BAB II
TINJAUAN PUSTAKA

2.1 Tinjauan Pustaka

Pada tahun 2015, sebuah *proceeding* yang dibuat pada Hawaii International Conference on System Science dengan judul "New Implications for Customization of ERP Systems" (Uppström, Lonn, Hoffsten, & Thorstrom, 2015) membahas tentang diperlukannya kustomisasi dari sebuah sistem ERP yang diimplementasi ke dalam sebuah perusahaan untuk menyesuaikan sistem dengan proses bisnis. Umumnya proses kustomisasi pada sistem ERP selama bertahun-tahun telah mendorongnya *design* ERP dengan *model* yang lebih fleksibel, serta berbagai cara untuk mengkostumisasi sebuah sistem ERP semakin bertambah dan mudah dicapai. Konklusi dari penelitian yang telah dilakukan menunjukkan perlunya kustomisasi serta kemudahan melakukan kustomisasi tersebut pada sebuah sistem ERP untuk memenuhi semakin meningkatnya permintaan berbagai perusahaan atas sistem yang lebih fleksibel,

Sebuah penelitian dengan judul "OpenERP/Odoo - An Open Source Concept to ERP Solution" oleh Ganesh, Shanil, Sunitha, & Midhundas (2016) menjelaskan keunggulan menggunakan sebuah sistem ERP yang bersifat *open source* dibandingkan dengan yang bersifat komersial. Penelitian ini didasarkan pada masalah mengenai sistem ERP komersial. Berbagai vendor yang dikenal menyediakan jasa ERP bagi perusahaan termasuk Oracle dan Microsoft. Mereka menyediakan solusi sistem ERP yang dikostumisasi berdasarkan permintaan klien, dengan harga yang sangat tinggi. Solusi ini tidak ekonomis bagi perusahaan berskala kecil dan menengah. Munculnya ERP *open source* dan semakin banyak

penggunaannya pada perusahaan baik berskala kecil, menengah maupun besar didasari oleh beberapa keunggulan sistem ERP *open source* dibandingkan ERP komersial. Pada penelitian ini, terdapat 5 keunggulan utama dari ERP *open source*. Yang pertama, adalah tentunya, biaya yang jauh lebih rendah daripada ERP komersial. Selanjutnya adalah fleksibilitas. Karena sistem ERP *open source* memiliki *source code* yang terbuka, perusahaan dapat melakukan kostumisasi dari *framework* ERP tersebut dalam, sehingga berbagai fitur-fitur yang dibutuhkan perusahaan dapat diimpelentasikan ke dalam sistem ERP yang sudah tersedia dengan mudah. Keunggulan yang ketiga yaitu, karena merupakan *software open source,* setiap perusahaan mempunyai kendali total pada sistem ERP serta *source code*-nya dan dokumentasi yang terkait dengan penggunaan ataupun bagian *technical* tersedia gratis di internet, serta berbagai informasi bisa didapatkan dari komunitas *open source*. Ini mengurangi ketergantungan perusahaan yang mengimplementasi sistem ERP dengan *vendor* dari sistem tersebut. Kualitas yang tinggi juga merupakan keunggulan dari ERP *open source*, karena seluruh kontribusi kepada sistem ERP tersebut adalah dari *developer* dari seluruh dunia yang ikut berpartisipasi dalam mengembangkan sistem *open source* tersebut. Yang terakhir, adalah kemudahan untuk melakukan *upgrade*. Karena kostumisasi dari sistem ERP *open source* tidak memerlukan *source code* dari *framework* diubah, sebuah sistem ERP yang sudah berjalan pada sebuah *server* dapat melakukan *upgrade* yang tidak bersifat mengganggu terhadap sistem internal dari ERP tersebut. Kesimpulan dari penelitian tersebut adalah sistem ERP *open source* akan diadopsi oleh semakin banyak perusahaan, dan ERP *open source* telah menjadi salah satu solusi sistem

informasi perusahaan yang berkualitas tinggi, setara dengan sistem ERP komersial yang ada pada saat ini.

