksa hasil kasus uji untuk beberapa kesesuaian fungsi dan pengecekan apakah terdapat kesalahan atau tidak.

Proses pengujian perangkat lunak memiliki dua tujuan:

1. Untuk menunjukkan kepada pengembang dan pengguna bahwa perangkat lunak memenuhi spesifikasi kebutuhan.
2. Untuk menemukan situasi di mana perilaku dari perangkat lunak apakah terdapat sebuah kesalahan, hal yang tidak diinginkan atau tidak sesuai dengan spesifikasi kebutuhan. Perilaku tersebut seperti *system crashes*, interaksi antar sistem, kesalahan dalam komputasi dan perubahan data.

Salah satu pengujian perangkat lunak adalah *black-box testing*. Black-*box testing* juga dibutuhkan pengujian perilaku berfokus pada persyaratan fungsional

perangkat lunak. *Black-box testing* memungkinkan pengembang perangkat lunak untuk men-*set* kondisi masukkan untuk melaksanakan semua persyaratan fungsional program. *Black-box testing* menggunakan pendekatan komplementer yang memungkinkan untuk mengungkap kesalahan [PRE-01]. *Black-box testing* dapat mengungkap kesalahan dalam kategori berikut:

1. Fungsi yang hilang atau tidak benar
2. Kesalahan antarmuka
3. Kesalahan dalam struktur data
4. Kesalahan perilaku atau kinerja
5. Kesalahan inisialisasi dan terminasi

Ada banyak keuntungan dalam penggunaan metode *black-box testing*. Berikut adalah beberapa keuntungan penggunaan metode *black-box testing*:

1. Kemudahan dalam penggunaan. Karena penguji hanya perlu terfokus dengan kasus uji yang bekerja dalam aplikasi.
2. Pengembangan uji kasus yang lebih cepat. Karena penguji hanya perlu terfokus dengan berbagai perilaku melalui antar muka pengguna aplikasi.
3. Pengujian dengan cara berfokus pada masukan valid dan tidak valid dan memastikan keluaran yang diterima sesuai dengan tujuan.

Pada skripsi ini menggunakan pengujian validasi dan kompatibilitas *browser* pada pengujian *black-box testing*. Pengujian validasi digunakan untuk mengetahui apakah sistem yang dibangun sudah benar sesuai dengan yang dibutuhkan. *Item-item* yang telah dirumuskan dalam daftar kebutuhan dan merupakan hasil analisis kebutuhan akan menjadi acuan untuk melakukan pengujian validasi. Sedangkan pengujian kompatibilitas *browser* digunakan untuk mengetahui kompabilitas antar muka sistem pada tiga *browser* pada perangkat komputer.

2.5 *Framework CodeIgniter*

CodeIgniter merupakan salah satu dari sekian banyak *framework* PHP yang ada. *CodeIgniter* dikembangkan oleh Rick Ellis. Tujuan dari pembuatan *framework CodeIgniter* ini menurut *user* manualnya adalah untuk menghasilkan *framework* yang akan dapat digunakan untuk pengembangan proyek pembuatan

website lebih cepat dibandingkan dengan pembuatan *website* dengan cara koding secara manual, dengan menyediakan banyak sekali pustaka yang dibutuhkan dalam pembuatan *website*, dengan antarmuka yang sederhana dan struktur logika untuk mengakses pustaka yang dibutuhkan. *CodeIgniter* membiarkan kita untuk memfokuskan diri pada pembuatan *website* dengan meminimalkan pembuatan kode untuk berbagai tujuan pembuatan *website*. *Framework Codeigniter* mempunyai beberapa kelebihan, diantaranya [ADE - 09]:

1. Gratis.

   CodeIgniter dilisensikan dibawah lisensi Apache/BSD *style open source* lisensi, ini berarti kita dapat menggunakannya sesuai dengan keinginan kita.

2. Berjalan di PHP versi 4 dan 5.

   Sekarang ini PHP sudah mencapai versi ke 5, meskipun begitu masih banyak orang yang tetap menggunakan PHP versi 4, oleh sebab itu *CodeIgniter* dikembangkan agar tetap kompatibel dengan PHP versi 4 dan dapat dijalankan pada PHP versi 5.

3. Ringan dan cepat.

   Secara *default CodeIgniter* hanya berjalan dengan me-*load* beberapa pustaka saja, dengan demikian hanya membutuhkan *resource* yang sedikit sehingga ringan dan cepat dijalankan. Pustaka-pustaka lain yang nantinya akan digunakan bisa di-*load* sesuai dengan kebutuhan.

4. Menggunakan MVC.

   CodeIgniter menggunakan lingkungan pengembangan dengan metode *Model View Controller* (MVC) yang membedakan antara logika dan presentasi/tampilan, sehingga tugas bisa lebih mudah dipecah-pecah. Ada bagian yang khusus membuat tampilan dan bagian yang membuat *core* programnya.

5. Dokumentasi.

   Salah satu hal yang bisa dijadikan barometer apakah sebuah aplikasi benar-benar dikembangkan atau tidak bisa dilihat dari dokumentasinya. Dalam hal ini *CodeIgniter* sangat luar biasa, terdapat dokumentasi yang sangat lengkap tentang semua hal yang ada dalam *CodeIgniter*. Mulai dari langkah instalasi sampai dokumentasi fungsi-fungsi nya tersedia. Adanya dokumentasi sangat

memudahkan bagi pemula dalam mempelajari lingkungan pengembangan *website* dengan *CodeIgniter*.

6. Pustaka yang lengkap.

   CodeIgniter dilengkapi dengan berbagai pustaka siap pakai untuk berbagai kebutuhan, misalnya saja koneksi *database, email, session* dan *cookies,* keamanan, manipulasi gambar dan banyak lagi.

2.6 *Service Oriented Architecture* (SOA)

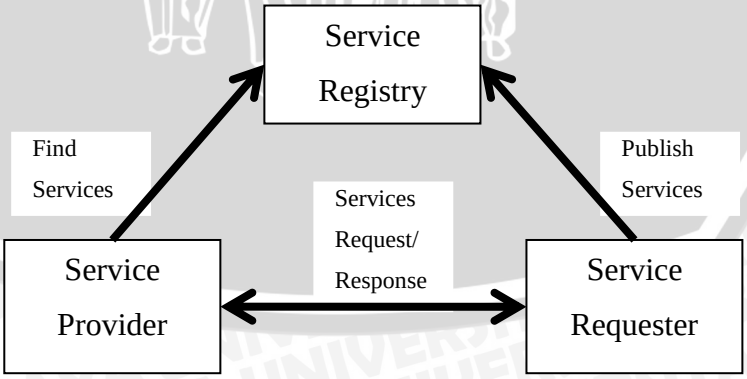
SOA merupakan sebuah model arsitektur untuk membangun aplikasi perangkat lunak yang menggunakan beberapa *service* yang tersedia di sebuah jaringan seperti Web. SOA merupakan konsep dasar yang melandasi implementasi dari sebuah *web services*. Model sistem dengan menerapkan SOA memiliki berbagai macam layanan yang disediakan untuk pengguna. *Service* merupakan sekumpulan aktifitas yang disusun sedemikian rupa dan berfungsi untuk sebuah tujuan [PSB - 12].

Seperti yang didefinisikan di atas, SOA adalah suatu cara perancangan aplikasi dengan menggunakan komponen-komponen atau pelayanan yang sudah ada. Dengan kata lain, suatu aplikasi dibangun secara modular. Sebenarnya pendekatan modular ini bukanlah sesuatu yang baru. Teknik-teknik pemrograman masa kini seperti *object oriented programming*, telah mengedepankan pendekatan modular dalam pembangunan aplikasi. Namun yang membuat SOA berbeda adalah komponen atau *service* tersebut dibangun dan berinteraksi satu sama lain secara bebas dan lepas (*loose coupled*). Dengan bersifat *loose coupled*, sebuah *service* dapat dipanggil oleh program atau *service* lainnya tanpa program pemanggil tersebut perlu memperhatikan dimana lokasi *service* yang dipanggil berada pada *platform* atau teknologi apa yang digunakan oleh *service* tersebut. *Loose coupling* sangat penting bagi SOA karena dengan demikian pemanggilan sebuah *service* oleh *service* lainnya dapat dilakukan pada saat *run-time* [PSB - 12].

SOA memiliki beberapa kelebihan, diantaranya adalah sebagai berikut [PSB - 12]:

1. Dapat menyatukan berbagai sistem yang memiliki *platform* berbeda, seperti J2EE dan .NET. Hal ini dikarenakan dengan adanya pendekatan ini, pengembang akan memilih untuk membangun sebuah *layer* di atas sistem yang dapat saling berkomunikasi dengan pesan yang sudah distandardisasi, misalnya menggunakan teknologi XML. Dalam sudut pandang SOA, kedua sistem itu masing-masingnya akan dianggap sebagai *service*.
2. Tahan terhadap perubahan. Perusahaan atau organisasi besar seringkali berubah struktur untuk meningkatkan efisiensi dan kinerja. Akibatnya, perangkat lunak juga terkena imbas untuk menyesuaikan diri terhadap proses bisnis yang baru. Permodelan perangkat lunak dengan SOA akan mengurangi *effort* untuk modifikasi perangkat lunak tersebut. Sebab, karena seluruh *logic* dari sistem sudah terpartisi secara bersih menjadi sekumpulan *services*, kita hanya perlu menyusun ulang seluruh *service* tersebut dan jika perlu menambahkan yang baru. Hal ini jelas mengurangi biaya.

Web Services seringkali dikaitkan atau bahkan disamakan dengan SOA. Namun sebenarnya keduanya adalah hal yang sangat berbeda. SOA adalah sebuah konsep untuk pengembangan perangkat lunak, sementara *Web Services* adalah sebuah aplikasi web yang berinteraksi dengan aplikasi *web* lainnya untuk pertukaran data. Pembangunan SOA tidak harus menggunakan *Web Services*, sebab ada bermacam-macam teknologi lain yang memungkinkan, tapi menggunakan *Web Services* untuk membangun sebuah sistem SOA adalah langkah yang baik [PSB - 12].



Gambar 2.2 Operasi Dasar SOA [SKG - 08]

Terdapat tiga bagian pada konsep SOA, yaitu *Service Provider*, *Service Registry*, dan *Service Requester*. *Service Provider* adalah penyedia *service*,

Service Registry adalah bagian yang menyimpan informasi tentang *service* yang tersedia, dan *Service Requester* adalah *user* atau pengguna *service*.

Service Provider mendefinisikan *service* yang disediakan dengan menggunakan WSDL (*Web Service Description Language*) dan mendaftarkan pada *Service Registry*. *Service Requester* mencari informasi *service* pada *Service Registry*, informasi yang disediakan adalah daftar *service*, lokasi *service* dan tata cara berkomunikasi. Setelah mendapatkan informasi dari *Service Registry*, *Service Requester* melakukan koneksi langsung ke *Service Provider* [SKG - 08].

2.7 *Web Services*

Web Services adalah sebuah teknik pemrograman dimana sebuah *service* menggunakan standar-standar berbasis XML dalam menjelaskan antar muka dan *protocol* yang harus digunakan untuk memanggil *service* tersebut [PSB - 12].

Gambar 2.6 merupakan blok bangunan *web service* yang menyediakan fasilitas komunikasi jarak jauh antara dua aplikasi.

a. Layer 1 : protokol internet standar yang digunakan sebagai sarana transportasi adalah HTTP dan TCP/IP.
b. Layer 2 : *Simple Object Access Protocol* (SOAP) berbasiskan XML dan digunakan untuk pertukaran informasi antar sekelompok layanan.
c. Layer 3 : *Web service Definition Language* (WSDL) digunakan untuk mendiskripsikan *attribute* layanan.
d. Layer 4 : *Universal Description, Discovery and Integration*, yang mana merupakan direktori pusat untuk deskripsi layanan.

2.8 *Web Service Definition Language* (WSDL)

WSDL merupakan sebuah bahasa berbasis XML yang digunakan untuk mendefinisikan *web service* dan menggambarkan bagaimana cara untuk mengakses *web service* tersebut. Fungsi utama WSDL dalam *web service* adalah untuk mengotomasi mekanisme komunikasi *business-to-business* dalam *web service* melalui protokol internet. WSDL merupakan representasi kontrak antara requestor dan *provider*-nya. Secara teknis merupakan representasi kontrak antara

kode klien dan kode di *server*. Dengan menggunakan WSDL klien dapat memanfaatkan fungsi-fungsi publik yang disediakan oleh *server* [DEV - 11].

Didalam dokumen WSDL terdapat lima elemen utama antara lain.

1. Elemen <type>, berfungsi untuk mendefinisikan tipe data-tipe data yang digunakan dalam pesan.
2. Elemen <message>, berfungsi untuk mendefinisikan format dari sebuah pesan. Pesan digunakan sebagai struktur masukan (*input*) atau keluaran (*output*) bagi operasi.
3. Elemen <portType>, berfungsi untuk mendefinisikan sekumpulan operasi-operasi. Tiap-tiap elemen <operation> mendefinisikan sebuah operasi dan pesan masukan atau keluaran yang berkaitan dengan operasi tersebut.
4. Elemen <binding>, berfungsi untuk memetakan operasi-operasi dan pesan yang terdefinisikan pada *port type* ke protokol tertentu.
5. Elemen <service>, berfungsi untuk mendefinisikan sekumpulan port-port yang saling berhubungan. Elemen <port> memetakan *binding* ke lokasi dari sebuah *web service* [PTK - 05].

2.9 *Simple Object Access Protocol* (SOAP)

SOAP adalah sebuah standarisasi dasar dari protokol komunikasi untuk saling bertukar informasi terstruktur di antara aplikasi-aplikasi yang berjalan.. SOAP merupakan gabungan antara HTTP dengan XML karena SOAP umumnya menggunakan protokol HTTP sebagai sarana *transport* datanya dan data akan dipertukarkan ditulis dalam format XML. Karena SOAP mengunakan HTTP dan XML maka SOAP memungkinkan pihak-pihak yang mempunyai *platform*, sistem operasi dan perangkat lunak yang berbeda dapat saling mempertukarkan datanya. SOAP mengatur bagaimana *request* dan respon dari suatu *web service* bekerja [DEV - 11]

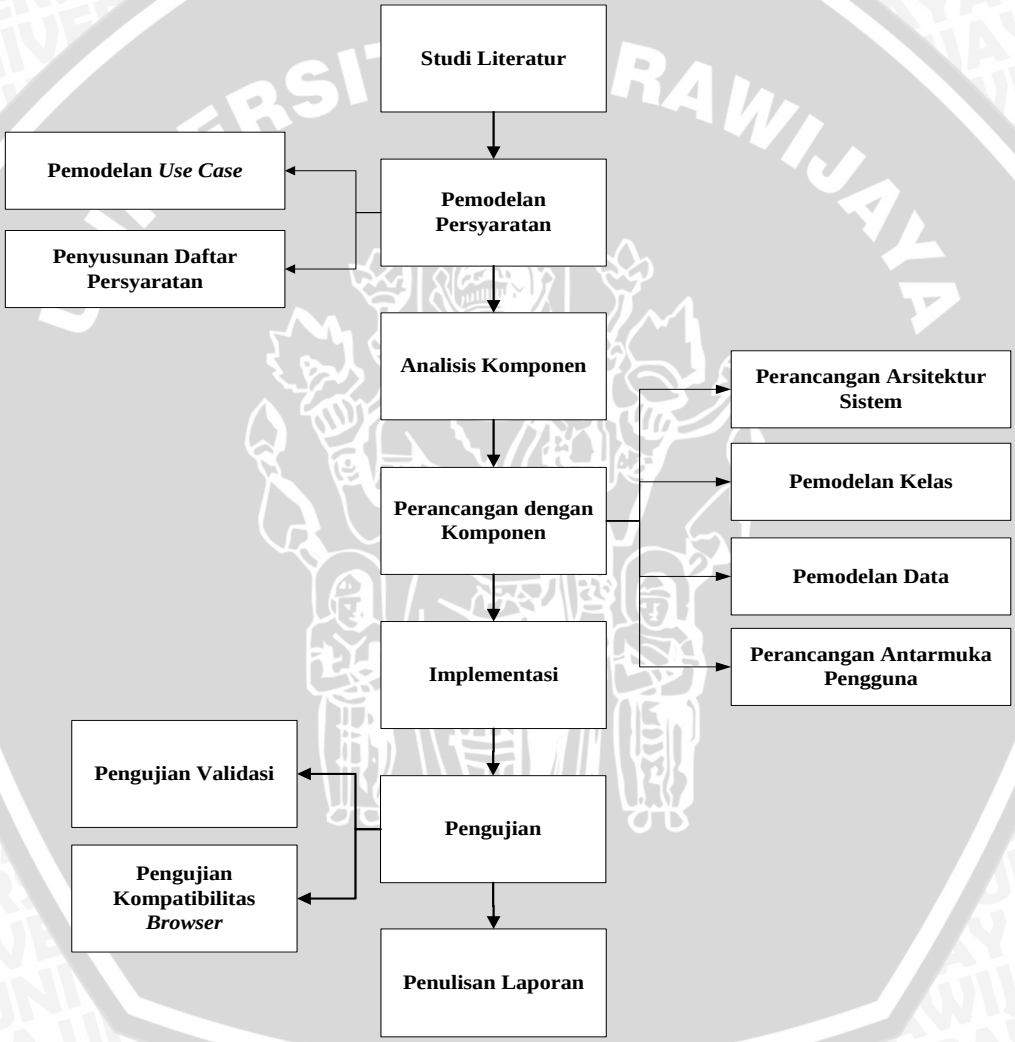
2.10 NuSOAP

NuSOAP adalah *library* yang digunakan untuk membangun *web service* berbasis SOAP yang ditulis dengan menggunakan bahasa PHP. NuSOAP

merupakan penulisan ulang dari SOAPx4, yang disediakan oleh NuSphere dan Dietrich Ayala. Ini adalah satu set dari kelas PHP, tidak ada ekstensi PHP diperlukan, yang memungkinkan *developer* untuk membuat dan mengkonsumsi layanan *web* berdasarkan SOAP 1.1, WSDL 1.1 dan HTTP 1.0/1.1. Penerapan NuSOAP PHP *Web Services* terintegrasi dengan PhpED – NuSphere's PHP IDE. Panggilan ke metode kelas NuSOAP dapat dihasilkan oleh NuSOAP Klien dalam PhpED dan dengan NuSOAP *wizard*. NuSOAP juga digunakan pada PhpDOCK *Currency Converter* [ANM - 12].

BAB III
METODOLOGI PENELITIAN

Bab ini membahas mengenai prosedur-prosedur dan kegiatan-kegiatan yang akan dilakukan dalam pengerjaan skripsi. Prosedur-prosedur tersebut meliputi studi literatur, pemodelan persyaratan, analisis komponen, perancangan dengan komponen, implementasi dan pengujian dalam pembuatan sistem ini yang akan dibuat.



Gambar 3.1 Tahap-Tahap Pengerjaan Skripsi

Pengerjaan skripsi dilakukan sesuai dengan tahap-tahap yang telah ditentukan. Tahap-tahap pengerjaan skripsi pada Gambar 3.1 dijelaskan pada sub-sub bab berikut:

3.1 Studi Literatur

Studi literatur merupakan pemilihan dan pembelajaran literatur yang digunakan dalam penyususunan dasar teori untuk menunjang penyusunan skripsi. Literatur yang digunakan bersumber dari buku dan *internet.*.

Tahap-tahap studi literatur yang pertama adalah memahami permasalahan yang digunakan dalam pokok bahasan skripsi ini dengan melakukan observasi dan mencari informasi ke beberapa tempat seperti DPPKAD, Kelurahan/Kecamatan dan Bank Jatim. Selain melakukan observasi ke beberapa tempat tersebut, penulis melakukan pencarian dan pengumpulan literatur untuk menunjang pengerjaan skripsi dari media *internet* dan buku.

Dasar teori disusun setelah referensi-referensi pendukung penulisan skripsi ini terkumpul. Referensi pendukung yang digunakan dalam skripsi ini adalah model *reuse-oriented software engineering.* Pengembangan sistem yang dirancang meliputi pemodelan persyaratan, analisis komponen, perancangan dengan komponen, implementasi, dan pengujian.

3.2 Pemodelan Persyaratan

Pemodelan persyaratan digunakan untuk memahami dan menyusun persyaratan-persyaratan yang dibutuhkan oleh sistem. Pemodelan persyaratan merupakan langkah awal untuk merancang sistem yang akan dibuat sesuai kebutuhan. Sistem yang baik dan sesuai dengan kebutuhan pengguna sangat bergantung kepada keberhasilan dalam memodelkan persyaratan. Pemodelan persyaratan digambarkan dengan pemodelan *use case*. Tahap-tahap pemodelan *use case* dijelaskan dengan mengidentifikasi aktor, mengidentifikasi *use case* dan diagram *use case,* dan spesifikasi *use case*.

Untuk mempermudah dalam mengidentifikasi persyaratan terdapat dua jenis persyaratan. Dua jenis persyaratan tersebut adalah persyaratan fungsional dan persyaratan nonfungsional.

3.3 Analisis Komponen

Analisa komponen dilakukan untuk mencari komponen-komponen yang sesuai digunakan dalam mengimplementasikan persyaratan yang telah dirancang.

Analisis komponen sangat diperlukan dalam memilih komponen-komponen yang benar-benar sesuai dengan kebutuhan. Tetapi bisa saja komponen yang tidak sesuai dengan kebutuhan mengalami tahap modifikasi dengan tujuan ingin memaksimalkan dalam pembuatan sistem. Hal tersebut menyebabkan sistem dapat digunakan lebih baik dan lebih lengkap. Tetapi dalam tahap modifikasi membutuhkan waktu yang tidak sedikit sehingga dalam skripsi ini tidak melakukan tahap modifikasi melainkan memilih dan menggunakan komponen yang ada dan sesuai dengan kebutuhan.

3.4 Perancangan dengan Komponen

Pada tahap ini komponen-komponen yang telah ditentukan akan digunakan dalam pembuatan sistem. Tahap selanjutnya adalah memodelkannya ke dalam bentuk arsitektur sistem, pemodelan kelas, pemodelan data, dan perancangan antar muka pengguna.