Dalam sebuah sistem informasi yang dirancang oleh Zuana & Sidharta (2014) untuk PT. Rajawali Tehnik, sistem tersebut dirancang untuk mengatur data gaji karyawan serta perpajakan PPh 21 dalam bentuk aplikasi Windows Presentation Format (WPF), yang diimplementasi ke dalam perusahaan tersebut untuk mendukung dan memudahkan kalkulasi dari gaji karyawan serta perhitungan nilai pemotongan pajak penghasilan. Kalkulasi akurat dari perhitungan gaji serta PPh 21 membantu menyelesaikan kendala perusahaan di mana pada awalnya, perhitungan gaji serta pajak penghasilan karyawan menggunakan aplikasi pengolah angka Microsoft Excel, di mana kesalahan *input* data sering terjadi, dan kesalahan dari data tersebut menghasilkan hasil akhir kalkulasi yang tidak tepat. Selain mendukung perhitungan gaji serta PPh 21 dari perusahaan, sistem tersebut dapat menghasilkan laporan gaji serta laporan pajak.

Sebuah jurnal berjudul "Jurnal Ilmiah Komputer Akutansi" yang dibuat oleh Endaryati et al., (2015) membahas tentang implementasi sebuah Sistem Informasi Pengolahan Data Gaji dan Data Perhitungan PPh 21 pada CV. Sinar Jasa Teknik Ngaliyan Semarang. Implementasi sistem tersebut didasari oleh permasalahan mengenai pengolahan data gaji karyawan. Sistem awal yang digunakan merupakan *software* Microsoft Excel, di mana staf administrasi mengalami kesulitan mengolah data gaji, karena jika terjadi sebuah perubahan dalam data, staf harus secara manual mengubah data dari lebih dari 150 orang karyawan. Hal tersebut sangat memperlambat kinerja dan produktivitas staf administrasi karena jumlah waktu yang terbuang untuk melakukan operasi *update*

data secara manual pada seluruh karyawan. Masalah kedua yang dihadapi staf administrasi merupakan perhitungan PPh pasal 21, di mana proses tersebut masih manual serta memakan waktu banyak. Staf harus mengecek slip gaji bulanan dari setiap karyawan untuk menentukan nominal dari pemotongan gaji tersebut. Dan terakhir, pembuatan laporan gaji masih menggunakan Microsoft Word, di mana staf administrasi harus secara manual meng-*input* data dari gaji karyawan yang tersimpan pada Microsoft Excel, ke dalam Microsoft Word, dan proses tersebut memakan waktu yang lama. Solusi yang berupa sistem baru dalam pembahasan jurnal tersebut berupa aplikasi *desktop* yang dibuat dengan bahasa pemrograman Microsoft Visual Basic 6.0 beserta DataReport untuk penyusunan struktur laporan, serta menggunakan *database* MySQL. Sistem baru yang telah diimplementasi dapat menyimpan data karyawan, data gaji, serta menyediakan fitur kalkulasi gaji dan PPh 21 serta men-*generate* laporan gaji.

Sebuah sistem perhitungan Pajak Penghasilan (PPh) Pasal 21 yang dirancang serta diimplementasi kepada Kantor Kecamatan Todanan di Kabupaten Blora oleh Nurjayanti & Susanto (2015) merupakan sebuah aplikasi yang dibuat dengan bahasa pemrograman Visual Basic 6.0, serta menggunakan *database* MySQL. Sistem tersebut dibuat berdasarkan latar belakang masalah yaitu kendala yang dialami staf dari kantor kecamatan tersebut dalam melakukan PPh 21, yang pada awalnya masih menggunakan Microsoft Excel. Hal tersebut memperlambat kinerja serta memakan waktu yang lama. Kebutuhan akan data yang terintegrasi dengan benar tidak dapat dicapai dengan berbagai operasi manual pada Excel. Tidak hanya itu, hasil kalkulasi juga rawan kesalahan, dan *data* yang didapatkan bisa menjadi tidak *up-to-date*. Hasil akhir dari sistem yang diimplementasi

merupakan aplikasi berbasis *desktop*, yang mempunyai *data* karyawan, *form* perhitungan pajak, serta dapat membuat sebuah laporan mengenai *data* karyawan dan slip setoran pajak.