3.4.1 Perancangan Arsitektur Sistem

Perancangan arsitektur sistem digunakan untuk menggambarkan sistem yang akan dibuat secara keseluruhan. Hal ini bertujuan agar dapat mengetahui proses sistem yang dibuat nantinya yang sesuai dengan kebutuhan. Arsitektur sistem digambarkan secara detail mulai dari pengguna mengakses sistem sampai hasil yang dikeluarkan sistem dikembalikan ke pengguna.

3.4.2 Pemodelan Kelas

Pemodelan kelas dimodelkan dalam bentuk diagram kelas. Diagram kelas memberi gambaran tentang kelas-kelas yang akan dibuat, relasi-relasi yang terdapat di dalamnya, dan komponen yang digunakan. Kelas-kelas yang telah teridentifikasi kemudian mengindentifikasi hubungan antar kelas.

3.4.3 Pemodelan Data

Pemodelan data digambarkan dalam bentuk *entity-relationship diagram* yang akan menggambarkan basis data yang akan dibuat dan digunakan oleh sistem. Pemodelan data mengandung relasi antar entitas yang masing-masing entitas mempunyai atribut masing-masing.

3.4.4 Perancangan Antarmuka Pengguna

Antarmuka sistem merupakan jembatan interaksi antara pengguna dengan sistem sehingga antar muka perlu dibuat sederhana sehingga dapat diterima pengguna dengan mudah. Perancangan antarmuka bertujuan untuk menggambarkan antarmuka sistem yang akan dibuat sehingga nantinya dapat dibuat dengan mudah.

3.5 Implementasi

Implementasi dapat direalisasikan dengan baik setelah perancangan perasyaratan selesai dilakukan. Implementasi sistem ini dilakukan dengan mengacu pada identifikasi persyaratan dan perancangan sistem. Implementasi sistem ODP ini meliputi :

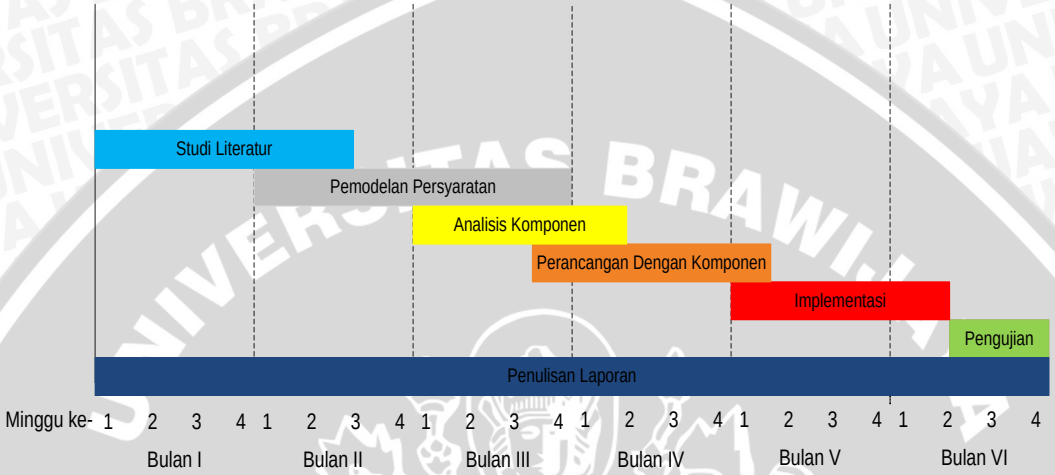
1. Pembuatan sistem ODP sesuai perancangan yang telah dibuat.
2. Pembuatan basis data sesuai dengan pemodelan data.
3. Data-data penduduk wajib pajak yang telah diperoleh akan dimasukan ke dalam basis data.
4. Pembuatan antarmuka dibangun sesuai dengan perancangan antarmuka pengguna.

3.6 Pengujian

Pengujian sistem pada penelitian ini dilakukan agar dapat menunjukan bahwa sistem telah mampu bekerja sesuai dengan spesifikasi persyaratan sistem yang telah melandasinya. Pengujian dilakukan dengan dua tahap, yaitu pengujian validasi dan kompatibilitas *browser*. Pengujian validasi berfungsi untuk menguji setiap fungsionalitas sistem sesuai dengan pemodelan persyaratan yang dibuat. Sedangkan kompatibilitas *browser* berfungsi untuk menguji antar muka sistem sesuai dengan perancangan antar muka pengguna dan menguji apakah *browser* dapat menampilkan seluruh antar muka secara lengkap. Maksud dari dapat menampilkan antar muka secara lengkap, yaitu dapat menampilkan keterangan-keterangan pendukung, seperti dapat menampilkan penjelasan dari kotak *login* (*placeholder*), dapat menampilkan data formulir pengajuan penanganan PBB, dan lain-lain.

3.7 Penulisan Laporan

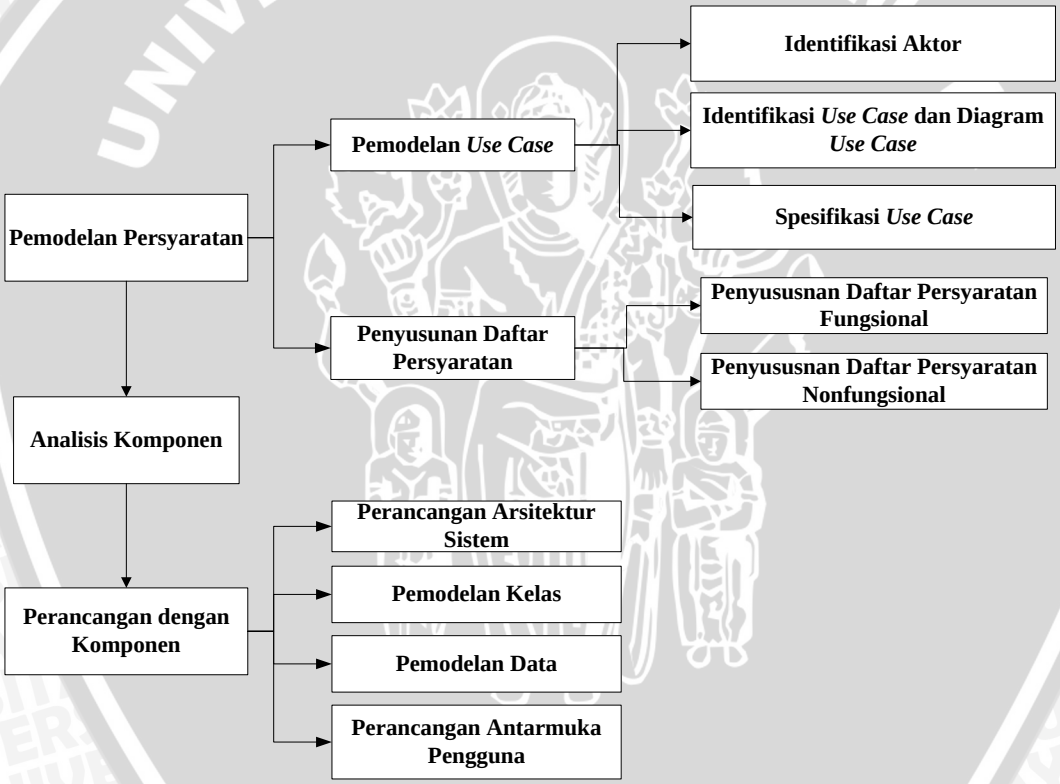
Laporan penelitian ditulis setelah semua proses pengerjaan skripsi dilalui. Laporan berisi dokumentasi perancangan sistem beserta saran untuk memberikan pertimbangan dan pengembangan sistem ODP selanjutnya. Dalam pengerjaan skripsi dibutuhkan jadwal (*timeline*) agar pengerjaan dapat selesai tepat pada waktunya. Jadwal (*timeline*) pengerjaan skripsi ini digambarkan pada Gambar 3.1.



Gambar 3.2 *Timeline* Pengerjaan Skripsi

BAB IV
IDENTIFIKASI PERSYARATAN DAN PERANCANGAN

Bab ini membahas mengenai proses pemodelan persyaratan, proses analisis komponen dan proses perancangan dengan komponen. Pemodelan persyaratan terdiri dari dua tahap yaitu melakukan pemodelan *use case* dan penyusunan daftar persyaratan. Analisis komponen membahas komponen-komponen apa saja yang dibutuhkan atau digunakan sesuai dengan spesifikasi persyaratan. Proses perancangan dengan komponen meliputi arsitektur sistem, pemodelan kelas, pemodelan data, dan perancangan antar muka pengguna. Gambar 4.1 mengilustrasikan tahap-tahap dari perancangan sistem ODP.



Gambar 4.1 Diagram Perancangan Sistem ODP

Hasil dari setiap tahapan pada Gambar 4.1 dijelaskan dalam sub-sub bab berikut:

4.1 Pemodelan Persyaratan

Pemodelan persyaratan ini bertujuan untuk menggambarkan persyaratan-persyaratan apa saja yang harus disediakan oleh sistem agar dapat memenuhi

kebutuhan pengguna. Proses identifikasi persyaratan diawali dengan pemodelan *use case* dan penyususnan daftar persyaratan.

4.1.1 Pemodelan *Use Case*

Pemodelan *use* case digunakan untuk memodelkan sistem dan peran aktor. Aktor harus diidentifikasi terlebih dahulu untuk mengetahui siapa saja yang akan berhubungan dengan sistem secara langsung. Tahapan-tahapan pemodelan *use case* antara lain identifikasi aktor, diagram *use case* dan spesifikasi *use case*.

4.1.1.1 Identifikasi Aktor

Tahap ini mempunyai tujuan untuk melakukan identifikasi terhadap aktor yang akan berinteraksi dengan sistem. Penjelasan dari masing-masing identifikasi aktor dapat dilihat pada tabel 4.1.

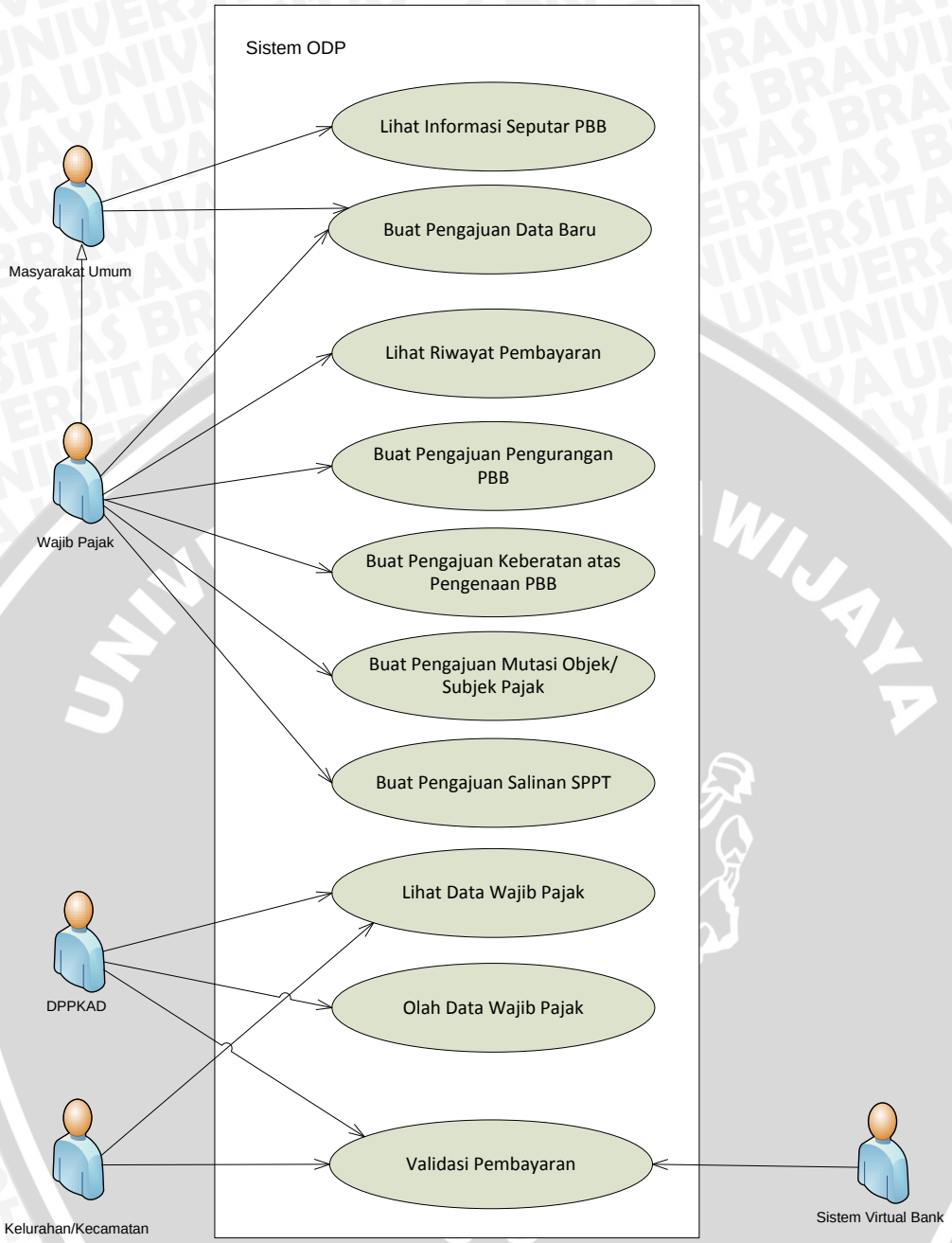
Tabel 4.1 Identifikasi Aktor

Aktor	Deskripsi
Masyarakat Umum	Masyarakat umum baik yang bukan wajib pajak atau yang merupakan wajib pajak yang dapat mengakses sistem ODP. Masyarakat umum ini dapat memanfaatkan beberapa fitur yang diberikan oleh sistem seperti melihat segala informasi tentang PBB mulai dari penjelasan tentang PBB, pengumuman seputar PBB dan belajar mengani aturan-aturan PBB. Masyarakat umum juga bisa membuat pengajuan data objek pajak baru melalui sistem.
Wajib Pajak	Wajib pajak dapat memanfaatkan fitur yang dimiliki sistem dengan melalui proses autentifikasi. Setiap Wajib pajak dapat melakukan autentikasi ke sistem dengan memasukkan Nomor Objek Pajak (NOP). Setelah proses autentikasi berhasil Wajib pajak dapat memanfaatkan beberapa fitur yang dimiliki oleh sistem seperti melihat riwayat pembayaran yang telah dilakukan dan dapat membuat beberapa pengajuan penangan PBB, seperti pengajuan pengurangan pembayaran PBB, pengajuan mutasi/pembetulan objek/subjek PBB, pengajuan keberatan atas PBB yang dikenakan, dan pengajuan salinan SPPT dengan mengisi formulir yang tersedia pada sistem.
DPPKAD	Merupakan aktor pengguna sistem di mana aktor tersebut berfungsi sebagai pusat pengelola sistem dalam melakukan pengolahan data, seperti mengubah data wajib pajak, melihat data wajib pajak, dan memasukkan data wajib pajak baru. Aktor juga dapat mem-validasi pembayaran apabila wajib pajak melakukan pembayaran di DPPKAD. Aktor ini memiliki satu akun khusus untuk

	melakukan autentikasi agar bisa mengolah data wajib pajak pada sistem.
Kelurahan/Kecamatan	Merupakan aktor pengguna sistem di mana aktor dapat melihat data wajib pajak dan mem-validasi pembayaran apabila wajib pajak melakukan pembayaran di Kelurahan/Kecamatan. Aktor ini memiliki satu akun khusus untuk bisa mengakses sistem.
Petugas Bank	Merupakan aktor pengguna sistem di mana aktor tersebut berfungsi sebagai pengelola sistem virtual bank. Sistem virtual Bank merupakan suatu sistem sederhana yang berfungsi untuk mem-validasi pembayaran apabila wajib pajak melakukan pembayaran di Bank atau hanya sekedar melakukan pelaporan pembayaran saja. Bank yang digunakan untuk pembayaran PBB dan mengeluarkan bukti pemabayaran sah untuk PBB di Kabupaten Jombang adalah Bank Jatim.

4.1.1.2 Identifikasi *Use* Case dan Diagram *Use Case*

Setiap aktor yang telah diidentifikasi pada sub bab sebelumnya kemudian akan dispesifikasikan aksi atau perannya terhadap sistem. Peran aktor terhadap sistem ini dirancang dalam diagram *use case*. Diagram *use case* adalah salah satu diagram untuk memodelkan aspek perilaku sistem. Diagram *use case* menunjukkan sekumpulan *use case*, aktor, dan hubungannya. Gambar 4.2 merupakan pemodelan diagram *use case* pada sistem ODP.

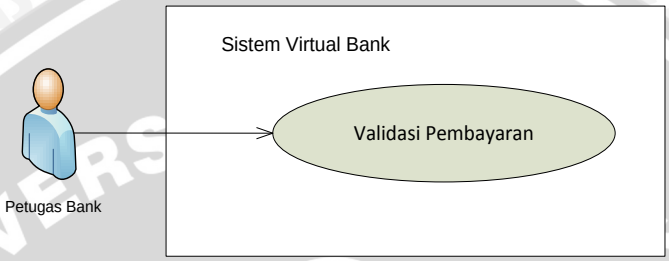


Gambar 4.2 Diagram *Use Case* Sistem ODP

Apabila wajib pajak melakukan pembayaran di DPPKAD atau Kelurahan atau Kecamatan maka perlu melakukan pelaporan ke pihak Bank untuk mendapatkan bukti pembayaran yang sah. Wajib pajak juga bisa melakukan pembayaran melalui Bank apabila ingin langsung mendapatkan bukti pembayaran yang sah.

Dalam skripsi ini pihak Bank dalam melakukan validasi pembayaran PBB tidak melalui sistem ODP melainkan melalui sistem virtual Bank. Sistem Virtual

Bank ini menggambarkan simulasi dari sistem Bank yang sesungguhnya. Sistem virtual Bank merupakan model atau penyederhanaan dari sebuah bank yang hanya mendemonstrasikan perilaku sistem untuk validasi pembayaran. Oleh karena itu pihak Bank tidak perlu mengakses sistem ODP yang memiliki banyak fitur dalam pengolahan data PBB karena pihak Bank hanya mempunyai peran atau terfokus pada validasi pembayaran PBB saja. Gambar 4.3 merupakan diagram *use case* dari sistem virtual Bank.



Gambar 4.3 Diagram *Use Case* Sistem Virtual Bank

4.1.1.3 Spesifikasi *Use Case*

Spesifikasi *use case* digunakan untuk memberikan penjelasan dari diagram *use case.* Spesifikasi *use case* pada bab ini akan diberikan uraian nama *use case*, aktor yang berhubungan dengan *use case* tersebut, tujuan dari *use case*, deskripsi global tentang *use case*, dan aktor. Untuk lebih detailnya masing-masing *use case* pada gambar di atas akan dijabarkan pada tabel-tabel di bawah ini.

Tabel 4.2 Spesifikasi *Use Case* Lihat Informasi Seputar PBB

Spesifikasi *Use Case*	
Nama	Lihat Informasi Seputar PBB
Tujuan	Untuk melihat semua infromasi seputar PBB
Deskripsi	*Use case* ini menjelaskan pengguna dapat melihat semua informasi seputar PBB mulai dari penjelasan tentang PBB, pengumuman seputar PBB dan belajar mengenai aturan-aturan PBB.
Aktor	Masyarakat umum dan wajib pajak

Untuk penjelasan lebih detail spesifikasi *use case* lihat informasi seputar pajak dapat dilihat pada lampiran tabel 2.

Tabel 4.3 Spesifikasi *Use Case* Buat Pengajuan Objek Pajak Baru

Spesifikasi *Use Case*	
Nama	Buat pengajuan objek pajak baru
Tujuan	Untuk membuat pengajuan data pajak baru

Deskripsi	*Use case* ini menjelaskan pengguna dapat membuat pengajuan objek pajak baru dengan mengisi formulir yang tersedia pada sistem.
Aktor	Masyarakat umum dan wajib pajak

Untuk penjelasan lebih detail spesifikasi *use case* buat pengajuan objek pajak baru dapat dilihat pada lampiran tabel 3.

Tabel 4.4 Spesifikasi *Use Case* Lihat Riwayat Pembayaran

Spesifikasi *Use Case*	
Nama	Lihat riwayat pembayaran
Tujuan	Untuk melihat riwayat pembayaran PBB yang sudah dilakukan
Deskripsi	*Use case* ini menjelaskan pengguna dapat melihat riwayat pembayaran PBB selama lima tahun terakhir.
Aktor	Wajib Pajak

Untuk penjelasan lebih detail spesifikasi *use case* lihat riwayat pembayaran dapat dilihat pada lampiran tabel 4.

Tabel 4.5 Spesifikasi *Use Case* Buat Pengajuan Pengurangan PBB

Spesifikasi *Use Case*	
Nama	Buat pengajuan pengurangan PBB
Tujuan	Untuk membuat pengajuan pengurangan PBB
Deskripsi	*Use case* ini menjelaskan aktor dapat mengajukan pengurangan PBB setelah melakukan autentikasi.
Aktor	Wajib Pajak

Untuk penjelasan lebih detail spesifikasi *use case* buat pengajuan pengurangan PBB dapat dilihat pada lampiran tabel 5.

Tabel 4.6 Spesifikasi *Use Case* Buat Pengajuan Keberatan Atas Pengenaan PBB

Spesifikasi *Use Case*	
Nama	Buat pengajuan keberatan atas pengenaan PBB
Tujuan	Untuk membuat pengajuan keberatan atas pengenaan PBB
Deskripsi	*Use case* ini menjelaskan aktor dapat mengajukan keberatan atas pengenaan PBB apabila PBB yang dikenakan tidak sesuai dengan kondisi yang ada. Pengajuan keberatan atas pengenaan PBB ini dapat dilakukan setelah melakukan autentikasi.
Aktor	Wajib Pajak

Untuk penjelasan lebih detail spesifikasi *use case* pengajuan keberatan atas pengenaan PBB dapat dilihat pada lampiran tabel 6.