Untuk merangkum seluruh kesimpulan dari kelima penelitian yang telah diuraikan di atas, di bawah ini merupakan tabel kesimpulan dari seluruh tinjauan pustaka untuk dibandingkan dengan proyek Kerja Praktek yang akan dilakukan:

Tabel 1 Rangkuman kesimpulan dari kelima hasil penelitian yang telah ditinjau

No.	Nama pengarang	Judul Penelitian	Tahun	Kesimpulan Penelitian
1.	Uppström, Elin Lonn, Carl-Mikael Hoffsten, Madeleine Thorstrom, Joakim	New Implications for Customization of ERP Systems.	2015	Kustomisasi penting untuk sebuah sistem ERP, setidaknya kustomisasi diperlukan untuk menyesuaikan sistem dengan kebutuhan hampir semua perusahaan.
2.	Ganesh, Amal Shanil, K. N. Sunitha, C. Midhundas, A. M.	OpenERP/Odoo - An Open Source Concept to ERP Solution.	2016	Pada awalnya sistem ERP terbatas hanya ke perusahaan besar, namun dengan hadirnya teknologi open source, banyak sistem ERP open source muncul di pasar dan menjadi sistem ERP yang unggul.
3.	Zuana, Kiky Rizky Sidharta, Iwan	Sistem Informasi Pemotongan PPh 21 Atas Gaji Karyawan PT. Rajawali Tehnik.	2014	Aplikasi yang diimplementasi mempercepat kinerja karyawan dalam penyimpanan data gaji, kalkulasi gaji, kalkulasi PPh 21 serta pembuatan laporan pajak.
4.	Alfiandanu, Wisnu Agusta Siswanto, Eko	Sistem Informasi Pengolahan Data Gaji Dan Perhitungan PPh Pasal 21 Pada CV. Sinar Jasa Teknik Ngaliyan Semarang.	2015	Sistem Informasi yang diimplementasi telah memudahkan proses pengolahan data gaji karyawan serta perhitungan PPh 21.
5.	Nurjayanti, Leni Susanto, Ajib	Rekayasa Sistem Penghitungan Pajak Penghasilan (PPh) Pasal 21 dengan Metode Gross-up pada Pegawai Kantor Kecamatan Todanan Kabupaten	2015	Aplikasi yang telah dibuat memudahkan pihak administrasi perusahaan dalam kalkulasi PPh 21 berdasarkan data gaji karyawan.

		Blora menggunakan Prototyping		

Kumpulan penelitian yang tertera pada tabel di atas merupakan inspirasi bagi penulis dalam melakukan Kerja Praktek ini. Pada penelitian lainnya (Uppström et al., 2015), dijelaskan pentingnya implementasi serta kustomisasi sebuah sistem ERP pada sebuah perusahaan untuk memenuhi berbagai kebutuhan spesifik, serta pemanfaatan ERP open source "Odoo" dalam sebuah perusahaan mengurangi biaya dan membuka peluang untuk kustomisasi yang jauh lebih fleksibel dibandingkan vendor sistem ERP yang lainnya dalam penelitian oleh Ganesh et al. (2016). Penelitian oleh Zuana & Sidharta (2014), Endaryati et al. (2015) dan Nurjayanti & Susanto (2015) memaparkan keunggulan dalam implementasi sebuah sistem yang didesain khusus untuk kalkulasi gaji serta PPh 21 untuk perusahaan, di mana terintegrasinya *database* yang mengandung data gaji dari karyawan memungkinkan kalkulasi akurat yang bersifat otomatis , serta pembuatan laporan berdasarkan hasil kalkulasi tersebut menghasilkan laporan yang rapi dan akurat. Oleh karena itu, penulis akan mengembangkan sistem pelaporan pajak yang terintegrasi dengan sistem ERP perusahaan sebagai bentuk kustomisasi ERP, seperti yang diuraikan dalam penelitian "New Implications for Customization of ERP Systems" (Uppström et al., 2015), serta memiliki fitur dasar yaitu kalkulasi PPh 21 dan pembuatan laporan pajak, seperti sistem yang dikembangkan oleh Zuana & Sidharta (2014).