Tabel 4.7 Spesifikasi *Use Case* Buat Pengajuan Mutasi PBB

Spesifikasi *Use Case*	
Nama	Buat pengajuan mutasi PBB
Tujuan	Untuk membuat pengajuan mutasi PBB
Deskripsi	*Use case* ini menjelaskan aktor dapat mengajukan mutasi atau pembetulan objek/subjek pajak. Pengajuan mutasi PBB ini dapat dilakukan setelah melakukan autentikasi.
Aktor	Wajib Pajak

Untuk penjelasan lebih detail spesifikasi *use case* pengajuan mutasi PBB dapat dilihat pada lampiran tabel 7.

Tabel 4.8 Spesifikasi *Use Case* Buat Pengajuan Salinan SPPT

Spesifikasi *Use Case*	
Nama	Buat pengajuan salinan SPPT
Tujuan	Untuk membuat pengajuan salinan SPPT
Deskripsi	*Use case* ini menjelaskan aktor dapat mengajukan salinan SPPT apabila terjadi kehilangan SPPT atau yang lainnya. Pengajuan salinan SPPT ini dapat dilakukan setelah melakukan autentikasi.
Aktor	Wajib Pajak

Untuk penjelasan lebih detail spesifikasi *use case* pengajuan salinan SPPT dapat dilihat pada lampiran tabel 8.

Tabel 4.9 Spesifikasi *Use Case* Lihat Data Wajib Pajak

Spesifikasi *Use Case*	
Nama	Lihat data wajib pajak
Tujuan	Untuk melihat data wajib pajak
Deskripsi	*Use case* ini menjelaskan aktor dapat melihat data pembayaran wajib pajak, data pengajuan objek pajak baru, data pengajuan pengurangan PBB, data pengajuan keberatan atas pengenaan PBB, dan data pengajuan salinan SPPT. Lihat data wajib pajak dapat dilakukan setelah melakukan autentikasi.
Aktor	DPPKAD/Kelurahan/Kecamatan

Untuk penjelasan lebih detail spesifikasi *use case* lihat data wajib pajak dapat dilihat pada lampiran tabel 9.

Tabel 4.10 Spesifikasi *Use Case* Olah Data Wajib Pajak

Spesifikasi *Use Case*	
Nama	Olah data wajib pajak
Tujuan	Untuk mengolah data wajib pajak
Deskripsi	*Use case* ini menjelaskan aktor dapat mengolah data wajib pajak yang meliputi menambah data wajib pajak

	baru dan mengubah data wajib pajak. Olah data wajib pajak dapat dilakukan setelah melakukan autentikasi.
Aktor	DPPKAD

Untuk penjelasan lebih detail spesifikasi *use case* olah data wajib pajak dapat dilihat pada lampiran tabel 10.

Tabel 4.11 Spesifikasi *Use Case* Validasi Pembayaran

Spesifikasi *Use Case*	
Nama	Validasi Pembayaran
Tujuan	Untuk melakukan validasi pembayaran PBB
Deskripsi	*Use case* ini menjelaskan aktor dapat melakukan validasi pembayaran PBB. Validasi pembayaran dapat dilakukan setelah melakukan autentikasi.
Aktor	DPPKAD/Kelurahan/Kecamatan

Untuk penjelasan lebih detail spesifikasi *use case* validasi pembayaran dapat dilihat pada lampiran tabel 11.

Tabel 4.12 Spesifikasi *Use Case* Validasi Pembayaran Dengan Sistem Virtual Bank

Spesifikasi *Use Case*	
Nama	Validasi Pembayaran
Tujuan	Untuk melakukan validasi pembayaran PBB dengan sistem virtual Bank
Deskripsi	*Use case* ini menjelaskan aktor dapat melakukan validasi pembayaran pajak yang dilakukan oleh wajib pajak. Sistem ini menggunakan arsitektur SOA dengan menggunakan *web service Simple Object Access Protocol* (SOAP) dalam mengakses data wajib pajak.
Aktor	Petugas Bank

Untuk penjelasan lebih detail spesifikasi *use case* validasi pembayaran dengan sistem virtual Bank dapat dilihat pada lampiran tabel 12.

4.1.2 Penyusunan Daftar Persyaratan

Daftar persyaratan terdiri dari persyaratan fungsional dan persyaratan non-fungsional yang harus disediakan oleh sistem. Daftar persyaratan ini disebut dengan *Software Requirement Specification* (SRS) yang didasarkan pada diagram *use case* yang telah dirancang pada sub bab 4.1.1.2. Daftar persyaratan fungsional dapat dilihat pada tabel 4.13.

Tabel 4.13 Daftar Persyaratan Fungsional

Nomor SRS	**Persyaratan**	**Aktor**	**Nama *Use Case***
SRS_001_01	Sistem dapat	Masyarakat Umum	Lihat Seputar

	menampilkan informasi seputar PBB		PBB
SRS_001_02	Aktor dapat membuat pengajuan objek pajak baru	Masyarakat Umum	Buat Pengajuan Objek Pajak Baru
SRS_001_03	Sistem dapat mengirim pengajuan objek pajak baru	Masyarakat Umum	
SRS_002_01	Sistem dapat menampilkan riwayat pembayaran	Wajib Pajak	Lihat Riwayat Pembayaran
SRS_002_02	Aktor dapat membuat dan mengirim pengajuan pengurangan PBB	Wajib Pajak	Buat Pengajuan Pengurangan PBB
SRS_002_03	Aktor dapat membuat dan mengirim pengajuan keberatan atas pengenaan PBB	Wajib Pajak	Buat Pengajuan Keberatan Atas Pengenaan PBB
SRS_002_04	Aktor dapat membuat dan mengirim pengajuan mutasi objek/subjek PBB	Wajib Pajak	Buat Pengajuan Mutasi Objek/Subjek PBB
SRS_002_05	Aktor dapat membuat dan mengirim pengajuan salinan SPPT	Wajib Pajak	Buat Pengajuan Salinan SPPT
SRS_003_01	Aktor dapat melihat data wajib pajak	DPPKAD	Lihat Data Wajib Pajak
SRS_003_02	Aktor dapat menambah data wajib pajak baru	DPPKAD	Olah Data Wajib Pajak
SRS_003_03	Aktor dapat mengubah data wajib pajak	DPPKAD	
SRS_003_04	Aktor mem-validasi pembayaran	DPPKAD	Validasi Pembayaran
SRS_004_01	Aktor dapat melihat data wajib pajak	Kelurahan/Kecamatan	Lihat Data Wajib Pajak
SRS_004_02	Aktor mem-validasi pembayaran	Kelurahan/Kecamatan	Validasi Pembayaran
SRS_005_01	Aktor mem-validasi pembayaran	Petugas Bank	Validasi Pembayaran

Persyaratan non-fungsional juga harus disediakan oleh sistem. Persyaratan non-fungsional merupakan batasan layanan atau fungsi yang ditawarkan sistem

seperti batasan pengembangan proses dan standarisasi. Daftar persyaratan non-fungsional dapat dilihat pada tabel 4.14.

Tabel 4.14 Daftar Persyaratan Non-Fungsional

Parameter	Deskripsi Persyaratan
Usability	Sistem harus dapat digunakan dengan mudah oleh pengguna sehingga antar muka harus dibuat sesederhana mungkin agar dapat mudah dilihat oleh pengguna. Oleh karena itu, antar muka harus didesain sesuai dengan perancangan antar muka pengguna. Selain didesain sesuai perancangan antar muka pengguna, antar muka harus dapat dilihat oleh pengguna secara lengkap atau dapat menampilkan keterangan-keterangan pendukung, seperti dapat menampilkan penjelasan dalam kotak *login* (*placeholder*), menampilkan data pada formulir pengajuan penanganan PBB, dan lain-lain. Oleh karena itu, diperlukan pengujian terlebih dahulu sebelum sistem diterima oleh pengguna. Berdasarkan penjelasan mengenai parameter *usability* di atas dapat dilakukan pengujian kompatibilitas *browser* yang berfungsi menguji antar muka sistem yang sesuai dengan perancangan antar muka pengguna dan menguji antar muka dalam menampilkan antar muka secara lengkap.

4.2 Analisis Komponen

Proses analisis komponen diawali dengan mencari semua komponen yang berhubungan dengan pokok bahasan skripsi. Langkah selanjutnya adalah memilih beberapa komponen yang sesuai dengan kebutuhan yang diperlukan dalam sistem. Komponen yang dapat memenuhi kebutuhan akan langsung dipakai dalam sistem, sedangkan komponen yang tidak sesuai akan mengalami tahap modifikasi kebutuhan. Tetapi dalam skripsi ini tidak melakukan tahap modifikasi sistem karena fokus skripsi ini hanya pada penggunaan komponen-komponen yang sesuai dengan kebutuhan. Komponen-komponen yang dapat memenuhi kebutuhan digunakan untuk memenuhi kebutuhan dua sistem dalam skripsi ini, yaitu sistem ODP dan Sistem Virtual Bank.

Komponen-komponen yang digunakan dalam sistem ODP adalah *framework* PHP *CodeIgniter* 2.1.4 (EllisLab). *Framework* ini akan digunakan untuk membangun aplikasi berbasis *web*. Tujuan penggunaan *framework* ini adalah untuk memudahkan pembuatan aplikasi berbasis web dengan komponen-kompenen yang tersedia di dalamnya. Kelebihan *framework* ini adalah

dokumentasi penggunaan yang lengkap dan mudah dipahami, komunitas pengguna yang besar, dan menggunakan pola MVC untuk memisahkan antara *bussiness process*, data yang digunakan, dan antar-muka.

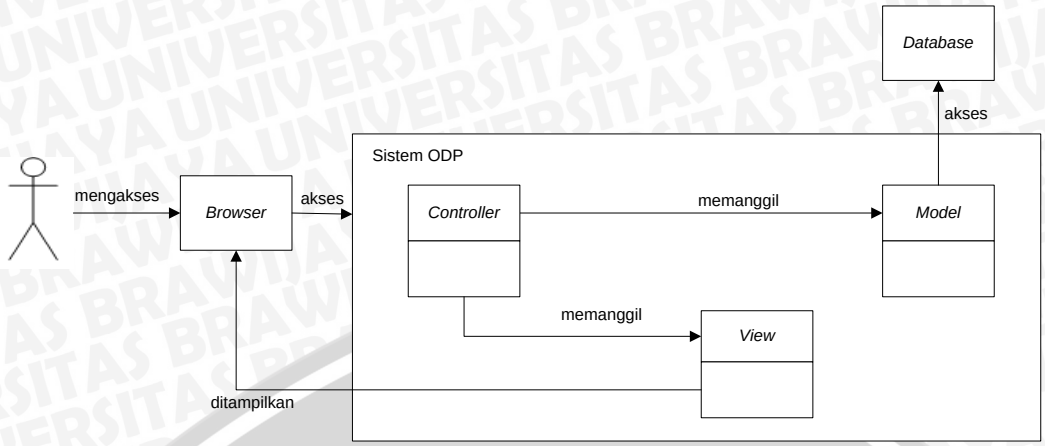
Sedangkan komponen-komponen lain yang digunakan adalah arsitektur SOA yang digunakan dalam Sistem Virtual Bank. Sistem Virtual Bank berfungsi untuk menjembatani sistem dalam mengakses data. SOA ini menggunakan *web service* SOAP karena SOAP ini memiliki kelebihan dalam keamanan data. Artinya, apabila sistem mengakses *service* maka *service* tidak langsung menampilkan data secara detail tetapi *service* hanya menampilkan struktur datanya saja. Hal ini berfungsi untuk menghindari penyalahgunaan data oleh pihak yang tidak bertanggungjawab. SOAP ini menggunakan *library* NuSOAP yang men-*generate Web Service Definition Language* (WSDL). WSDL digunakan untuk menampilkan struktur data yang disimpan dalam *service*. Jadi apabila mengakses layanan *service* maka yang dipanggil adalah WSDL-nya.

4.3 Perancangan Dengan Komponen

Perancangan dengan komponen bertujuan untuk menggunakan komponen-komponen yang telah dianalisis dan dipilih untuk digunakan dalam sistem. Dalam skripsi ini salah satu perancangan dengan komponen adalah mencakup penggunaan *framework* yang sudah ada dalam pembuatan sistem. Dengan menggunakan *framework* yang sudah ada, penulis hanya menggunakan fasilitas-fasilitas yang tersedia sehingga penulis tidak perlu lagi membuat sistem dari awal. Perancangan dengan komponen terdiri dari empat tahap, yaitu arsitektur sistem, pemodelan kelas, pemodelan data, dan perancangan antar muka pengguna.

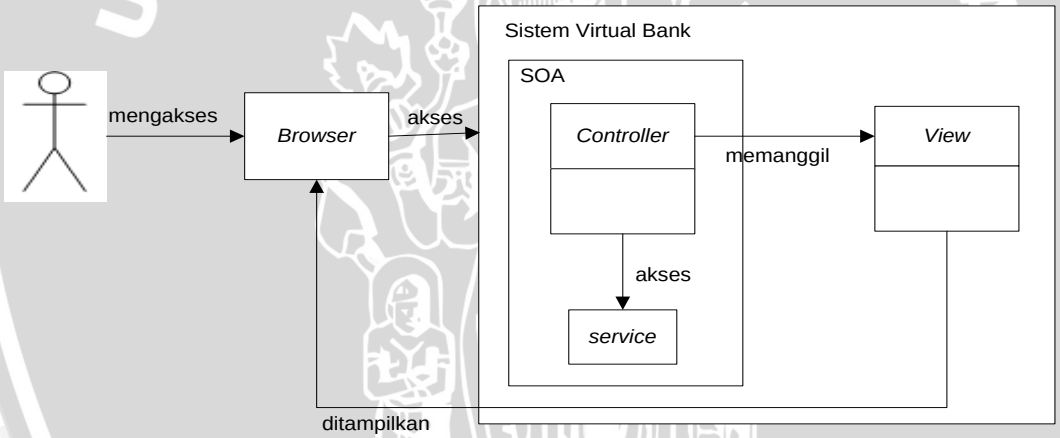
4.3.1 Perancangan Arsitektur Sistem

Perancangan arsitektur sistem merupakan gambaran dari sistem secara keseluruhan mulai dari awal pengguna mengakses sistem kemudian diproses sampai mengembalikan hasil pada pengguna. Perancangan arsitektur sistem ODP dapat dilihat pada Gambar 4.4.



Gambar 4.4 Perancangan Arsitektur Sistem ODP

Penjelasan dari Gambar 4.4 adalah pengguna mengakses sistem melalui *browser*. Kemudian sistem mengakses *controller* untuk dapat manampilkan *view*. *Controller* mengakses *model* untuk mengakses data yang diperoleh dari *database*. Setelah itu sistem dapat mengembalikan hasil berupa antar muka sistem.



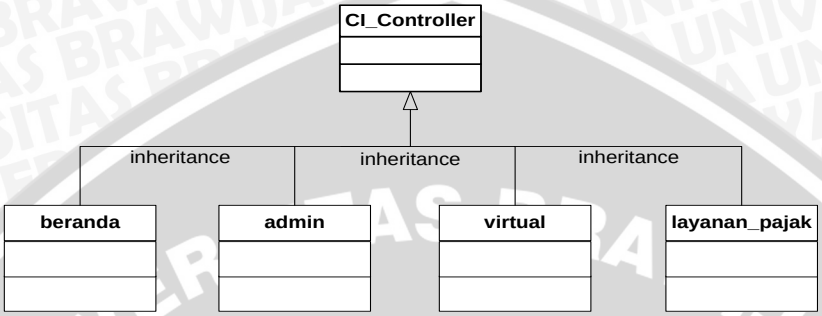
Gambar 4.5 Perancangan Arsitektur Sistem Virtual Bank

Gambar 4.5 merupakan perancangan arsitektur sistem virtual Bank. Penjelasan dari arsitektur ini adalah pengguna mengakses sistem melalui *browser*. Kemudian sistem mengakses *controller* untuk dapat manampilkan *view*. *Controller* mengakses data melalui *service* jadi tidak melalui *database*. Setelah itu sistem dapat mengembalikan hasil berupa antar muka sistem.

Sistem virtual Bank ini menggunakan arsitektur SOA karena untuk menjembatani sistem ODP dan sistem virtual Bank yang saling berjalan dalam mengolah data PBB. Dalam SOA menggunakan *web service* SOAP karena untuk memberikan batasan-batasan dalam pengolahan data agar sistem virtual Bank hanya terfokus dengan data-data yang diperlukan dalam validasi pembayaran saja.

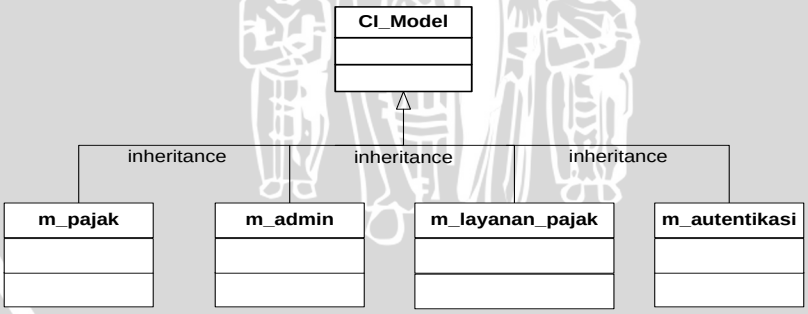
4.3.2 Perancangan Kelas

Diagram kelas memberikan gambaran tentang sistem dan relasi-relasi yang terdapat di dalamnya. Kelas-kelas yang telah teridentifikasi dapat memiliki hubungan antar kelas. Kelas-kelas juga dapat memiliki pewarisan dan ketergantungan terhadap komponen-komponen yang telah ditentukan.



Gambar 4.6 Diagram Kelas *Controller* Sistem ODP

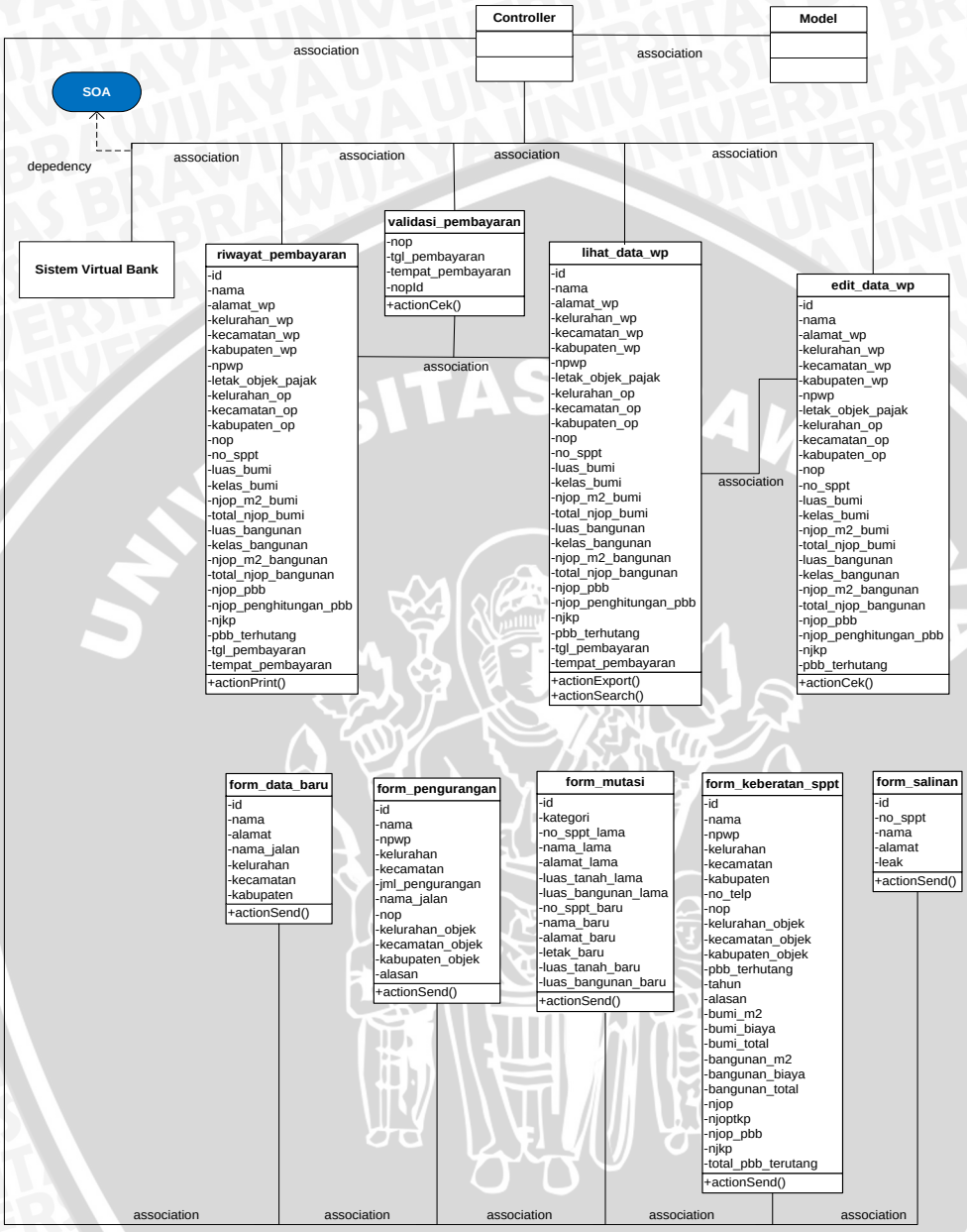
Gambar 4.6 merupakan diagram kelas *controller* sistem ODP dimana kelas CI_Controller mewariskan sifatnya ke anak kelas. Anak kelas terdiri dari kelas beranda, kelas admin, kelas virtual, dan kelas layanan_pajak. Kelas beranda berisi semua proses fungsi untuk wajib pajak. Kelas admin berisi semua proses fungsi untuk DPPKAD dan Kelurahan/Kecamatan. Kelas virtual berisi semua proses fungsi untuk sistem virtual Bank. Sedangkan kelas layanan_pajak berisi semua fungsi *service* yang digunakan oleh sistem virtual Bank.



Gambar 4.7 Diagram Kelas *Model* Sistem ODP

Gambar 4.7 merupakan diagram kelas *model* sistem ODP dimana kelas CI_Model mewariskan sifatnya ke anak kelas. Anak kelas terdiri dari kelas m_pajak, kelas m_admin, kelas m_layanan_pajak, dan kelas m_autentikasi. Kelas m_pajak berisi semua proses pengolahan data untuk wajib pajak. Kelas m_admin berisi semua proses pengolahan data untuk DPPKAD dan Kelurahan/Kecamatan. Kelas m_layanan_pajak berisi semua proses pengolahan data untuk sistem virtual

Bank. Sedangkan kelas m_autentikasi berisi semua proses pengolahan data untuk autentikasi pengguna.