2.2 Landasan Teori

Dalam perancangan dan implementasi sistem pelaporan pajak, penulis membuat landasan teori. Landasan teori yang akan diuraikan merupakan teori-teori yang diperlukan dalam pembuatan sistem tersebut serta digunakan untuk memperkuat teori penelitian. Berikut adalah teori-teori yang digunakan:

2.2.1 Sistem Informasi

Tidak ada istilah atau definisi yang pasti untuk apa yang disebut sebagai "sistem informasi". Sebagai sebuah sistem yang berada di hampir seluruh perusahaan, organisasi dan institusi di seluruh dunia ataupun di kehidupan sehari-hari seorang individu, sistem informasi menyediakan berbagai fungsi dalam mendukung berbagai jenis aktivitas yang berbeda-beda. Dalam istilah yang paling sederhana, sistem informasi dibentuk dari informasi, dan untuk setiap kasus pemakaian yang berbeda, informasi yang dibutuhkan serta cara memproses dan menyajikan informasi tersebut juga berbeda, yang mengarah kepada terbentuknya berbagai sistem informasi yang dikembangkan berdasarkan kebutuhan yang beragam serta terbentuknya konsep yang beragam mengenai sistem informasi (Bajdor & Grabara, 2014).

Menurut Power dan Hadidi (2018) dalam jurnal yang berjudul "Journal of the Midwest Association for Information Systems", dari segi teknis sistem informasi dapat didefinisikan sebagai rangkaian komponen-komponen yang saling terintegrasi antara satu dengan yang lain, dengan fungsi mengumpulkan, menyimpan ataupun memproses data dalam tujuan mendukung keputusan, menyediakan informasi, pengetahuan ataupun produk digital. Data merupakan

pendorong utama dari sistem informasi, serta didukung oleh berbagai teknologi komputasi serta komunikasi.

Penggunaan sistem informasi dalam sebuah perusahaan ataupun organisasi merupakan salah satu kunci kesuksesan, dan semakin menjadi kebutuhan dasar berbagai perusahaan di dalam dunia bisnis sekarang ini yang telah berorientasi kepada e-business dan m-commerce, seperti yang telah diuraikan dalam jurnal "International Journal for Research in Business, Management and Accounting" oleh Bertozzi, Ali dan Gul (2017). Jurnal tersebut juga menjelaskan sebuah sistem informasi dalam perspektif yang lebih luas, di mana sistem informasi merupakan infrastruktur dalam sebuah organisasi, di mana teknologi informasi serta sumber daya manusia bekerja sama dalam menghasilkan sebuah *output* yang berguna. Ada tiga peran mendasar sistem informasi dalam sebuah perusahaan, yaitu *automation, information* dan *transformation.* Peran sistem informasi dalam *automation* merupakan bentuk substitusi berbagai aktivitas perusahaan yang sebelumnya dilakukan oleh manusia. Aktivitas ataupun proses tersebut dijalankan oleh sistem secara otomatis. Dalam peran *information*, sistem informasi menyediakan informasi mengenai suatu aktivitas atau proses bisnis yang relevan untuk mendukung pekerjaan manusia. Dan terakhir, *transformation* atau transformasi dilakukan pada sebuah aktivitas atau proses dengan terintegrasinya sistem informasi, mengubah berbagai prosedur mengenai aktivitas tersebut.