Gambar 4.8 Diagram Kelas Sistem ODP

Gambar 4.8 merupakan diagram kelas keseluruhan untuk sistem ODP. Semua kelas *view* dan kelas *model* pada sistem berhubungan dengan kelas *Controller* pada *framework* CI. *Controller* sistem virtual Bank memiliki ketergantungan pada komponen SOA untuk mengakses data wajib pajak dalam melakukan validasi pembayaran PBB. Berikut penjelasan dari beberapa kelas

pada gambar 4.8, seperti kelas *controller*, kelas *model*, kelas riwayat_pembayaran, dan kelas lihat_data_wp.

1. Kelas *Controller*

Tabel 4.15 Penjelasan Kelas *Controller*

Nama Kelas	*Controller*
Deskripsi	Kelas dari komponen *framework* CI yang berfungsi sebagai penyedia banyak fungsionalitas dari sistem.

2. Kelas Model

Tabel 4.16 Penjelasan Kelas Model

Nama Kelas	Model
Deskripsi	Kelas dari komponen *framework* CI yang berfungsi sebagai penyedia banyak proses yang menjalankan aksi pada *controller* yang ditambahkan pada sistem

3. Kelas riwayat_pembayaran

Tabel 4.17 Penjelasan Kelas Riwayat Pembayaran

Nama Kelas	riwayat_pembayaran
Deskripsi	Kelas ini berfungsi untuk menampilkan riwayat pembayaran yang telah dilakukan oleh wajib pajak.

4. Kelas lihat_data_wp

Tabel 4.18 Penjelasan Kelas Lihat Data Wajib Pajak

Nama Kelas	lihat_data_wp
Deskripsi	Kelas ini berfungsi untuk menampilkan data wajib pajak dalam melakukan pembayaran pajak dan pengajuan penanganan PBB.

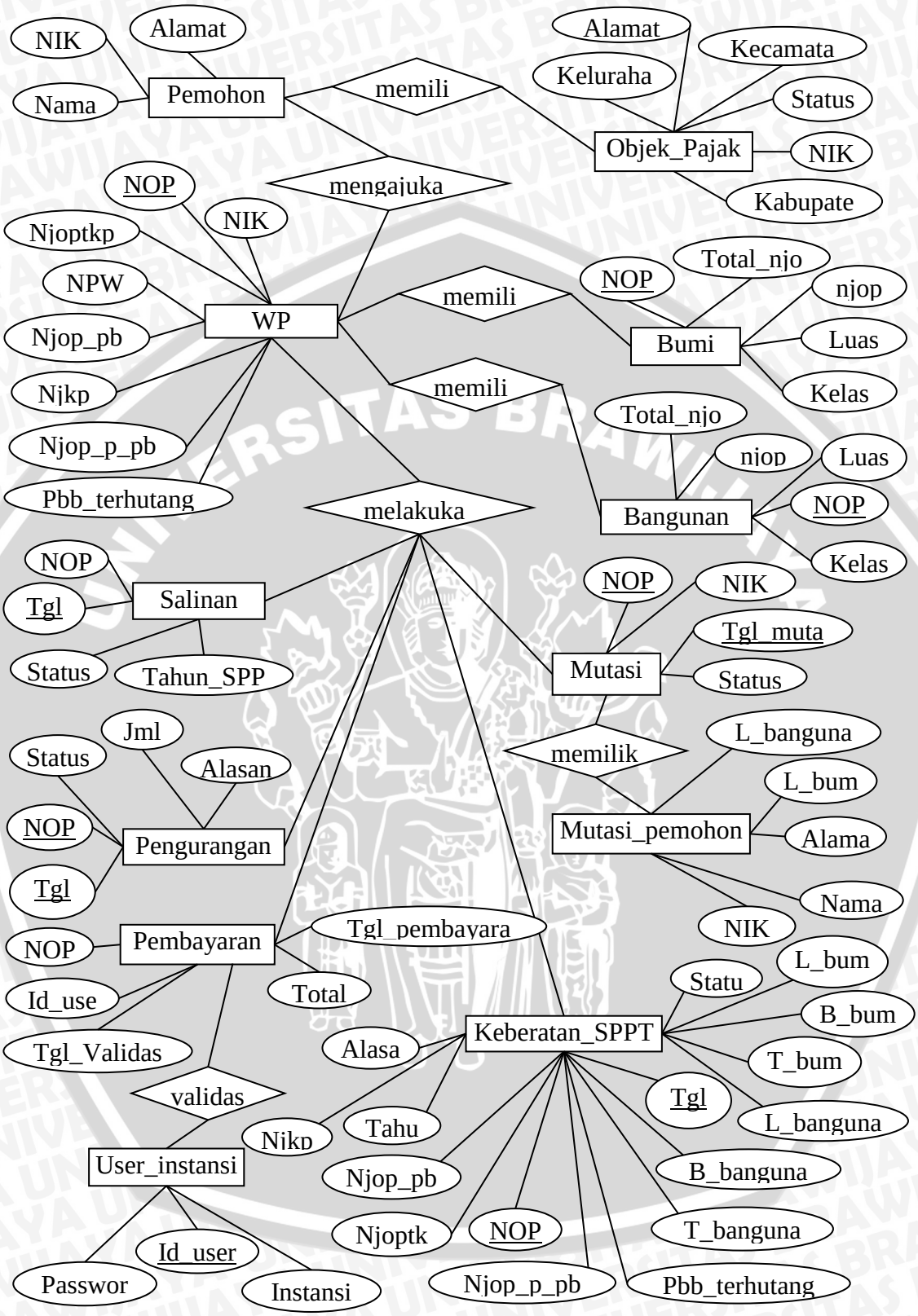
5. Kelas untuk Sistem Virtual Bank

Tabel 4.19 Penjelasan Kelas Untuk Sistem Virtual Bank

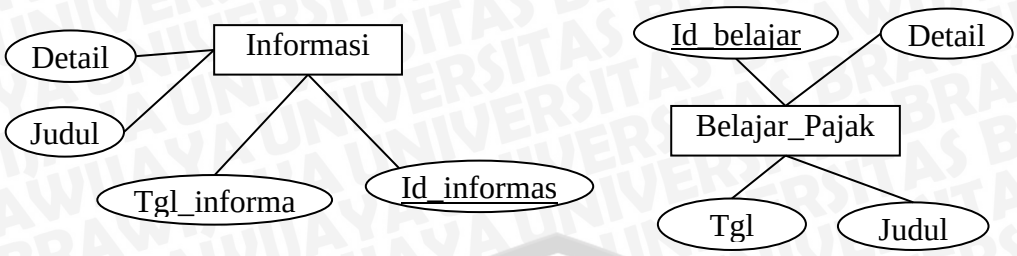
Nama Kelas	virtual
Deskripsi	Kelas ini berfungsi untuk menangkap dan menyimpan data wajib pajak yang diambil dari *database* dalam *service* yang nantinya akan dipanggil dan ditampilkan dalam sistem virtual bank. *Service* yang digunakan adalah *web service* SOA.

4.3.3 Pemodelan Data

Pemodelan data merupakan perancangan manajemen data yang akan digunakan sistem. Manajemen data termasuk basis data, yang mengandung data relevan dari berbagai situasi dan diatur oleh perangkat lunak yang disebut *Database Management System* (DBMS). Pemodelan data pada sistem ini digambarkan melalui *entity-relationship diagram.* Gambar 4.9 menunjukkan diagram *entity-relationship* pengolahan data pajak dan validasi pembayaran PBB.



Gambar 4.9 Diagram *Entity Relationship* Pengolahan Data Pajak dan Validasi Pembayaran PBB

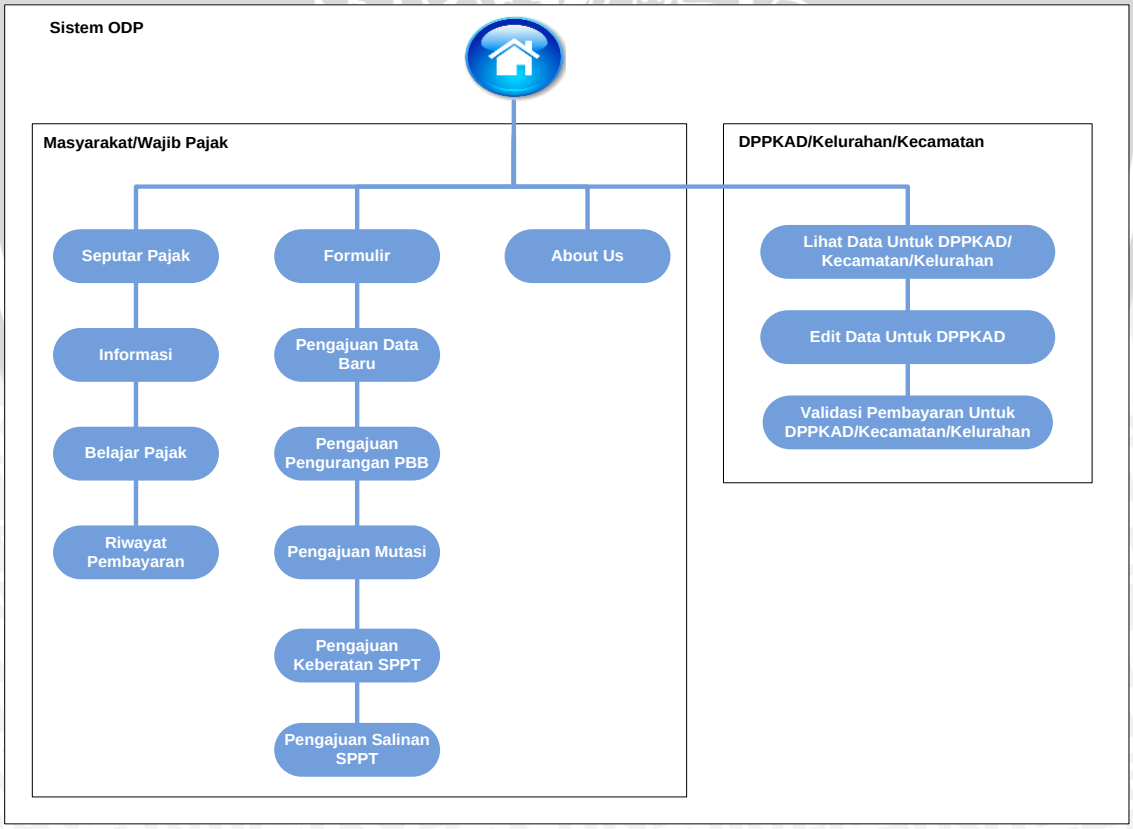


Gambar 4.10 Diagram *Entity Relationship* Informasi dan Belajar_Pajak

Gambar 4.10 menunjukkan diagram *entity-relationship* dari setiap entitas dengan dilengkapi beberapa atribut yang dimiliki oleh masing-masing entitas.

4.3.4 Perancangan Antar Muka Pengguna

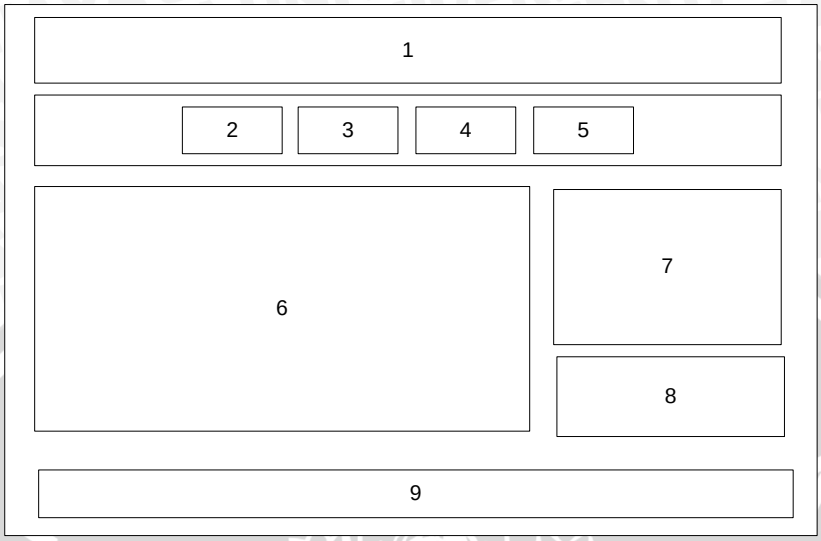
Tujuan dari perancangan antar muka pengguna adalah untuk merancang antar muka sistem yang akan dibuat agar dapat mudah diterima oleh pengguna. Perancangan antar muka pengguna mewakili keadaan sebenarnya dari sistem yang akan dibangun. *Site map* sistem ODP untuk masyarakat/wajib pajak dan DPPKAD/Kelurahan/Kecamatan ditunjukkan pada Gambar 4.11.



Gambar 4.11 *Site Map* Sistem ODP

4.3.4.1 Perancangan Halaman Utama

Gambar 4.12 merupakan antar muka halaman utama dari sistem ODP berbasis *web*.



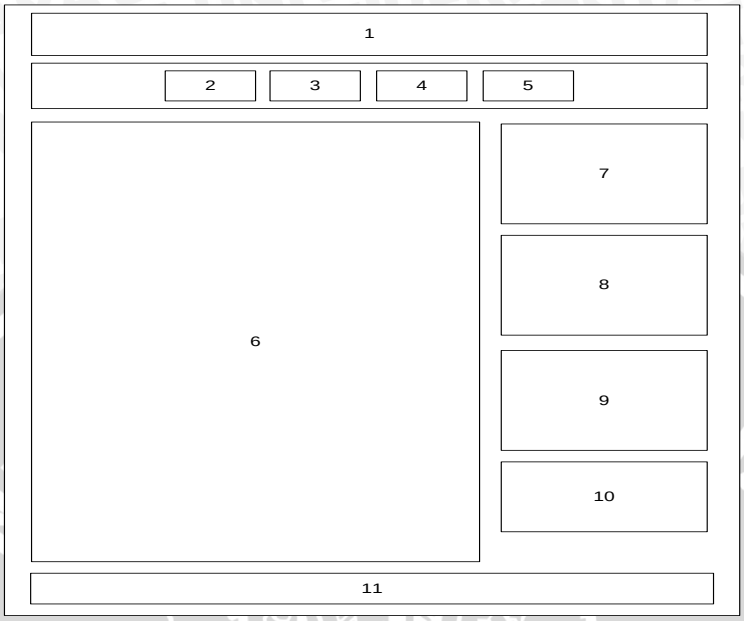
Gambar 4.12 Antar Muka Halaman Utama

Halaman utama menampilkan sekilas mengenai gambaran tentang informasi-informasi mengenai PBB. Gambar 4.12 memiliki keterangan sebagai berikut :

1. *Header* berisi logo dan judul dari sistem.
2. Menu *home* yang merupakan halaman utama yang memuat sekilas mengenai gambaran tentang informasi-informasi mengenai PBB.
3. Menu seputar pajak digunakan untuk memberikan informasi tentang PBB.
4. Menu formulir digunakan untuk memberikan kemudahan dalam membuat pengajuan penanganan PBB dengan mengisi formulir penangan PBB yang tersedia pada sistem.
5. Menu tentang digunakan untuk memberikan kontak dari DPPKAD.
6. *Slide show* untuk menampilkan gambar-gambar tentang kejadian yang terjadi seputar PBB.
7. Kotak *Login* digunakan untuk autentikasi wajib pajak atau DPPKAD/Kelurahan/Kecamatan.
8. Tabel riwayat pembayaran apabila wajib pajak telah melakukan autentikasi.
9. *Footer* berisi hak cipta dan tahun dari sistem ODP.

4.3.4.2 Perancangan Halaman Seputar Pajak

Halaman seputar pajak menampilkan beberapa informasi tentang PBB mulai dari pengertian PBB, pengumuman seputar PBB dan belajar pajak



Gambar 4.13 Antar Muka Halaman Seputar Pajak

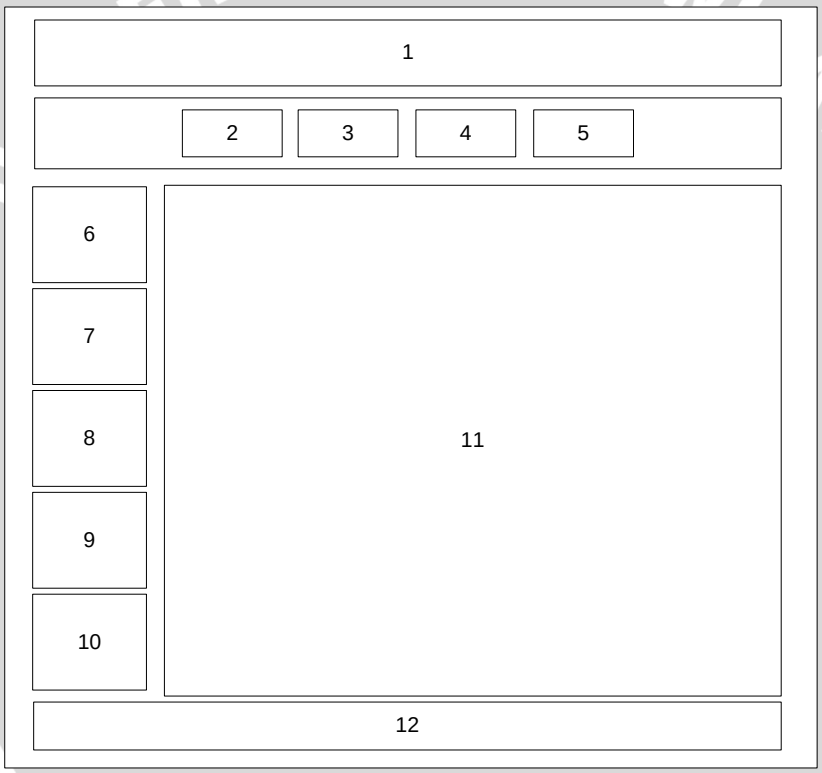
Gambar 4.13 merupakan rancangan antar muka halaman seputar pajak dalam sistem ODP. Keterangan-keterangan Gambar 4.13 adalah sebagai berikut:

1. *Header* berisi logo dan judul dari sistem.
2. Menu *home* yang merupakan halaman utama yang memuat sekilas mengenai gambaran tentang informasi-informasi mengenai PBB.
3. Menu seputar pajak digunakan untuk memberikan informasi tentang PBB.
4. Menu formulir digunakan untuk memberikan kemudahan dalam membuat pengajuan penanganan PBB dengan mengisi formulir penangan PBB yang tersedia pada sistem.
5. Menu tentang digunakan untuk memberikan kontak dari DPPKAD.
6. Penjelasan semua informasi tentang PBB mulai dari pengertian PBB, penjelasan tentang subjek/objek pajak, cara pendaftaran PBB, dan tarif PBB.
7. Kotak *Login* digunakan untuk autentikasi wajib pajak atau DPPKAD/Kelurahan/Kecamatan.
8. Kumpulan atau cuplikan pengumuman tentang PBB.

9. Belajar pajak yang memuat beberapa informasi tentang aturan-aturan PBB.
10. Tabel riwayat pembayaran apabila wajib pajak telah melakukan autentikasi.
11. *Footer* berisi hak cipta dan tahun dari sistem ODP.

4.3.4.3 Perancangan Halaman Formulir

Halaman formulir menampilkan formulir pengajuan penanganan PBB, seperti formulir pengajuan data baru, formulir pengajuan pengurangan PBB, formulir pengajuan mutasi, formulir pengajuan keberatan atas pengenaan PBB, dan formulir pengajuan salinan SPPT.



Gambar 4.14 Antar Muka Halaman Formulir

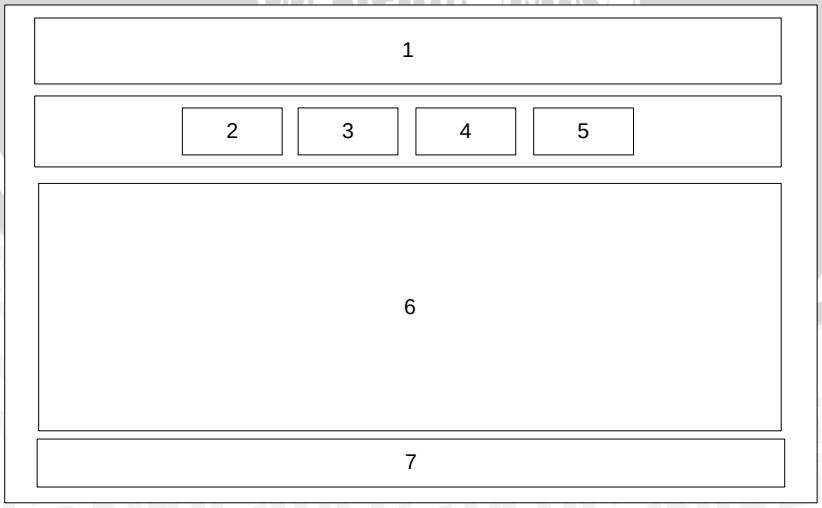
Gambar 4.14 merupakan rancangan antar muka halaman formulir dalam sistem ODP. Keterangan-keterangan Gambar 4.14 adalah sebagai berikut:

1. *Header* berisi logo dan judul dari sistem.
2. Menu *home* yang merupakan halaman utama yang memuat sekilas mengenai gambaran tentang informasi-informasi mengenai PBB.
3. Menu seputar pajak digunakan untuk memberikan informasi tentang PBB.

4. Menu formulir digunakan untuk memberikan kemudahan dalam membuat pengajuan penanganan PBB dengan mengisi formulir penangan PBB yang tersedia pada sistem.
5. Menu tentang digunakan untuk memberikan kontak dari DPPKAD.
6. Menu pengajuan data objek pajak baru digunakan apabila masyarakat/wajib pajak yang memiliki objek pajak baru.
7. Menu pengajuan pengurangan PBB digunakan apabila wajib pajak ingin mengajukan pengurangan biaya PBB karena beberapa alasan.
8. Menu pengajuan mutasi PBB digunakan apabila terjadi perubahan subjek/objek pajak.
9. Menu pengajuan keberatan atas pengenaan PBB digunakan apabila wajib pajak keberatan terhadap PBB yang dikenakan.
10. Menu pengajuan salinan SPPT digunakan apabila wajib pajak ingin meminta salinan SPPT apabila terjadi sesuatu hal.
11. Formulir dari penanganan PBB
12. *Footer* berisi hak cipta dan tahun dari sistem ODP.