Menurut Jailia, Kumar, Agarwal dan Sinha (2017), proses pengembangan sebuah *software* didasari oleh sebuah pola desain yang telah ditetapkan. Salah satu pola desain yang umum dipakai untuk membentuk sebuah aplikasi berbasis web adalah arsitektur MVC (Model View Controller). Arsitektur MVC adalah pola

desain *software* yang memisahkan *business logic* dari *user interface*. MVC memiliki 3 komponen utama yaitu:

1. Model: mengandung data dari aplikasi, *business logic*, serta spesifikasi fungsional.
2. View: menangani berbagai metode pemaparan data yang diperlukan kepada *user* tanpa memaparkan data internal yang bersifat sensitif.
3. Controller: merupakan penghubung *user* dengan sistem, di mana aliran data dikontrol serta diarahkan ke model dan meng-*update* View setiap kali ada perubahan data.

Pengembangan sebuah sistem memerlukan prosedur yang tetap dalam proses dari awal hingga akhir. Umumnya, prosedur tersebut menggunakan SDLC (*Software Development Life Cycle*). SDLC merupakan dasar panduan dari seluruh proses perancangan serta semua aktivitas yang diperlukan dalam pengembangan sebuah perangkat lunak. Karena kebutuhan setiap sistem informasi berbeda dan bervariasi, maka ada lebih dari satu variasi dari model SDLC yang digunakan untuk setiap jenis kebutuhan yang berbeda (Lemke, 2018).

Menurut Alshamrani dan Bahattab (2015), terdapat 3 model SDLC yang umum: yaitu *Waterfall Model*, *Spiral Model* dan *Incremental/Iterative Model*. Untuk Kerja Praktek ini, *Waterfall Model* dipilih sebagai dasar perancangan proyek. *Waterfall Model* merupakan *model* SDLC yang pertama, dan yang paling dikenal oleh semua orang. Penggunaan ekstensif dari *Waterfall Model* dilakukan oleh berbagai pemerintah, dan perusahaan berskala besar. Struktur dari model ini merupakan 5 tahapan yang merupakan: *Requirement*, *Design*, *Coding*, *Testing & Implementation* dan *Maintenance*.

Dalam proses pengembangan sebuah *software*, seorang *programmer* tentunya memerlukan serangkaian *tools* untuk mempermudah proses *development* serta manajemen *source code*. Serangkaian *tools* tersebut umumnya ditemukan dalam satu buah *software* yang bernama *Integrated Development Environment* (IDE). IDE didefinisikan sebagai sebuah *software* yang digunakan oleh *programmer* untuk keperluan *editing*, *build* ataupun *debugging source code*. Fitur-fitur yang terdapat dalam sebuah IDE dapat ditambah, melalui *plugin* yang dapat di-*install* ke dalam IDE tersebut, di mana *plugin* yang dikhususkan untuk IDE tersebut menambah fungsi yang spesifik. Berbagai *plugin* terdapat pada sebuah *plugin repository* (Frier, 2017).

Di zaman ini, *software* yang bersifat *open source* banyak ditemukan di seluruh dunia. Menurut Open Source Organization, *Open Source Software* atau OSS didefinisikan sebagai sebuah *software* yang dapat dipakai, dimodifikasi *source code*-nya serta dibagikan ke siapa pun, secara bebas (Sang, Xu, & De Vrieze, 2016). Ada juga definisi dari OSS menurut Open Source Initiative (OSI), di mana OSS berarti *software* yang dapat dibagikan secara bebas/gratis ke siapa pun, dan dapat diakses oleh siapa pun, sehingga disebut sebagai Free and Open Source Software (FOSS). FOSS dikatakan gratis karena *source code* dari *software* dipublikasikan di *internet* (Vijaya, Chander, & Raju, 2017). Kebebasan memodifikasi serta mendistribusi *source code* dari sebuah OSS tergantung pada lisensi *open source* dari *software* tersebut (Garmabaki, Barabadi, Yuan, Lu, & Ayele, 2016). Karena dikembangkan dengan cara yang berbeda dibandingkan dengan *closed source software*, OSS memiliki sifat-sifat sebagai berikut (Malhotra, Pritam, Nagpal, & Upmanyu, 2014):

1. Jauh lebih murah, karena umumnya gratis dan tersedia langsung dari internet.
2. Lebih aman karena perubahan apapun pada *repository* dari OSS terpampang ke publik.
3. Lebih berkualitas karena kontribusi dari berbagai *developer*. Siapapun dapat ikut serta dalam pengembangan OSS, berkolaborasi bersama para *developer* dari seluruh dunia.
4. Tidak akan mengalami permasalahan *copyright*.
5. Pengguna atau *user* dari OSS dapat dikatakan sebagai *co-developer*.

2.2.2 Programming Language

Programming language (bahasa pemrograman) adalah bahasa yang dibuat untuk memerintahkan instruksi kepada sebuah mesin, khususnya sebuah komputer, untuk melakukan komputasi dan/atau mengendalikan perangkat keras yang terhubung dengan mesin tersebut. Bentuk dari sebuah *programming language* adalah sintaks, yang umumnya tersusun atas berbagai teks seperti huruf, kata, angka dan tanda baca, serupa dengan bahasa alami (Sethi, 2014).

Menurut tingkat abstraksi pada instruksi mesin, *programming language* dibagi menjadi dua kategori, yaitu *low-level programming language* dan *high-level programming language*. *Low-level programming language* merupakan bahasa pemrograman yang dapat langsung dimengerti oleh sebuah mesin, karena sedikit atau tidak adanya abstraksi, dan struktur bahasanya mendekati instruksi mesin murni. Beberapa contoh dari *low-level programming language* adalah *machine code* dan *assembly*. *High-level programming language* merupakan bahasa

pemrograman yang mendekati bahasa manusia, dan memiliki tingkat abstraksi yang tinggi sehingga mudah dipakai untuk pengembangan aplikasi modern. Beberapa contoh dari *high-level programming language* adalah C, C++, C#, Python, Pascal, FORTRAN dan LISP (Trivedi, 2017).

Berdasarkan cara berjalannya sebuah *programming language*, terdapat dua jenis, yaitu *compiled language* dan *interpreted language* (*interpreted* atau disebut juga *scripting*). Sebuah *compiled language* adalah bahasa pemrograman yang dirangkum ke dalam bentuk *file executable* oleh sebuah *compiler*. Beberapa *programming language* yang metode eksekusinya adalah melalui *compiler* termasuk: C, C++, D, Fortran, Objective-C, Java, C#, Kotlin, Scala, F# dan Rust. Sementara *interpreted language* adalah bahasa pemrograman yang di mana eksekusi dari aplikasi merupakan menjalankan setiap barisan kode pada aplikasi oleh sebuah *interpreter*, tanpa merangkum seluruh kode menjadi sebuah *file executable* atau *bytecode*. Contoh dari *programming language* bertipe *interpreted* adalah Python, Javascript, PHP, Lisp, Haskell, Julia dan Ruby (Aruoba & Fernández-Villaverde, 2015).

2.2.3 Database

Sebuah *database* didefinisikan sebagai kumpulan data yang terorganisir dalam bentuk *model* yang menyerupai realitas, di mana data tersebut dapat diakses, ditambah, dikurangi ataupun diubah dengan mudah untuk berbagai aktivitas ataupun proses yang membutuhkan informasi. Umumnya *database* digunakan untuk menyimpan berbagai data dari sebuah perusahaan seperti data transaksi pelanggan, data finansial dan data karyawan, ataupun digunakan untuk menyimpan

data sebuah institusi pendidikan seperti data mahasiswa, data perpustakaan sebuah universitas, serta data registrasi sebuah mata pelajaran. (Reddy, Prakash, & Sharma, 2017).