4.3.4.4 Perancangan Halaman Tentang

Halaman tentang menampilkan beberapa informasi singkat tentang kontak yang bisa dihubungi atau alamat yang bisa dituju apabila terdapat pertanyaan atau hal yang perlu diurus mengenai PBB.



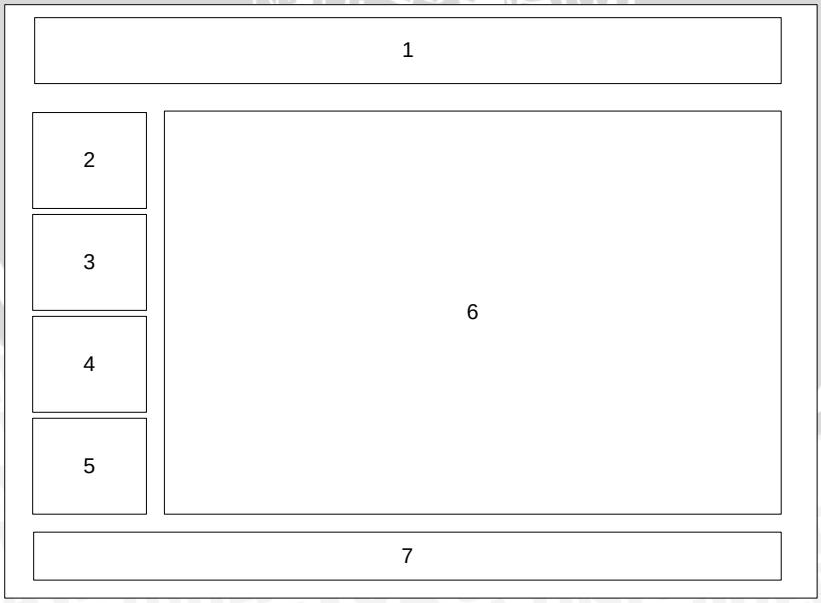
Gambar 4.15 Antar Muka Halaman Tentang

Gambar 4.15 merupakan rancangan antar muka halaman tentang dalam sistem ODP. Keterangan-keterangan Gambar 4.15 adalah sebagai berikut:

1. *Header* berisi logo dan judul dari sistem.
2. Menu *home* yang merupakan halaman utama yang memuat sekilas mengenai gambaran tentang informasi-informasi mengenai PBB.
3. Menu seputar pajak digunakan untuk memberikan informasi tentang PBB.
4. Menu formulir digunakan untuk memberikan kemudahan dalam membuat pengajuan penanganan PBB dengan mengisi formulir penangan PBB yang tersedia pada sistem.
5. Menu tentang digunakan untuk memberikan kontak dari DPPKAD.
6. Kontak yang bisa dihubungi atau alamat yang bisa dituju.
7. *Footer* berisi hak cipta dan tahun dari sistem ODP.

4.3.4.5 Perancangan Halaman Olah Data Wajib Pajak Untuk DPPKAD/ Kelurahan/Kecamatan

Halaman olah data wajib pajak untuk DPPKAD/Kelurahan/Kecamatan digunakan untuk mengolah data wajib pajak yang meliputi tambah data wajib pajak dan ubah data wajib.



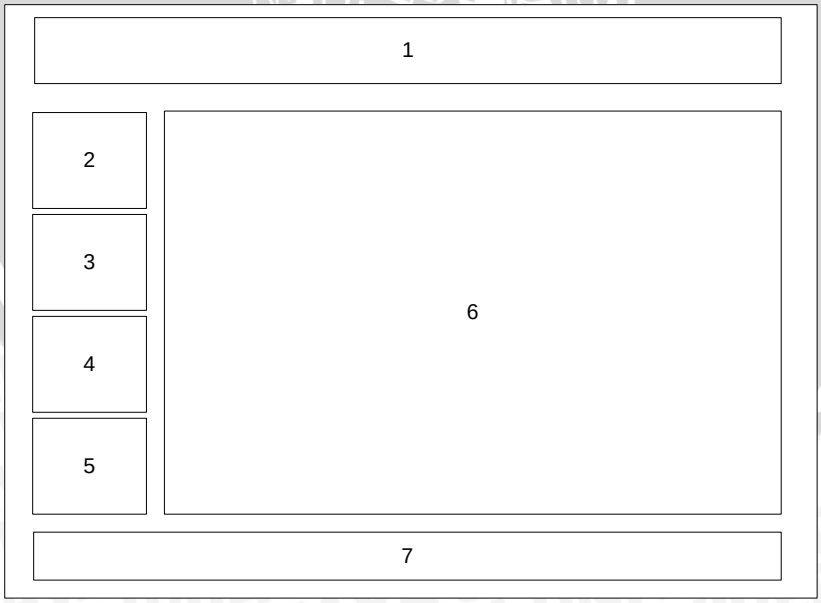
Gambar 4.16 Antar Muka Halaman Olah Data Wajib Pajak Untuk DPPKAD/Kelurahan/Kecamatan

Gambar 4.16 merupakan rancangan antar muka halaman olah data wajib pajak untuk DPPKAD/Kelurahan/Kecamatan dalam sistem ODP. Keterangan-keterangan Gambar 4.16 adalah sebagai berikut:

1. *Header* berisi logo dan judul dari sistem.
2. Menu tambah data wajib pajak baru
3. Menu lihat data wajib pajak untuk melihat data pembayaran PBB dan melihat data pemohon pengajuan penanganan PBB.
4. Menu ubah data wajib pajak digunakan untuk mengubah data wajib pajak atau objek pajak apabila terjadi perubahan.
5. Menu validasi pembayaran digunakan untuk memasukkan data wajib pajak yang telah melakukan pembayaran.
6. Isi dari ubah atau tambah data wajib pajak.
7. *Footer* berisi hak cipta dan tahun dari sistem ODP.

4.3.4.6 Perancangan Halaman Validasi Pembayaran Untuk DPPKAD/ Kelurahan/Kecamatan

Halaman validasi pembayaran untuk DPPKAD/Kelurahan/Kecamatan digunakan untuk mem-validasi data wajib pajak yang telah melakukan pembayaran.



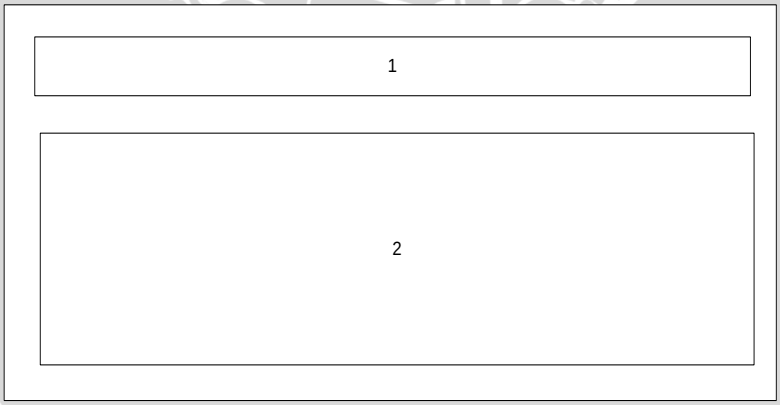
Gambar 4.17 Antar Muka Halaman Validasi Pembayaran Untuk DPPKAD/Kelurahan/Kecamatan

Gambar 4.17 merupakan rancangan antar muka halaman validasi pembayaran untuk DPPKAD/Kelurahan/Kecamatan dalam sistem ODP. Keterangan-keterangan Gambar 4.17 adalah sebagai berikut:

1. *Header* berisi logo dan judul dari sistem.
2. Menu tambah data wajib pajak baru
3. Menu lihat data wajib pajak berfungsi untuk melihat data pembayaran PBB dan melihat data pemohon pengajuan penanganan PBB.
4. Menu ubah data wajib pajak digunakan untuk mengubah data wajib pajak atau objek pajak apabila terjadi perubahan.
5. Menu validasi pembayaran digunakan untuk memasukkan data wajib pajak yang telah melakukan pembayaran.
6. Formulir untuk validasi.
7. *Footer* berisi hak cipta dan tahun dari sistem *ODP*.

4.3.4.7 Perancangan Halaman Sistem Virtual Bank

Halaman sistem virtual bank digunakan untuk melakukan proses mem-validasi data wajib pajak yang telah melakukan pembayaran. Sistem virtual Bank hanya digunakan oleh pihak Bank saja.



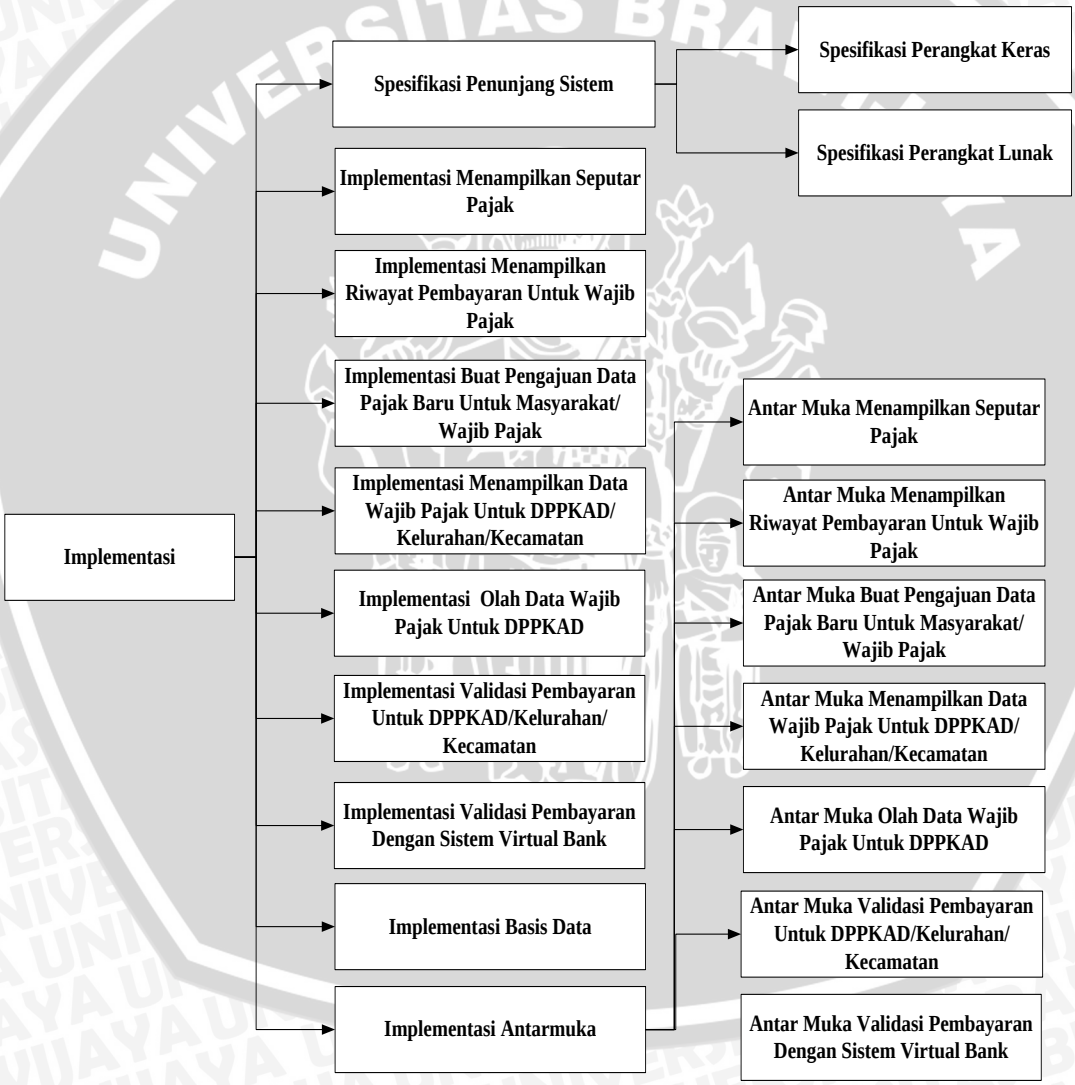
Gambar 4.18 Antar Muka Halaman Sistem Virtual Bank

Gambar 4.18 merupakan rancangan halaman sistem virtual bank yang berfungsi untuk validasi pembayaran PBB. Keterangan-keterangan Gambar 4.18 adalah sebagai berikut:

1. *Header* berisi judul dari sistem.
2. Formulir untuk proses memasukkan data wajib pajak yang telah melakukan pembayaran.

BAB V
IMPLEMENTASI

Bab ini membahas mengenai implementasi dan pengujian dari sistem yang telah dibuat berdasarkan hasil yang telah didapatkan dari pemodelan persyaratan dan proses perancangan sistem yang dibuat. Pembahasan implementasi terdiri atas penjelasan tentang spesifikasi sistem, implementasi proses pada sistem dan implementasi antarmuka. Tahap-tahap implementasi yang dikerjakan digambarkan pada Gambar 5.1 berikut ini.



Gambar 5.1 Diagram Implementasi

Sistem ODP merupakan sebuah sistem berbasis *web* yang digunakan dalam pengolahan data pembayaran PBB. Sistem ODP dibuat menggunakan perangkat keras dan perangkat lunak yang ditentukan oleh penulis. Sistem ODP mempunyai beberapa fitur yang dibuat sesuai dengan kebutuhan pengguna. Pada penulisan skripsi ini hanya dicantumkan beberapa proses dari fitur saja sehingga tidak semua fitur akan dicantumkan. Fitur yang dicantumkan antara lain adalah lihat seputar pajak, lihat riwayat pembayaran untuk wajib pajak, buat pengajuan data pajak baru untuk masyarakat/wajib pajak, lihat data wajib pajak untuk DPPKAD/Kelurahan/Kecamatan, olah data wajib pajak untuk DPPKAD, validasi pembayaran untuk DPPKAD/Kelurahan/Kecamatan, dan validasi pembayaran dengan sistem virtual bank. Implementasi fitur-fitur ini akan direpresentasikan dalam bentuk kode dengan bahasa pemrograman PHP pada *framework CodeIgniter*. Tahap-tahap implementasi pada Gambar 5.1 dijelaskan pada sub-sub bab berikut:

5.1 Spesifikasi Sistem

Hasil pemodelan persyaratan yang telah diuraikan pada perancangan sebelumnya menjadi acuan dalam melakukan implementasi pembuatan sistem sesuai dengan kebutuhan. Spesifikasi sistem diimplementasikan pada spesifikasi perangkat keras dan perangkat lunak yang digunakan untuk membangun aplikasi.

5.1.1 Spesifikasi Perangkat Keras

Spesifikasi perangkat keras meliputi spesifikasi prosesor, memori, kapasitas HDD, dan kartu grafis dijelaskan pada Tabel 5.1.

Tabel 5.1 Spesifikasi Perangkat Keras

Nama Komponen	Spesifikasi
Prosesor	2.67 GHz Intel(R) Core(TM) i5 CPU
Memori (RAM)	4.00 GB (3.86 GB *usable*)
Kapasitas HDD	640 GB
Kartu Grafis	RADEON HD 6470 1 GB

5.1.2 Spesifikasi Perangkat Lunak

Spesifikasi perangkat lunak yang dipakai dalam proses pengembangan dan pengujian sistem ODP dijelaskan pada tabel 5.2.

Tabel 5.2 Spesifikasi Perangkat Lunak

Nama Komponen	Spesifikasi
Sistem Operasi	Windows 7 Ultimate, 64 Bit
Browser	1. Mozilla Firefox Version 16.0.2 2. Google Chrome Version 33.0.1750.154 m 3. Internet Explorer Version 8.0.7601.17514
Tool	1. Adobe Dreamweaver CS5 version 11.0. *Tools* ini digunakan untuk menulis kode PHP untuk membuat aplikasi berbasis *web*. 4. XAMPP version 1.7.3. *Tools* ini digunakan untuk membuat *local server,* sehingga aplikasi berbasis web dapat dijalankan.

5.2 Implementasi Lihat Seputar Pajak

Halaman seputar pajak menyediakan beberapa informasi yang terdiri dari seputar pajak, pengumuman tentang pajak, dan belajar pajak. Seputar pajak berisi penjelasan-penjelasan mulai dari pengertian PBB, objek/subjek pajak, cara pendaftaran PBB, dan tarif PBB. Pengumuman berisi tentang segala informasi yang terjadi berkaitan dengan PBB. Sedangkan belajar pajak berisi tentang aturan-aturan yang mengikat mengenai PBB. Gambar 5.2 merupakan cuplikan kode yang ada pada *controller*.

```
1.  $data['sidebar'] = 1;
2.  $data['belajar_pajak']          =          $this->m_pajak-
3.  >getAllBelajarPajak();
    $data['informasi']= $this->m_pajak->getAllInformasi('4');
```

Gambar 5.2 Implementasi Lihat Seputar Pajak

Penjelasan implementasi seputar pajak dari Gambar 5.2 yaitu:

1. Baris 1 berfungsi memberikan status nilai 1 untuk menampilkan *sidebar* pengumuman dan informasi belajar pajak.
2. Baris 2 berfungsi menampilkan data tentang belajar pajak.
3. Baris 3 berfungsi menampilkan data tentang pengumuman terbaru seputar PBB.

5.3 Implementasi Lihat Riwayat Pembayaran Untuk Wajib Pajak

Riwayat pembayaran merupakan fungsi yang disediakan untuk setiap wajib pajak yang telah melakukan autentikasi terlebih dahulu ke sistem ODP. Gambar 5.3 merupakan cuplikan kode dari riwayat pembayaran.

```
1.  $data['riwayat'] = $this->m_pajak->getHistoryPembayaran(5);
```

Gambar 5.3 Implementasi Lihat Riwayat Pembayaran Untuk Wajib Pajak

Penjelasan implementasi riwayat pembayaran untuk wajib pajak dari Gambar 5.3 adalah baris 1 berfungsi memanggil model m_pajak untuk mengambil data pembayaran yang telah dilakukan oleh wajib pajak yang melakukan autentikasi.

5.4 Implementasi Buat Pengajuan Data Pajak Baru Untuk Masyarakat/ Wajib Pajak

Buat pengajuan data pajak baru berfungsi untuk membantu masyarakat/wajib pajak dalam membuat pengajuan objek pajak baru. Gambar 5.4 merupakan cuplikan kode pengajuan data pajak baru.

```
 1.  $data['visible'] = 1;
 2.  $data['judul'] = 'Pengajuan Data Baru';
 3.  $data['sidebar'] = 0;
 4.  $data['errors'] = 0;
 5.  $isclick = $this->input->post('isclick');
 6.  if($isclick==1){
 7.    $nik = $this->input->post('nik');
 8.    $nama = $this->input->post('nama');
 9.    $alamat = $this->input->post('alamat');
10.    $n_jalan = $this->input->post('n_jalan');
11.    $kelurahan = $this->input->post('kelurahan');
12.    $kecamatan = $this->input->post('kecamatan');
13.    $kabupaten = $this->input->post('kabupaten');
14.  if(!empty($nik)&&!empty($nama)&&!empty($alamat)&&!empty($n_
15.  jalan)&&!empty($kelurahan)&&!empty($kecamatan)&&!empty($kab
16.  upaten)){
17.    $check = $this->m_pajak->findPemohon($nik);
18.  if($check==1){
19.    $this->m_pajak->insert_data_baru(1);
20.    $data['errors'] = 'Data berhasil dimasukkan!';
21.  }else if($check==-1){
22.    $this->m_pajak->insert_data_baru(-1);
23.    $data['errors'] = 'Data berhasil dimasukkan!';
24.  }}else{
25.    $data['errors'] ='Data yang Anda masukkan tidak lengkap';
26.  }
```

Gambar 5.4 Implementasi Pengajuan Data Pajak Baru

Penjelasan implementasi pengajuan data pajak baru untuk masyarakat dari Gambar 5.4 yaitu:

1. Baris 1-5 berfungsi untuk menyimpan data pada variabel.
2. Baris 6-13 berfungsi untuk menangkap data pemohon setelah mengisi formulir.
3. Baris 14-16 berfungsi untuk memberikan kondisi untuk pengecekan berdasarkan Nomor Induk Kependudukan (NIK) jika NIK sudah terdaftar maka satu NIK mempunyai daftar objek pajak lebih dari satu.
4. Baris 17-23 berfungsi sebagai kondisi untuk memanggil proses model untuk memasukkan data pengajuan.
5. Baris 24-26 berfungsi sebagai kondisi lain jika kondisi awal tidak terpenuhi.

5.5 Implementasi Lihat Data Wajib Pajak Untuk DPPKAD/Kelurahan/ Kecamatan

Lihat data wajib pajak untuk DPPKAD/Kelurahan/Kecamatan digunakan untuk memantau atau mengecek data pembayaran PBB dan pengajuan penanganan PBB yang dilakukan oleh wajib pajak. Menu lihat data wajib pajak ini didasarkan pada beberapa kategori. Kategorinya adalah lihat seluruh data pembayaran PBB dan lihat data wajib pajak berdasarkan yang melakukan pengajuan penanganan PBB baik yang valid maupun tidak valid. Selain lihat data wajib pajak, Dalam menu ini pihak DPPKAD juga bisa meng-*export* data wajib pajak ke bentuk excel sebagai rekapan data tahunan. Implementasi cuplikan kode lihat data wajib pajak untuk DPPKAD/Kelurahan/Kecamatan dapat dilihat pada gambar 5.5.

```
1.  $data['kategori'] = -1;
2.  $data['errors'] = '';
3.  $kategori = $this->input->post('kategori');
4.  $nop = $this->input->post('nop_search');
5.  $status = $this->input->post('status');
6.  $data['wajib_pajak']=NULL;
7.  $data['pengurangan']=NULL;
8.  $data['mutasi']=NULL;
9.  $data['keberatan']=NULL;
10. $data['salinan']=NULL;
11. if($kategori == '1'){
12.       $data['kategori'] = 1;
13.       $data['nama'] = 'Data Wajib Pajak';
14.       if(!empty($nop)){
15.             $result = $this->m_admin->getWajibPajak($nop);
16.             if($result==NULL){
17.                   $data['wajib_pajak'] = '';
```

```
18.                     $data['errors'] = 'Data tidak ditemukan!';
19.               }else{
20.                     $data['wajib_pajak'] = $result;
21.               }
22.         }else{
23.               $data['wajib_pajak']      =      $this->m_admin-
24. >getAllWajibPajak();          }
25. }
```

Gambar 5.5 Implementasi Lihat Data Wajib Pajak Untuk DPPKAD/Kelurahan/Kecamatan

Penjelasan implementasi lihat data wajib pajak untuk DPPKAD/Kelurahan/Kecamatan dari Gambar 5.5 yaitu:

1. Baris 1-10 berfungsi untuk menyimpan data ke dalam variabel.
2. Baris 11-13 berfungsi untuk memberikan kondisi jika kategori yang dipilih mempunyai nilai 1 maka akan menampilkan data wajib pajak.
3. Baris 14-21 berfungsi memberikan kondisi jika *textbox* NOP diisi maka akan menampilkan data wajib pajak berdasarkan NOP yang dimasukkan.
4. Baris 22-25 berfungsi memberikan kondisi lain jika kondisi awal tidak terpenuhi.

5.6 Implementasi Olah Data Wajib Pajak Untuk DPPKAD

Olah data wajib pajak terdiri dari ubah data wajib pajak dan tambah data wajib pajak. Ubah data wajib pajak digunakan untuk mengubah data pajak jika terdapat perubahan dari subjek/objek pajak. Implementasi cuplikan kode ubah data wajib pajak untuk DPPKAD dapat dilihat pada gambar 5.6.

```
1.  $isclick = $this->input->post('isclick');
2.  $kategori = $this->input->post('kategori');
3.  if($isclick=='1'){
4.  if($kategori=='1'){
5.        $this->form_validation-
6.  >set_rules('nik_search','nik_search','required');
7.        if($this->form_validation->run()){
8.              $hasil = $this->m_admin->getPemohon($this->input-
9.  >post('nik_search'));
10.             if($hasil!=NULL){
11.                   $data['model'] = $hasil;
12.                   $data['visible'] = 'pemohon';
13.             }else{
14.                   $data['model'] = NULL;
15.                   $data['errors'] = 'Data yang Anda cari
16. tidak ditemukan!';
17.             }
18.       }else{
19.             $data['errors'] = 'Data yang dimasukkan tidak
20. lengkap!';
```

```
21.         }
    }
```

Gambar 5.6 Implementasi Ubah Data Wajib Pajak Untuk DPPKAD

Cuplikan kode pada Gambar 5.6 merupakan cuplikan kode ubah data dengan kategori pemohon. Penjelasan implementasi ubah data wajib pajak untuk DPPKAD dari Gambar 5.6 yaitu:

1. Baris 1-2 berfungsi untuk menangkap masukkan kategori yang diinginkan pengguna.
2. Baris 3-4 berfungsi sebagai kondisi jika pengguna memilih kategori 1, yaitu kategori pemohon.
3. Baris 5-12 berfungsi untuk mengecek apakah NOP yang dimasukkan pengguna untu diubah datanya tersedia. Pencarian data ini didasarkan pada NIK karena NIK sebagai *primary key*. Jika data ditemukan maka sistem akan menampilkan formulir data untuk diubah.
4. Baris 13-17 merupakan kondisi jika data yang dicari tidak ditemukan.
5. Baris 18-21 merupakan kondisi jika kotak *text* tidak lengkap atau ada yang tidak diisi.

Tambah data wajib pajak digunakan untuk menambahkan data wajib pajak baru jika terdapat pengajuan subjek/objek pajak baru. Implementasi cuplikan kode tambah data wajib pajak untuk DPPKAD dapat dilihat pada gambar 5.7.

```
1.  $data['nama'] = 'Tambah Data Wajib Pajak';
2.  $data['visible'] = -1;
3.  $data['errors'] = 0;
4.  $data['kategori'] = '';
5.  $data['wajib_pajak'] = NULL;
6.  $nik = $this->input->post('nik_search');
7.  $kategori = $this->input->post('kategori');
8.  $isclick = $this->input->post('isclick');
9.  if(!empty($nik)&&$kategori!=0){
10.       $result                =                $this->m_admin-
11. >searchWajibPajak($nik,$kategori);
12.       if($result!=NULL){
13.             $data['wajib_pajak'] = $result;
14.             $data['visible'] = 1;
15.             $data['kategori'] = $kategori;
16.       }else{
17.             $data['errors'] = -2;
18.       }
19.       }else if(!empty($isclick)){
20.             if(empty($nik)||empty($kategori)){
21.                   $data['errors'] = -1;
22.             }
          }
```

Gambar 5.7 Implementasi Tambah Data Wajib Pajak Untuk DPPKAD

Penjelasan implementasi tambah data wajib pajak untuk DPPKAD dari Gambar 5.7 yaitu:

1. Baris 1-8 berfungsi untuk menyimpan data ke dalam variabel.
2. Baris 9-10 berfungsi untuk memberikan kondisi jika *textbox* NIK dan kategori sudah diisi.
3. Baris 11-22 berfungsi memberikan kondisi berhasil maka akan menampilkan data wajib pajak.

5.7 Implementasi Validasi Pembayaran Untuk DPPKAD/Kelurahan/ Kecamatan

Validasi pembayaran digunakan untuk mencatat siapa yang telah membayar, tanggal berapa PBB dibayar, dan tempat pembayaran sebagai bukti wajib pajak telah mekakukan pembayaran PBB. Implementasi cuplikan kode validasi pembayaran untuk DPPKAD/ Kecamatan/Kelurahan dapat dilihat pada gambar 5.8.

```
1.  $data['errors'] = NULL;
2.  $isclick = $this->input->post('isclick');
3.  if(!empty($isclick)){
4.        $nop = $this->input->post('nop_validasi');
5.        $jumlah = $this->input->post('jumlah');
6.              if(!empty($nop)&&!empty($jumlah)){
7.                    $result          =          $this->m_admin-
8.  >findWajibPajak($nop);
9.              if($result==1){
10.                   $this->m_admin->insert_pembayaran();
11.                   $data['errors']     =     'Data     berhasil
12. dimasukkan!';
13.             }else{
14.                   $data['errors'] = 'Data yang Anda masukkan
15. tidak ditemukan!';}
16.             }else{
17.                   $data['errors'] = 'Data yang Anda masukkan
18. tidak lengkap!';
19.              }
20. }
```

Gambar 5.8 Implementasi Validasi Pembayaran Untuk DPPKAD/Kelurahan/Kecamatan

Penjelasan implementasi validasi pembayaran untuk DPPKAD/Kelurahan/Kecamatan dari Gambar 5.8 yaitu:

1. Baris 1 berfungsi untuk memberikan status nilai 1 jika validasi pembayaran berhasil dilakukan.

2. Baris 2 berfungsi untuk memberikan status nilai 1 jika *textbox* tahun, nop, tanggal, bulan, tempat, dan total pembayaran kosong karena validasi pembayaran dapat dilakukan dengan memasukkan data tersebut.
3. Baris 3-8 berfungsi untuk menangkap data tahun, nop, tanggal, bulan, tempat, dan total pembayaran yang dimasukkan.
4. Baris 9-12 berfungsi memberikan kondisi jika *textbox* tahun, nop, tanggal, bulan, tempat, dan total pembayaran kosong maka tidak bisa melakukan validasi pembayaran dan akan muncul kotak peringatan.
4. Baris 13-20 berfungsi memberikan kondisi lain jika *textbox* diisi maka validasi pembayaran dapat dilakukan.

5.8 Implementasi Validasi Pembayaran Dengan Sistem Virtual Bank

Validasi pembayaran dengan Sistem Virtual Bank digunakan untuk mencatat siapa yang telah membayar dan tanggal berapa PBB dibayar atau dilaporkan sebagai bukti wajib pajak telah mekakukan pmebayaran PBB. Implementasi cuplikan kode validasi pembayaran dengan Sistem Virtual Bank dapat dilihat pada gambar 5.9.

```
1.  $nop = $this->input->post('nop_wp');
2.  $tgl_pembayaran = $this->input->post('tgl_bayar');
3.  $tgl_validasi = $this->input->post('validasi');
4.  $client = new nusoap_client($this->url, true);
5.  $param=array('nop'=>$nop,'tgl_pembayaran'=>$tgl_pembayaran,'
6.  tgl_validasi'=>$tgl_validasi);
7.  $hasil = $client->call('setTanggunganWP',$param);
8.  $data['validasi'] = NULL;
9.  $data['errors'] =  NULL;
10. $data['visible'] = NULL;
11. if($hasil=='1'){
12.     $data['errors'] =  'Data berhasil dimasukkan!';
13. }else{
14.     $data['errors'] =  'Data tidak valid!';
15. }
16. $this->load->vars($data);
17. $this->load->view('virtual_bank/virtual_view');
```

Gambar 5.9 Implementasi Validasi Pembayaran Dengan Sistem Virtual Bank

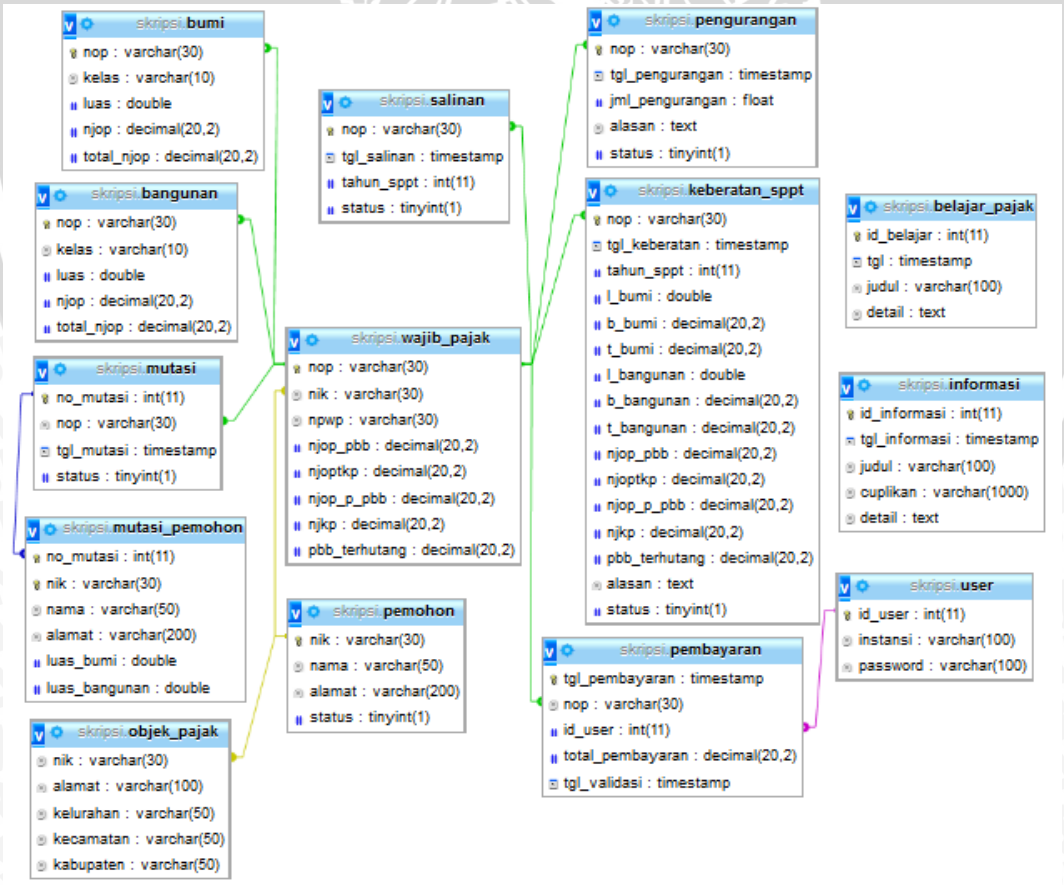
Penjelasan implementasi pembayaran dengan Sistem Virtual Bank dari Gambar 5.9 yaitu:

1. Baris 1-3 berfungsi untuk menangkap data yang telah dimasukkan.
2. Baris 4 berfungsi untuk membuat instansiasi variabel.

3. Baris 5-6 berfungsi untuk mencocokkan nama label pada *form* validasi dengan nama kolom pada *database*.
4. Baris 7 memanggil fungsi setTanggunganWP pada *service* yang berfungsi untuk menyimpan data pada *service*.
5. Baris 8-10 berfungsi untuk pengaturan *sidebar* dan beranda pada tampilan antarmuka.
6. Baris 11-15 berfungsi untuk memberikan pesan jika data sukses dimasukkan atau data tidak sesuai.
7. Baris 16-17 berfungsi untuk mengirim hasil data ke *view*.

5.9 Implementasi Basis Data

Implementasi penyimpanan data dilakukan dengan *Database Management System* (DBMS). Implementasi pada basis data ini dimodelkan dalam diagram konseptual *entity relationship.* Gambar 5.10 menggambarkan diagram konseptual *entitiy relationship* dari sistem ODP.



Gambar 5.10 Implementasi Basis Data Sistem ODP

5.10 Implementasi Antar Muka

Antar muka aplikasi ODP digunakan oleh pengguna untuk berinteraksi dengan sistem. Antar muka sistem ini dibagi menjadi tujuh, yaitu antar muka lihat seputar pajak, antar muka lihat riwayat pembayaran untuk wajib pajak, antar muka buat pengajuan data pajak baru untuk masyarakat/wajib pajak, antar muka lihat data wajib pajak untuk DPPKAD/Kelurahan/Kecamatan, antar muka olah data wajib pajak untuk DPPKAD, antar muka validasi pembayaran untuk DPPKAD/Kelurahan/Kecamatan, dan antar muka validasi pembayaran dengan sistem virtual bank.

5.10.1 Antar Muka Seputar Pajak

Halaman antar muka seputar pajak pada sistem ODP berbasis *web* menyediakan beberapa menu informasi. Informasi tersebut antara lain seputar pajak yang memuat penjelasan tentang PBB, pengumuman seputar PBB dan belajar PBB. Implementasi halaman antar muka seputar pajak dapat dilihat pada gambar 5.11.

Gambar 5.11 Antar Muka Seputar Pajak

5.10.2 Antar Muka Lihat Riwayat Pembayaran Untuk Wajib Pajak

Antar muka riwayat pembayaran untuk wajib pajak bertujuan untuk melihat atau memantau setiap pembayaran yang telah dilakukan. Implementasi

antar muka riwayat pembayaran untuk wajib pajak dapat dilihat pada gambar 5.12.

Gambar 5.12 Antar Muka Riwayat Pembayaran Untuk Wajib Pajak

Antar muka lihat pembayaran dapat dilihat pada setiap halaman setelah melakukan autentikasi. Antar muka di atas berfungsi menampilkan riwayat pembayaran yang telah dilakukan. Data pajak yang ditampilkan berupa tanggal pembayaran dan total PBB yang dibayar.

5.10.3 Antar Muka Buat Pengajuan Data Pajak Baru Untuk Masyarakat/ Wajib Pajak

Antar muka buat pengajuan data pajak baru digunakan untuk membuat pengajuan data pajak baru apabila masyarakat/wajib pajak mempunyai pajak baru. Implementasi halaman antar muka buat pengajuan data pajak baru untuk masyarakat/wajib pajak dapat dilihat pada gambar 5.13.

Gambar 5.13 Antar Muka Buat Pengajuan Data Pajak Baru Untuk Masyarakat/Wajib Pajak

5.10.4 Antar Muka Lihat Data Wajib Pajak Untuk DPPKAD/Kelurahan/ Kecamatan

Antar muka lihat data wajib pajak digunakan untuk menampilkan data pembayaran wajib pajak dan data pemohon pengajuan penaganan PBB. Implementasi halaman antar muka lihat data wajib pajak untuk DPPKAD/Kelurahan/Kecamatan dapat dilihat pada gambar 5.14.

Gambar 5.14 Antar Muka Lihat Data Wajib Pajak Untuk DPPKAD/Kelurahan/Kecamatan

5.10.5 Antar Muka Olah Data Wajib Pajak Untuk DPPKAD

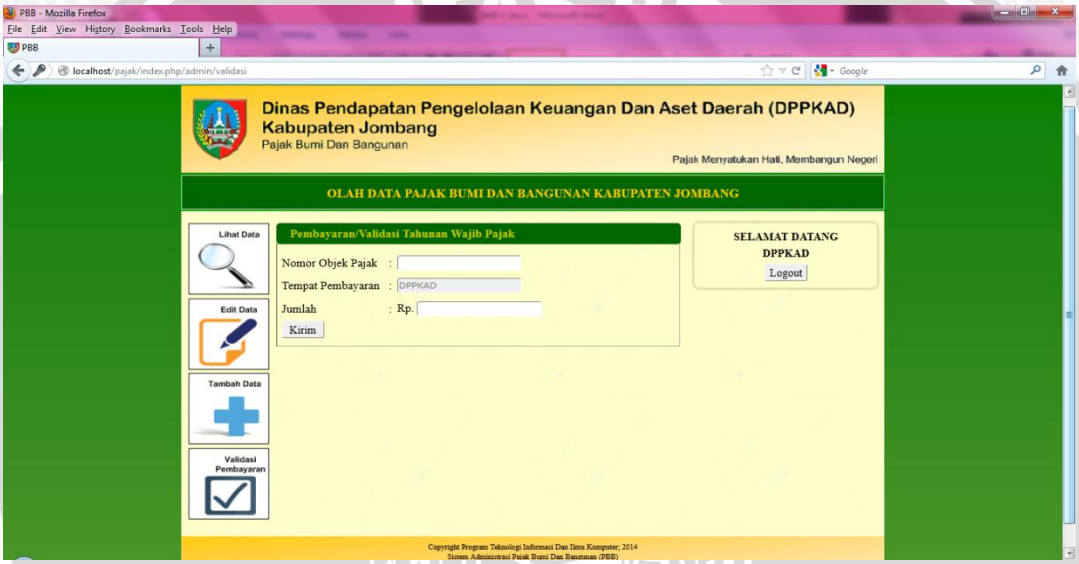
Halaman olah data wajib pajak digunakan untuk mengubah data wajib pajak jika terdapat perubahan baik dari subjek pajak atau objek pajak. Antar muka olah data wajib pajak untuk DPPKAD menampilkan formulir data wajib pajak yang akan diubah datanya. Implementasi halaman antar muka awal sebelum melakukan ubah data dapat dilihat pada gambar 5.15 dan halaman antar muka olah data wajib pajak untuk DPPKAD dapat dilihat pada gambar 5.16.

Gambar 5.15 Antar Muka Awal Sebelum Melakukan Ubah Data

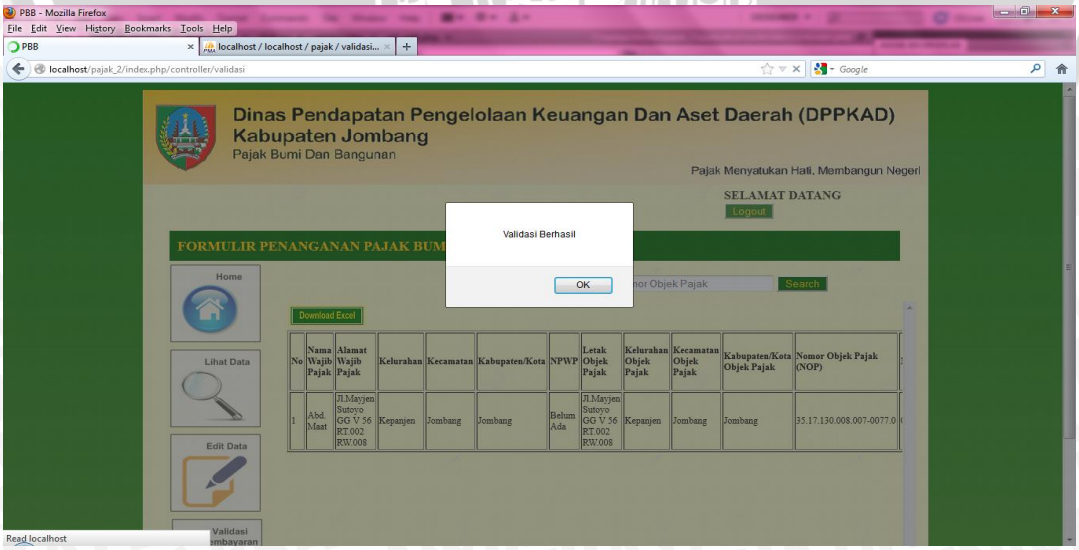
Gambar 5.16 Antar Muka Ubah Data Wajib Pajak Untuk DPPKAD

5.10.6 Antar Muka Validasi Pembayaran Untuk DPPKAD/Kelurahan/ Kecamatan

Halaman validasi pembayaran digunakan untuk memasukkan data wajib pajak yang telah melakukan pembayaran. Data yang dimasukkan berupa NOP, tempat pembayaran, dan jumlah pajak yang dibayar. Jika validasi berhasil maka akan menampilkan pesan validasi berhasil. Implementasi halaman antar muka validasi pembayaran untuk DPPKAD/Kelurahan/Kecamatan dapat dilihat pada gambar 5.17 dan halaman antar muka jika validasi berhasil dapat dilihat pada gambar 5.18.