Sebuah *database* tersusun dari satu atau lebih *table* yang dapat terhubung satu dengan yang lain. *Table* terdiri dari baris dan kolom seperti tabel pada Excel (Manning, 2015), dan merupakan representasi dari sebuah model data, di mana setiap instansi dari model tersebut/obyek tersimpan dalam satu baris pada *table* (disebut sebagai *record*) . Kolom dari sebuah *table* merepresentasikan atribut dari sebuah model. Sebagai identitas unik pada setiap *record*, sebuah *primary key* merupakan satu atau lebih atribut yang membedakan setiap *record* dari yang lainnya. Hubungan antara jenis model/entitas yang berbeda didefinisikan sebagai relasi. Terdapat 3 jenis derajat pada sebuah relasi (Letkowski, 2014):

1. One-to-one: Satu *record* yang terhubung dengan satu *record* pada *table* yang lain
2. One-to-many: Satu *record* pada sebuah *table* terhubung ke satu atau lebih *record* pada *table* yang lain.
3. Many-to-many: Satu atau lebih *record* pada sebuah *table* dapat terhubung ke satu atau lebih *record* pada *table* yang lain.

Dalam mengakses sebuah *database*, dibutuhkan sebuah bahasa *query* yang disebut sebagai SQL. SQL bukan merupakan sebuah bahasa yang didedikasikan untuk pemrograman. Karena, secara esensi SQL tidak dikhususkan dalam pengembangan sebuah aplikasi, melainkan ditujukan untuk keperluan mengakses atau memanipulasi data serta metadata dari sebuah *database*, dan dapat diintegrasikan dengan aplikasi yang dikembangkan menggunakan bahasa

pemrograman sejati seperti C# atau Python, agar aplikasi tersebut dapat berinteraksi dengan sebuah *database*. SQL menjadi sebuah standar bagi berbagai DBMS yang ada di seluruh dunia.

Sebagai bahasa *query*, SQL memiliki sintaks yang jelas, dan mirip dengan bahasa inggris serta memiliki susunan *statement* yang natural. Berbagai kata kunci atau *keyword* dari SQL merupakan kata bahasa inggris yang umum seperti SELECT, INSERT, FROM dan WHERE. Struktur dari SQL terdiri dari 4 bagian instruksi, yaitu:

1. DML (Data Manipulation Language) yang dipakai untuk keperluan mengakses data, dan merupakan klasifikasi dari seluruh instruksi yang melakukan operasi membaca data (*read*) atau menulis data (*write*).
2. DDL (Data Definition Language) sebagai bentuk mengakses atau memanipulasi metadata seperti pembuatan tabel baru.
3. DCL (Data Control Language) menangani berbagai hal mengenai hak akses pengguna pada sebuah *database*. Hak akses seorang pengguna terhadap sebuah aksi pada sebuah *object* dalam *database* bisa diberikan, ataupun dicabut.
4. TCL (Transaction Control Language) memberikan para *developer* kemampuan untuk mengontrol transaksi data di dalam logika *database*.

Untuk mengatur sebuah *database*, DBMS adalah sebuah aplikasi komputer yang menyediakan sebuah *interface* di antara pengguna dan *database,* digunakan untuk keperluan manipulasi data seperti mengakses, menambah, mengurangi, mengubah ataupun mengurutkan data, serta untuk berbagai keperluan

administrasi *database* yang mencakup pengaturan hak akses sebuah *user*, *backup* data, pembuatan *database* baru ataupun penghapusan sebuah *database*. (Ahmed, Ahamed, Rafiq, & Rahim, 2017).

Berbagai vendor DBMS seperti Microsoft dan Oracle, telah mengembangkan bahasa pemrograman *database* mereka tersendiri. Microsoft dengan Transact-SQL untuk *database* SQL Server mereka, dan PL/SQL yang dikembangkan oleh Oracle untuk *database* Oracle. Keuntungan dari SQL adalah status bahasa SQL sebagai standar yang harus dipatuhi berbagai bahasa pemrograman *database*. Sintaks dari PL/SQL ataupun T-SQL tidak jauh berbeda dari SQL murni, dan perbedaan sintaks yang ada sangat minimal, dan *developer* tidak akan mengalami kesulitan mempelajari sebuah bahasa pemrograman *database* yang berbeda (Ardeleanu, 2016).