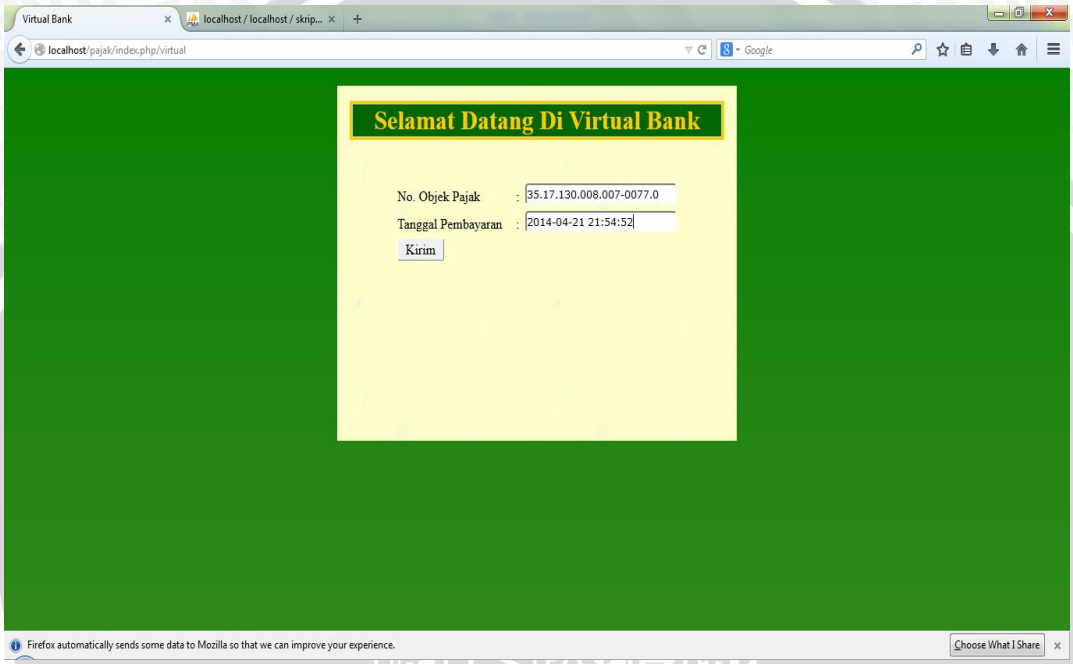
Gambar 5.17 Antar Muka Validasi Pembayaran Untuk DPPKAD/Kelurahan/Kecamatan



Gambar 5.18 Antar Muka Jika Validasi Berhasil

5.10.7 Antar Muka Validasi Pembayaran Dengan Sistem Virtual Bank

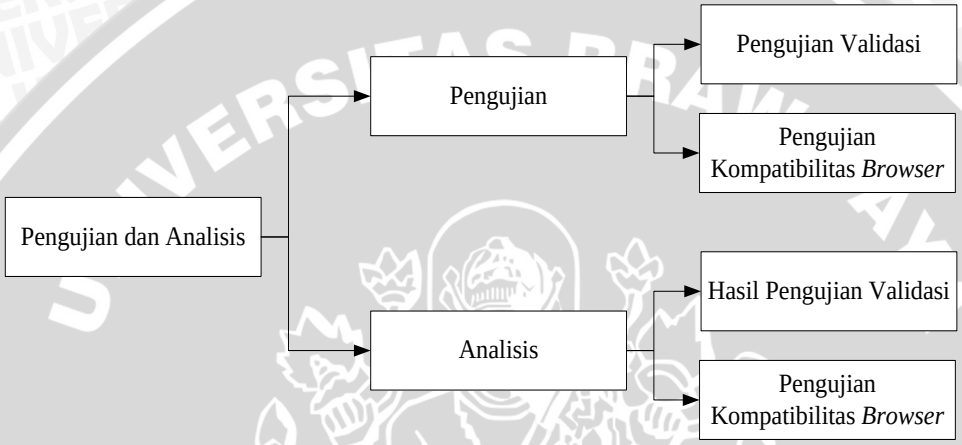
Halaman validasi pembayaran dengan Sistem Virtual Bank digunakan untuk memvalidasi pembayaran yang dilakukan oleh wajib pajak untuk mendapatkan bukti pembayaran secara sah. Data yang dimasukkan untuk proses validasi berupa NOP dan tanggal pembayaran yang dilakukan. Implementasi halaman antar muka validasi pembayaran dengan Sistem Virtual Bank dapat dilihat pada gambar 5.19.



Gambar 5.19 Antar Muka Validasi Pembayaran Dengan Sistem Virtual Bank

BAB VI
PENGUJIAN DAN ANALISIS

Bab ini membahas mengenai tahapan pengujian dan analisis sistem ODP. Pada pengujian ini melalui dua tahapan yaitu pengujian validasi dan pengujian kompatibilitas *browser*. Pada pengujian validasi akan digunakan teknik pengujian *Black-Box*. Sedangkan pengujian kompabiltas *browser* dilakukan dengan cara mengakses sistem dengan tiga *browser* pada perangkat komputer.



Gambar 6.1 Tahap-Tahap Pengujian

6.1 Pengujian

Proses pengujian dilakukan dengan dua tahapan yaitu pengujian validasi dan pengujian kompabilitas *browser*. Pengujian validasi dilakukan untuk mengetahui apakah sistem yang dibangun sudah menyediakan fungsi-fungsi yang sesuai dengan yang dibutuhkan. Pengujian kompabilitas *browser* dilakukan untuk mengetahui kesesuaian tampilan antar muka pada *browser* yang akan diuji.

6.1.1 Pengujian Validasi

Pengujian validasi digunakan untuk mengetahui apakah sistem yang dibangun sudah benar sesuai dengan yang dibutuhkan. Pemodelan persyaratan yang telah dirancang pada bab sebelumnya akan menjadi acuan untuk melakukan pengujian validasi. Pengujian validasi menggunakan metode pengujian *Black-Box* karena tidak diperlukan konsentrasi terhadap alur jalannya program/aplikasi.

6.1.1.1 Kasus Uji Validasi Olah Data Untuk Wajib Pajak

Proses olah data untuk wajib pajak meliputi lihat riwayat pembayaran dan buat pengajuan penanganan PBB. Kasus uji validasi lihat riwayat pembayaran dan buat pengajuan penanganan PBB untuk wajib pajak dapat dilihat pada Tabel 6.1 dan Tabel 6.2.

Tabel 6.1 Kasus Uji Validasi Lihat Riwayat Pembayaran Untuk Wajib Pajak

Nama kasus uji	Lihat riwayat pembayaran
Tujuan pengujian	Untuk menguji validitas kinerja dari sistem dalam menyediakan fasilitas lihat riwayat pembayaran untuk wajib pajak.
Prosedur uji	1. Wajib pajak telah melakukan autentikasi 2. Masuk ke setiap halaman
Hasil yang diharapkan	Sistem dapat menampilkan daftar riwayat pembayaran 5 tahun terakhir sesuai dengan *user* yang telah *login*.
Hasil yang didapatkan	Sistem menampilkan daftar riwayat pembayaran5 tahun terakhir yang sesuai *user* yang telah login.
Status Validasi	Valid

Tabel 6.2 Kasus Uji Validasi Buat Pengajuan Penanganan PBB Untuk Wajib Pajak

Nama kasus uji	Buat pengajuan penanganan PBB
Tujuan pengujian	Untuk menguji validitas kinerja dari sistem dalam menyediakan fasilitas buat pengajuan penanganan PBB untuk wajib pajak.
Prosedur uji	1. Wajib pajak telah melakukan autentikasi 2. Masuk ke halaman formulir 3. Wajib pajak memilih salah satu dari lima pengajuan yang diinginkan 4. Mengisi formulir yang ada sesuai pengajuan yang dipilih 5. Mengirim formulir yang telah diisi
Hasil yang	1. Sistem dapat menampilkan formulir sesuai dengan

diharapkan	pengajuan yang dipilih. 2. Sistem dapat menyimpan data yang telah diisi pada formulir pengajuan ke dalam *database*.
Hasil yang didapatkan	1. Sistem menampilkan formulir sesuai dengan pengajuan yang dipilih. 2. Sistem menyimpan data yang telah diisi pada formulir pengajuan ke dalam *database*.
Status Validasi	Valid

6.1.1.2 Kasus Uji Validasi Olah Data Wajib Pajak Untuk DPPKAD/ Kelurahan/Kecamatan

Proses olah data wajib pajak meliputi lihat data wajib pajak berdasarkan kategori, *export* data wajib pajak, memasukan data wajib pajak baru, mengubah data wajib pajak, dan validasi pembayaran PBB. Kasus uji validasi olah data wajib pajak untuk DPPKAD/Kelurahan/Kecamatan dapat dilihat pada tabel 6.3.

Tabel 6.3 Kasus Uji Validasi Lihat Data Wajib Pajak Berdasarkan Kategori

Nama kasus uji	Lihat data wajib pajak berdasarkan kategori
Tujuan pengujian	Untuk menguji validitas kinerja dari sistem dalam menyediakan fasilitas lihat data wajib pajak berdasarkan kategori. Kategorinya adalah lihat data pembayaran PBB oleh wajib pajak dan lihat data wajib pajak yang melakukan pengajuan penanganan PBB.
Prosedur uji	1. Pihak DPPKAD/Kelurahan/Kecamatan telah melakukan autentikasi 2. Masuk ke halaman lihat data 3. Mengisi kategori atau NOP atau status yang diinginkan
Hasil yang diharapkan	Sistem dapat menampilkan data wajib pajak berdasarkan kategori atau NOP atau status yang dimasukkan.
Hasil yang didapatkan	Sistem menampilkan data wajib pajak berdasarkan kategori atau NOP atau status yang dimasukkan.
Status Validasi	Valid

Tabel 6.4 Kasus Uji Validasi *Export* Data Wajib Pajak Ke Excel

Nama kasus uji	*Export* data wajib pajak ke excel
Tujuan pengujian	Untuk menguji validitas kinerja dari sistem dalam menyediakan fasilitas *export* data wajib pajak ke bentuk excel.
Prosedur uji	1. Pihak DPPKAD telah melakukan autentikasi 2. Masuk ke halaman lihat data 3. Meng-*export* data 4. Menyimpan *file* excel
Hasil yang diharapkan	Sistem dapat meng-*export* dan menyimpan data dalam bentuk excel
Hasil yang didapatkan	Sistem meng-*export* dan menyimpan data dalam bentuk excel
Status Validasi	Valid

Tabel 6.5 Kasus Uji Validasi Tambah Data Wajib Pajak

Nama kasus uji	Tambah data wajib pajak
Tujuan pengujian	Untuk menguji validitas kinerja dari sistem dalam menyediakan fasilitas tambah data wajib pajak baru.
Prosedur uji	1. Pihak DPPKAD telah melakukan autentikasi 2. Masuk ke halaman tambah data 3. Mengisi formulir data
Hasil yang diharapkan	Sistem dapat menyimpan data wajib pajak baru yang dimasukkan.
Hasil yang didapatkan	Sistem menyimpan data wajib pajak baru yang dimasukkan.
Status Validasi	Valid

Tabel 6.6 Kasus Uji Validasi Ubah Data Wajib Pajak

Nama kasus uji	Ubah data wajib pajak
Tujuan pengujian	Untuk menguji validitas kinerja dari sistem dalam menyediakan fasilitas ubah data wajib pajak.

Prosedur uji	1. Pihak DPPKAD telah melakukan autentikasi 2. Masuk ke halaman ubah data 3. Mengisi NOP yang datanya akan diubah 4. Mengisi formulir data wajib pajak berdasarkan NOP yang dimasukkan sebelumnya
Hasil yang diharapkan	1. Sistem dapat menampilkan data formulir data wajib pajak berdasarkan NOP yang dimasukkan sebelumnya. 2. Sistem dapat menyimpan data wajib pajak setelah di-ubah.
Hasil yang didapatkan	1. Sistem menampilkan data formulir data wajib pajak berdasarkan NOP yang dimasukkan sebelumnya. 2. Sistem menyimpan data wajib pajak setelah diubah.
Status Validasi	Valid

Tabel 6.7 Kasus Uji Validasi Validasi Pembayaran PBB

Nama kasus uji	Validasi pembayaran PBB
Tujuan pengujian	Untuk menguji validitas kinerja dari sistem dalam menyediakan fasilitas validasi pembayaran PBB.
Prosedur uji	1. Pihak DPPKAD/Kelurahan/Kecamatan telah melakukan autentikasi 2. Masuk ke halaman validasi pembayaran 3. Mengisi NOP, tempat pembayaran, dan jumlah PBB yang dibayar.
Hasil yang diharapkan	Sistem dapat menyimpan data validasi yang telah dimasukkan.
Hasil yang didapatkan	Sistem menyimpan data validasi yang telah dimasukkan.
Status Validasi	Valid

6.1.1.3 Kasus Uji Validasi Validasi Pembayaran PBB Oleh Sistem Virtual Bank

Sistem Virtual Bank digunakan untuk proses validasi pembayaran PBB. Kasus uji validasi sistem virtual bank dapat dilihat pada Tabel 6.8.

Tabel 6.8 Kasus Uji Validasi Validasi Pembayaran PBB Oleh Sistem Virtual Bank

Nama kasus uji	Validasi pembayaran PBB oleh Sistem Virtual Bank
Tujuan pengujian	Untuk menguji validitas kinerja dari sistem dalam menyediakan fasilitas validasi pembayaran PBB oleh sistem virtual bank.
Prosedur uji	1. Masuk ke halaman virtual bank 2. Memasukkan tanggal pembayaran dan total pembayaran
Hasil yang diharapkan	Sistem dapat menyimpan data validasi yang telah dimasukkan.
Hasil yang didapatkan	Sistem menyimpan data validasi yang telah dimasukkan.
Status Validasi	Valid

6.1.2. Pengujian Kompatibilitas *Browser*

Pengujian kompatibilitas *browser* digunakan untuk mengetahui kompabilitas antar muka sistem pada tiga *browser* pada perangkat komputer. *Browser* yang digunakan dalam pengujian kompatibilitas *browser* adalah Firefox *browser*, Chrome *browser* dan *Internet Explerer browser*. Proses pengujian kompatibilitas pada ketiga *browser* dijelaskan pada table berikut.

Tabel 6.9 Pengujian Kompatibilitas pada Firefox *Browser*

Nama kasus uji	Pengujian kompatibilitas pada Firefox *browser*
Tujuan pengujian	Untuk menguji validitas kinerja dari sistem dalam menampilkan antar muka sistem sesuai perancangan antar muka pengguna yang dilakukan dan dalam menampilkan antar muka sistem secara lengkap.
Prosedur uji	Membuka tiap halaman sistem.

Hasil yang diharapkan	Sistem dapat menampilkan antar muka sesuai perancangan antar muka pengguna yang dilakukan dan dapat menampilkan antar muka sistem secara lengkap.
Hasil yang didapatkan	Sistem dapat menampilkan antarmuka sesuai perancangan antar muka pengguna yang dilakukan dan menampilkan antar muka sistem secara lengkap.
Status Validasi	Valid

Tabel 6.10 Pengujian Kompatibilitas pada Chrome *Browser*

Nama kasus uji	Pengujian kompatibilitas pada Chrome *browser*
Tujuan pengujian	Untuk menguji validitas kinerja dari sistem dalam menampilkan antarmuka sistem sesuai perancangan antar muka pengguna yang dilakukan dan dalam menampilkan antar muka sistem secara lengkap.
Prosedur uji	Membuka tiap halaman sistem.
Hasil yang diharapkan	Sistem dapat menampilkan antar muka sesuai perancangan antar muka pengguna yang dilakukan dan dapat menampilkan antar muka sistem secara lengkap.
Hasil yang didapatkan	Sistem menampilkan antar muka sesuai perancangan antar muka pengguna yang dilakukan dan menampilkan antar muka sistem secara lengkap.
Status Validasi	Valid

Tabel 6.11 Pengujian Kompatibilitas pada *Internet Explorer Browser*

Nama kasus uji	Pengujian kompatibilitas pada *Internet Explorer browser*
Tujuan pengujian	Untuk menguji validitas kinerja dari sistem dalam menampilkan antarmuka sistem sesuai perancangan antar muka pengguna yang dilakukan dan dalam menampilkan antar muka sistem secara lengkap.
Prosedur uji	Membuka tiap halaman sistem.
Hasil yang diharapkan	Sistem dapat menampilkan antar muka sesuai perancangan antar muka pengguna yang dilakukan dan dapat menampilkan antar muka sistem secara lengkap.
Hasil yang	Sistem dapat menampilkan antar muka sesuai

didapatkan	perancangan antar muka pengguna yang dilakukan tetapi tidak dapat menampilkan antar muka secara lengkap pada antar muka yang terdapat kotak *login* karena dalam kotak *login* tidak bisa menampilkan *placeholder* atau penjelasan dari masing-masing kotak jadi akan membuat pengguna bingung dalam melakukan autentikasi.
Status Validasi	Tidak Valid

6.2 Analisis

Proses analisis bertujuan untuk mendapatkan kesimpulan dari hasil pengujian sistem ODP yang telah dilakukan. Analisis dilakukan terhadap hasil pengujian disetiap tahap pengujian. Proses analisis yang dilakukan meliputi analisis hasil pengujian validasi dan kompatibilitas *browser*.

6.2.1 Analisis Pengujian Validasi

Proses analisis terhadap hasil pengujian validasi dilakukan dengan melihat kesesuaian antara hasil kinerja sistem dengan daftar persyaratan. Hasil pengujian validasi pada semua Subbab 6.1.1 dengan metode *black-box testing* adalah sistem dapat menjalankan setiap fungsi sistem sesuai dengan pemodelan persyaratan.

6.2.2 Analisis Pengujian Kompatibilitas *Browser*

Hasil pengujian kompatibilitas *browser* pada perangkat komputer pada Subbab 6.1.2 adalah seluruh *browser* dapat menampilkan antar muka sesuai dengan perancangan antar muka pengguna yang dilakukan. Pada *browser* Firefox dan *browser* Chrome juga dapat menampilkan antar muka secara lengkap tetapi pada *browser Internet Explorer* tidak dapat menampilkan antar muka secara lengkap. Hal ini disebabkan karena pada *browser Internet Explorer* tidak dapat menampilkan secara sempurna kotak *login.* Pada kotak *login* seharusnya terdapat penjelasan pada masing-masing kotak untuk memudahkan autentikasi pengguna tetapi pada *Internet Explorer* tidak dapat menampilkan penjelasan masing-masing kotak. Hal ini disebabkan karena *javascript* yang digunakan pada sistem tidak kompatibel dengan *javascript* untuk *browser Internet Explorer* sehingga untuk mendapatkan hasil yang sempurna maka disarankan untuk menggunakan Firefox *browser* dan Chrome *browser*.

BAB VII
PENUTUP

7.1 Kesimpulan

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

1. Perancangan sistem ODP dan sistem virtual Bank telah dibuat sesuai pemodelan persyaratan. Pemodelan persyaratan dilakukan melalui dua tahap, yaitu tahap pemodelan *use case* dan penyusunan daftar persyaratan. Tahap-tahap pemodelan *use case* meliputi identifikasi aktor, identifikasi *use case* dan diagram *use case* dan spesifikasi *use case*. Sedangkan daftar persyaratan meliputi penyusunan daftar persyaratan fungsional sesuai diagram *use* case dan daftar non-fungsional yang berfungsi untuk menyusun daftar batasan fungsi sistem.
2. Implementasi sistem ODP menggunakan model *reuse-oriented software engineering*. Dengan model tersebut mudah dalam mengembangkan sistem dengan langsung menggunakan fasilitas-fasilitas yang sudah disediakan. Sistem ODP dan sistem virtual Bank telah dibuat sesuai perancangan dan diimplementasikan dari komponen-komponen yang telah ditentukan, seperti: *framework CodeIgniter* dan SOA.
3. Berdasarkan hasil pengujian validasi dengan metode *Black-box testing*, sistem dapat menjalankan seluruh fungsi, seperti lihat riwayat pembayaran, lihat data wajib pajak, validasi pembayaran, dan lain-lain sesuai pemodelan persyaratan. Pengujian kompatibilitas *browser* diuji menggunakan tiga *browser* pada perangkat komputer, yaitu Firefox b*rowser*, Chrome *browser* dan *Internet Explorer browser*. Berdasarkan hasil pengujian kompatibilitas *browser*, ketiga *browser* dapat menampilkan seluruh antar muka sesuai dengan perancangan antar muka pengguna. Pada Firefox *browser* dan Chrome *browser* juga dapat menampilkan antar muka secara lengkap. Namun, pada *Internet Explorer browser* tidak dapat menampilkan antar muka secara lengkap karena pada kotak *login* tidak dapat menampilkan *placeholder* atau penjelasan dari masing-

masing kotak *login.* Oleh karena itu disarankan untuk menggunakan Firefox *browser* dan Chrome *browser* untuk mendapatkan hasil yang sesuai.

7.2 Saran

Sistem ODP masih dibuat sebagai prototipe dan belum dapat terealisasikan karena masih perlu dilakukan pengembangan sistem lebih lanjut. Oleh karena itu, beberapa saran yang diberikan untuk pengembangan sistem ODP tersebut, antara lain:

1. Memberikan implementasi sistem yang dapat diakses pada *browser mobile phone* sesuai dengan spesifikasi layar.
2. ,Sistem pembayaran yang dapat dilakukan secara *online* sehingga wajib pajak tidak perlu datang ke tempat pembayaran.
3. Pengajuan penanganan PBB dilengkapi dengan pengiriman kelengkapan berkas secara *online*, seperti *fotocopy* Kartu Keluarga (KK), *fotocopy* Kartu Tanda Penduduk (KTP), dan lain-lain sehingga wajib pajak tidak perlu lagi mengurus kelengkapan berkas untuk pengajuan penanganan PBB secara manual dengan datang ke DPPKAD.

DAFTAR PUSTAKA

[ADE - 09] Ade. *Apa itu CodeIgniter?*. 2009. http://www.kuliahit.com/kuliahit/article/24/_Apa-itu-CodeIgniter. Diakses pada tanggal 19 Januari 2014.

[ANY - 09] *Anonym*. 2009. *Pajak Bumi dan Bangunan*. http://www.pajak.go.id. Diakses pada tanggal 19 Januari 2014.

[DEV - 11] Deviana, Hartati. 2011. *Penerapan Xml Web Service pada Sistem Distribusi Barang*, Generic, Vol. 6, No. 2, hal. 55-62.

[DSR - 03] Dharwiyanti, Sri. Romi Satria Wahono. 2003. Pengantar *Unified Modeling Language (UML)*. http://setia.staff.gunadarma.ac.id, Diakses pada tanggal 19 Januari 2013.

[GTS - 13] Gunawan, Tjahjanulin, Siswidiyanto. 2013. *Analisis Tunggakan Pajak Bumi Dan Bangunan*. *Journal of Public Administration Research* (JOPAR), Vol 1, No.1, Malang.

[PRE - 01] Pressman, Roger S. 2001. *Software Engineering A Practitioner's Approach*. 5th Edition. Boston: McGraw-Hill.

[PSB - 12] Permana, Sigit Budi. 2012. *Pengertian Service Oriented Architecture (SOA)*. http://cgeduntuksemua.blogspot.com/2012/04/pengertian-service-oriented.html. Diakses pada tanggal 18 Maret 2014.

[PTK - 05] Priyambodo, Tri Kuntoro. 2005, *Implementasi Web-Service Untuk Pengembangan Layanan Pariwisata Terpad*u. TEKNOIN, Vol.10, No.2, Yogyakarta.

[ANM - 12] *Anonym*. 2012. *PHP web services with NuSOAP*. http://www.nusphere.com/php_script/nusoap.htm. Diakses pada 11 Maret 2014.

[RJB - 98] Rumbaugh, James, Jacobson, Ivar, Booch, Grady. 1998, *"The Unified Modeling Language Reference Manual"*, Addison Wesley, Canada.

[SKG - 08] Sahin,K., Gumusay, M.U. 2008. *Service Oriented Architecture* (SOA) *Based Web Service For Geograpic Information Systems.*

The Internasional Archives of The Photogrammetry Remote Sensing and Spatial Information Sciences Vol. XXXVII Part B2.

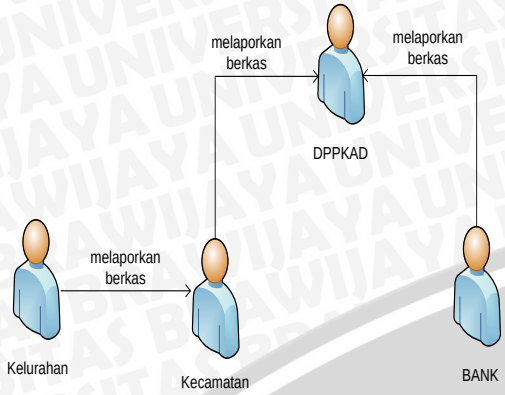
[SMI - 11] Sommerville, Ian. 2011. *Software Engineering*, 9th edition. Addison-Wesley, New York.

LAMPIRAN

Pembayaran PBB di Kabupaten Jombang dapat dilakukan dengan beberapa cara, yaitu melalui DPPKAD atau Kelurahan atau Kecamatan setempat dan melalui Bank Jatim. Apabila Wajib Pajak melakukan pembayaran melalui DPPKAD atau Kelurahan atau Kecamatan, maka Wajib Pajak hanya menerima bukti pembayaran sementara karena bukti pembayaran yang sah hanya dikeluarkan oleh pihak Bank Jatim saja. Jadi setelah Wajib Pajak menerima bukti pembayaran sementara, Wajib Pajak harus melakukan pelaporan ke Bank Jatim untuk dapat mendapatkan bukti pembayaran yang sah berupa Surat Tanda Terima Setoran (STTS).

Tabel 1 Perbedaan Proses Bisnis Saat Ini Dengan Proses Bisnis Yang Diharapakan Dengan Adanya Sistem ODP dan Sistem Virtual Bank

Proses Bisinis Saat Ini	Proses Bisinis Sistem ODP dan Sistem Virtual Bank
Pelaporan pembayaran PBB ke Dinas Pendapatan Pengelolaan Keuangan dan Aset Daerah (DPPKAD) di Kabupaten Jombang oleh pihak Kelurahan/Kecamatan dan Bank Jatim dilakukan secara manual. Pihak Kelurahan/Kecamatan dan Bank Jatim melakukan pelaporan setiap seminggu sekali ke DPPKAD. Berkas yang dilaporakan berupa data Wajib Pajak yang melakukan pembayaran dan nantinya berkas dari pihak Kelurahan/Kecamatan akan dicocokkan secara manual oleh pihak DPPKAD dengan berkas dari Bank Jatim.	Pihak Kelurahan/Kecamatan tidak perlu lagi melakukan pelaporan data pembayaran PBB dan DPPKAD tidak perlu lagi melakukan pengolahan data wajib pajak secara manual karena sistem ODP menyediakan fasilitas validasi pembayaran dan pengolahan data wajib pajak. Setiap data pembayaran yang dilakukan dan pengolahan data wajib pajak akan disimpan dalam *database*. Data pembayaran yang disimpan dapat dilihat melalui sistem ODP dan di-*export* oleh pihak DPPKAD sebagai simpanan data tahunan. Sedangkan pihak Bank dalam

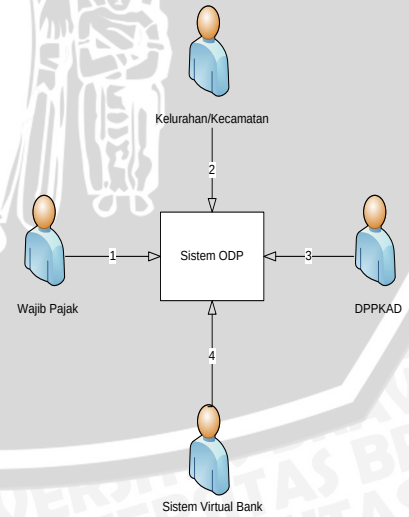


Mekanisme Pelaporan Pembayaran PBB di Kabupaten Jombang Saat Ini

Selain pelaporan data pembayaran PBB yang dilakukan secara manual, pengolahan data wajib pajak seperti mengubah data wajib pajak dan menambah data pajak baru juga masih dilakukan secara manual. Pihak DPPKAD masih melakukan pengolahan data wajib pajak melalui *Microsoft excel* dengan mencari dan melihat data wajib pajak satu per satu.

melakukan validasi pembayaran PBB melalui sistem virtual Bank. Sistem virtual Bank ini dibuat sebagai simulasi dari sebuah sistem Bank yang sebenarnya.

Dengan adanya sistem ODP dan sistem virtual Bank, pihak DPPKAD tidak perlu lagi mencocokkan data pembayaran PBB yang ada di Kelurahan/Kecamatan dan DPPKAD dengan data pembayaran PBB yang divalidasi oleh Bank secara manual. Hal ini dikarenakan setiap data pembayaran PBB yang ada di Kelurahan/Kecamatan dan DPPKAD serta data validasi pembayaran yang dilakukan pihak Bank tersimpan dalam satu tabel dalam *database*.



Proses Bisnis Sistem ODP

Keterangan:

1. Wajib Pajak dapat melihat

	beberapa informasi tentang pembayaran PBB, melihat riwayat pembayaran yang telah dilakukan, dan mengajukan permohonan penanganan PBB, seperti permohonan pengajuan data baru, permohonan pengurangan biaya PBB, permohonan mutasi/pembetulan objek/subjek pajak, permohonan salinan SPPT, dan permohonan keberatan atas pengenaan PBB yang diberikan dengan mengisi formulir yang disediakan oleh sistem. 2. Pihak Kelurahan/Kecamatan memasukkan data Wajib Pajak yang melakukan pembayaran Kelurahan/kecamatan setempat melalui sistem jadi pihak Kelurahan/Kecamatan tidak perlu lagi mendata dan melaporkan secara manual. 3. Pihak DPPKAD dapat mengolah data Wajib Pajak mulai dari melihat data wajib pajak, menambah data, dan mengubah data wajib pajak. 4. Pihak Bank Jatim memasukkan data Wajib Pajak melalui sistem Virtual Bank untuk mem-validasi pembayaran. Sistem

	virtual Bank mengakses data pajak dari sistem ODP untuk mencocokkan data wajib pajak yang telah melakukan pembayaran PBB untuk dilakukan validasi oleh pihak Bank. Sistem virtual Bank menggunakan arsitektur SOA yang berfungsi untuk menjembatani antara sistem ODP dan sistem virtual Bank yang sedang berjalan dalam menggunakan data.

Detail Spesifikasi *Use Case*

Tabel 2 Detail Spesifikasi *Use Case* Lihat Seputar Pajak

Nama	Lihat Informasi Seputar PBB
Tujuan	Untuk melihat semua informasi seputar PBB
Deskripsi	*Use case* ini menjelaskan pengguna dapat melihat semua informasi seputar PBB mulai dari penjelasan tentang PBB, pengumuman seputar PBB dan belajar mengenai aturan-aturan PBB.
Aktor	Masyarakat umum dan wajib pajak
Alur Dasar	1. Aktor mengakses sistem ODP 2. Aktor memasuki halaman seputar pajak
Kemungkinan Alur Lain	-
Skenario	Jika aktor mengakses sistem ODP pada alur utama dan memasuki halaman seputar pajak pada alur dasar, maka sistem menampilkan halaman seputar pajak.
Kondisi Awal	Sistem menampilkan halaman utama
Kondisi Akhir	Sistem menampilkan halaman seputar pajak

Tabel 3 Detail Spesifikasi *Use Case* Buat Pengajuan Objek Pajak Baru

Nama	Buat pengajuan objek pajak baru
Tujuan	Untuk membuat pengajuan data pajak baru
Deskripsi	*Use case* ini menjelaskan pengguna dapat membuat pengajuan objek pajak baru dengan mengisi formulir yang tersedia pada sistem.
Aktor	Masyarakat umum dan wajib pajak
Alur Dasar	1. Aktor memasuki halaman formulir 2. Aktor memasuki halaman pengajuan data baru 3. Aktor mengisi formulir 4. Aktor mengirim formulir
Kemungkinan Alur Lain	Formulir belum diisi dengan lengkap
Skenario	Aktor mengisi formulir pengajuan objek pajak baru pada alur utama. Jika aktor tidak mengisi data dengan lengkap pada kemungkinan alur lain maka akan muncul pesan data tidak lengkap.
Kondisi Awal	Sistem menampilkan formulir pengajuan objek pajak baru
Kondisi Akhir	Sistem mengirim pengajuan objek pajak baru

Tabel 4 Detail Spesifikasi *Use Case* Lihat Riwayat Pembayaran

Nama	Lihat riwayat pembayaran
Tujuan	Untuk melihat riwayat pemabayaran PBB yang telah dilakukan
Deskripsi	*Use case* ini menjelaskan pengguna dapat melihat riwayat pembayaran PBB selama lima tahun terakhir.
Aktor	Wajib pajak
Alur Dasar	1. Aktor mengakses sistem ODP 2. Aktor melihat tabel riwayat pembayaran pada setiap halaman
Kemungkinan Alur Lain	-
Skenario	Aktor mengakses sistem ODP pada alur utama sehingga dapat melihat tabel riwayat pembayaran di setiap halaman

	sistem ODP pada alur dasar.
Kondisi Awal	Aktor telah melakukan autentikasi.
Kondisi Akhir	Sistem menampilkan tabel riwayat pembayaran di setiap halaman.

Tabel 5 Detail Spesifikasi *Use Case* Buat Pengajuan Pengurangan PBB

Nama	Buat pengajuan pengurangan PBB
Tujuan	Untuk membuat pengajuan pengurangan PBB
Deskripsi	*Use case* ini menjelaskan pengguna dapat mengajukan permohonan pengurangan PBB karena beberapa alasan.
Aktor	Wajib pajak
Alur Dasar	1. Aktor memasuki halaman formulir 2. Aktor memilih pengajuan pengurangan PBB 3. Aktor mengisi formulir 4. Aktor mengirim formulir
Kemungkinan Alur Lain	Data yang diisi tidak lengkap
Skenario	Aktor memasuki halaman formulir pada alur utama. Kemudian aktor memilih pengajuan pengurangan PBB dan mengisi formulir pengajuan pengurangan PBB pada alur dasar. Setelah itu aktor mengirim formulir yang sudah diisi pada alur dasar. Jika aktor tidak mengisi data dengan lengkap pada kemungkinan alur lain maka akan muncul pesan data tidak lengkap.
Kondisi Awal	Aktor telah melakukan autentikasi.
Kondisi Akhir	Sistem mengirim formulir pengajuan pengurangan PBB yang selesai diisi.

Tabel 6 Detail Spesifikasi *Use Case* Buat Pengajuan Keberatan Atas Pengenaan PBB

Nama	Buat pengajuan keberatan atas pengenaan PBB
Tujuan	Untuk membuat pengajuan keberatan atas pengenaan PBB
Deskripsi	*Use case* ini menjelaskan aktor dapat mengajukan keberatan

	atas pengenaan PBB apabila PBB yang dikenakan tidak sesuai dengan kondisi yang ada. Pengajuan keberatan atas pengenaan PBB ini dapat dilakukan setelah melakukan autentikasi.
Aktor	Wajib pajak
Alur Dasar	1. Aktor memasuki halaman formulir 2. Aktor memilih pengajuan keberatan atas PBB 3. Aktor mengisi formulir 4. Aktor mengirim formulir
Kemungkinan Alur Lain	Data yang diisi tidak lengkap
Skenario	Aktor memasuki halaman formulir pada alur utama. Kemudian aktor memilih pengajuan keberatan atas PBB dan mengisi formulir pengajuan pengenaan PBB pada alur dasar. Setelah itu aktor mengirim formulir yang sudah diisi pada alur dasar. Jika aktor tidak mengisi data dengan lengkap pada kemungkinan alur lain maka akan muncul pesan data tidak lengkap.
Kondisi Awal	Aktor telah melakukan autentikasi.
Kondisi Akhir	Sistem mengirim formulir pengajuan keberatan atas pengenaan PBB yang selesai diisi.

Tabel 7 Detail Spesifikasi *Use Case* Buat Pengajuan Mutasi PBB

Nama	Buat pengajuan mutasi PBB
Tujuan	Untuk membuat pengajuan mutasi PBB
Deskripsi	*Use case* ini menjelaskan pengguna dapat mengajukan permohonan mutasi objek/subjek PBB karena beberapa alasan seperti hibah/waris/jual beli.
Aktor	Wajib pajak
Alur Dasar	1. Aktor memasuki halaman formulir 2. Aktor memilih pengajuan mutasi PBB 3. Aktor mengisi formulir 4. Aktor mengirim formulir
Kemungkinan	Data yang diisi tidak lengkap

Alur Lain	
Skenario	Aktor memasuki halaman formulir pada alur utama. Kemudian aktor memilih pengajuan mutasi PBB dan mengisi formulir pengajuan mutasi PBB pada alur dasar. Setelah itu aktor mengirim formulir yang sudah diisi pada alur dasar. Jika aktor tidak mengisi data dengan lengkap pada kemungkinan alur lain maka akan muncul pesan data tidak lengkap.
Kondisi Awal	Aktor telah melakukan autentikasi.
Kondisi Akhir	Sistem mengirim formulir pengajuan mutasi PBB yang selesai diisi.

Tabel 8 Detail Spesifikasi *Use Case* Buat Pengajuan Salinan SPPT

Nama	Buat pengajuan salinan SPPT
Tujuan	Untuk membuat pengajuan salinan SPPT
Deskripsi	*Use case* ini menjelaskan aktor dapat mengajukan salinan SPPT apabila terjadi kehilangan SPPT atau yang lainnya. Pengajuan salinan SPPT ini dapat dilakukan setelah melakukan autentikasi.
Aktor	Wajib pajak
Alur Dasar	1. Aktor memasuki halaman formulir 2. Aktor memilih pengajuan salinan SPPT 3. Aktor mengisi formulir 4. Aktor mengirim formulir
Kemungkinan Alur Lain	Data yang diisi tidak lengkap
Skenario	Aktor memasuki halaman formulir pada alur utama. Kemudian aktor memilih pengajuan salinan SPPT dan mengisi formulir pengajuan salinan SPPT pada alur dasar. Setelah itu aktor mengirim formulir yang sudah diisi pada alur dasar. Jika aktor tidak mengisi data dengan lengkap pada kemungkinan alur lain maka akan muncul pesan data tidak lengkap.

Kondisi Awal	Aktor telah melakukan autentikasi.
Kondisi Akhir	Sistem mengirim formulir pengajuan salinan SPPT yang selesai diisi.

Tabel 9 Detail Spesifikasi *Use Case* Lihat Data Wajib Pajak

Nama	Lihat Data Wajib Pajak
Tujuan	Untuk melihat data wajib pajak
Deskripsi	*Use case* ini menjelaskan aktor dapat melihat data pembayaran wajib pajak, data pengajuan objek pajak baru, data pengajuan pengurangan PBB, data pengajuan keberatan atas pengenaan PBB, dan data pengajuan salinan SPPT. Lihat data wajib pajak dapat dilakukan setelah melakukan autentikasi.
Aktor	DPPKAD/Kelurahan/Kecamatan
Alur Dasar	1. Aktor memasuki halaman lihat data 2. Aktor memasukkan kategori 3. Aktor memasukkan NOP 4. Aktor memasukkan status 5. Aktor meng-*export* data wajib pajak dalam bentuk excel
Kemungkinan Alur Lain	1. Data untuk kategori yang diisi tidak lengkap 2. Salah kategori atau NOP yang dimasukkan salah
Skenario	1. Aktor memasuki halaman lihat data pada alur utama. Kemudian aktor mengisi kategori atau NOP atau status apabila ingin melihat berdasarkan kategori atau NOP atau status pada alur dasar. Jika aktor tidak mengisi data dengan lengkap pada kemungkinan alur lain maka akan muncul pesan data tidak lengkap. 2. Aktor mengisi kategori atau NOP atau status apabila ingin melihat berdasarkan kategori atau NOP atau status pada alur utama. Jika kategori atau NOP yang dimasukkan salah pada kemungkinan alur lain maka akan muncul pesan data tidak ditemukan.
Kondisi Awal	Aktor telah melakukan autentikasi.
Kondisi Akhir	Sistem menampilkan data wajib pajak sesuai kategori atau NOP atau status yang dipilih.

Tabel 10 Detail Spesifikasi *Use Case* Olah Data Wajib Pajak

Nama	Olah Data Wajib Pajak
Tujuan	Untuk mengolah data wajib pajak
Deskripsi	*Use case* ini menjelaskan aktor dapat mengolah data wajib pajak yang meliputi menambah data wajib pajak baru dan mengubah data wajib pajak. Olah data wajib pajak dapat dilakukan setelah melakukan autentikasi.
Aktor	DPPKAD
Alur Dasar	1. Aktor memasuki halaman tambah atau ubah data 2. Aktor memasukkan kategori 3. Aktor memasukkan NOP 4. Aktor mengisi formulir data
Kemungkinan Alur Lain	1. Data yang diisi tidak lengkap 2. Kategori atau NOP atau data yang diisi pada formulir salah
Skenario	1. Aktor memasuki halaman tambah data atau ubah data pada alur utama. Kemudian aktor mengisi kategori atau NOP dan formulir data apabila ingin menambah atau mengubah data pada alur dasar. Jika aktor tidak mengisi data dengan lengkap pada kemungkinan alur lain maka akan muncul pesan data tidak lengkap. 2. Aktor mengisi kategori atau NOP dan formulir data apabila ingin menambah atau mengubah data pada alur utama. Jika kategori atau NOP dan data pada formulir yang dimasukkan salah pada kemungkinan alur lain maka akan muncul pesan data tidak ditemukan.
Kondisi Awal	Aktor telah melakukan autentikasi.
Kondisi Akhir	Sistem menampilkan pesan apakah berhasil dalam menambah atau mengubah data.

Tabel 11 Detail Spesifikasi *Use Case* Validasi Pembayaran

Nama	Validasi Pembayaran
Tujuan	Untuk mem-validasi pembayaran PBB
Deskripsi	*Use case* ini menjelaskan aktor dapat melakukan validasi pembayaran PBB. Validasi pembayaran dapat dilakukan setelah melakukan autentikasi.
Aktor	DPPKAD/Kelurahan/Kecamatan
Alur Dasar	1. Aktor memasuki halaman validasi pembayaran 2. Aktor mengisi NOP, tempat pembayaran dan jumlah PBB yang dibayar

	3. Aktor mencetak bukti pembayaran sementara
Kemungkinan Alur Lain	1. Data yang diisi tidak lengkap 2. NOP atau tempat pembayaran atau jumlah PBB yang dimasukkan salah
Skenario	1. Aktor memasuki halaman validasi pembayaran pada alur utama. Kemudian aktor mengisi NOP, tempat pembayaran dan jumlah PBB pada alur dasar. Jika aktor tidak mengisi data dengan lengkap pada kemungkinan alur lain maka akan muncul pesan data tidak lengkap. 2. Aktor mengisi NOP, tempat pembayaran dan jumlah PBB pada alur utama. Jika NOP, tempat pembayaran dan jumlah PBB yang dimasukkan salah pada kemungkinan alur lain maka akan muncul pesan data tidak ditemukan.
Kondisi Awal	Aktor telah melakukan autentikasi.
Kondisi Akhir	Sistem menampilkan pesan apakah berhasil dalam mem-validasi pembayaran.

Tabel 12 Detail Spesifikasi *Use Case* Validasi Pembayaran Dengan Sistem Virtual Bank

Nama	Validasi Pembayaran
Tujuan	Untuk mem-validasi pembayaran PBB
Deskripsi	*Use case* ini menjelaskan aktor dapat melakukan validasi pembayaran pajak yang dilakukan oleh wajib pajak. Sistem ini menggunakan arsitektur SOA dengan menggunakan *web service Simple Object Access Protocol* (SOAP) dalam mengakses data wajib pajak.
Aktor	Petugas Bank
Alur Dasar	1. Aktor memasuki halaman sistem virtual Bank 2. Aktor mengisi NOP, tempat pembayaran dan jumlah PBB yang dibayar
Kemungkinan Alur Lain	1. Data yang dimasukkan tidak lengkap 2. NOP atau tempat pembayaran atau jumlah PBB yang dimasukkan salah
Skenario	1. Aktor memasuki halaman virtual Bank pada alur utama. Kemudian aktor mengisi NOP, tempat pembayaran dan jumlah PBB pada alur dasar. Jika aktor tidak mengisi data dengan lengkap pada kemungkinan alur lain maka akan muncul pesan data tidak lengkap. 2. Aktor mengisi NOP, tempat pembayaran dan jumlah PBB

	pada alur utama. Jika NOP, tempat pembayaran dan jumlah PBB yang dimasukkan salah pada kemungkinan alur lain maka akan muncul pesan data tidak ditemukan.
Kondisi Awal	Sistem menampilkan halaman sistem virtual Bank.
Kondisi Akhir	Sistem menampilkan pesan apakah berhasil dalam mem-validasi pembayaran.