2.2.4 Version Control System (VCS)

Version Control System (VCS) merupakan *software* yang digunakan oleh *developer* untuk keperluan manajemen *source code*, *backup source code*, dan kolaborasi dengan sesama *developer* dalam pengembangan sebuah proyek. (Muşlu, Bird, Nagappan, & Czerwonka, 2014). Dalam pengembangan sebuah *software*, lebih dari satu *developer* harus bekerja sama dalam pembentukan sebuah proyek yang optimal, sehingga kebutuhan sebuah sistem VCS untuk me-*manage source code* sebuah proyek sangat diperlukan, karena VCS menyediakan sebuah *platform* yang bersifat kolaboratif, sehingga memudahkan para *developer* bekerja sama dengan efektif.

Dalam proses pengembangan *software*, tidak jarang bagi seorang *developer* untuk melakukan perubahan pada sebuah fitur dalam *software* tersebut. Sejumlah revisi pada *software* tersebut dilakukan oleh *developer* sebelum mencapai versi yang final. Tanpa sebuah sistem VCS, *developer* harus menyimpan setiap revisi dari *software* tersebut dalam kumpulan *file* yang berbeda. Proses ini sangat beresiko, dikarenakan kemungkinan dari kumpulan file tersebut dapat terhapus dengan tidak sengaja, atau sebuah hasil revisi yang terbaru dapat tertimpa oleh revisi yang lama karena men-*copy* kode yang salah. Dengan adanya sebuah VCS, setiap revisi dari *software* serta setiap perubahan dari kode yang dilakukan oleh *developer* tercatat secara otomatis dalam sebuah *repository,* yang merupakan gabungan seluruh versi berbeda *source code* dan *metadata*. (Zolkifli, Ngah, & Deraman, 2018).

Dalam cara sebuah VCS mengatur *source code*, perubahan *source code* serta setiap revisi dari *source code* tersebut, VCS tergolong ke dalam dua kategori (Rao & Sekharaiah, 2016):

1. Centralized Version Control System (CVCS)

   Sebuah sistem VCS yang tersentralisasi (CVCS) menyerupai arsitektur *client-server*, di mana *repository* dan daftar perubahan *source code* tersimpan dalam sebuah *server* pusat, dan dapat diakses menggunakan jaringan internet LAN ataupun WAN. Setiap tahap dari perubahan disebut sebagai *revision* (revisi), dan *revision* yang terbaru disebut sebagai *head*.
2. Distributed Version Control System (DVCS)

   Sistem DVCS memiliki persamaan dengan CVCS mengenai sistem *revision* dan *head*, namun terdapat perbedaan yang signifikan dengan CVCS, dimana

repository tidak disimpan dalam sebuah server pusat, melainkan pada direktori lokal setiap komputer *developer* yang terlibat dalam pengembangan *software* yang sama. *Repository* lokal milik *seorang* developer dapat disinkronisasi dengan *developer* yang lain.

2.2.5 PPh Pasal 21

Menurut Hasibuan (2019) pada skripsi yang berjudul "Analisis Penerapan dan Perbandingan Perhitungan PPh 21 atas Tunjangan berdasarkan UU No. 36 Tahun 2008 pada PT. Antar Lintas Sumatera Medan", Pajak Penghasilan Pasal 21 (PPh Pasal 21) merupakan ketentuan wajib pajak orang pribadi, termasuk bentuk usaha tetap yang diwajibkan untuk melakukan pemotongan pajak atas penghasilan sehubungan dengan pekerjaan jasa, dan kegiatan orang pribadi.

2.2.6 PPh Pasal 23

Pada jurnal bisnis dan akutansi yang dibuat oleh Setiadi dan Akhadi (2017), Pajak Penghasilan Pasal 23 (PPh Pasal 23) merupakan pajak yang dikenakan atas modal, penyerahan jasa, atau hadiah dan penghargaan, selain yang telah dipotong PPh Pasal 21. Umumnya penghasilan jenis ini terjadi saat adanya transaksi antara dua belah pihak, dimana pihak yang menerima penghasilan atau penjual atau pemberi jasa akan dikenakan PPh pasal 23.

2.2.7 PPh Pasal 4 ayat 2

Pada skripsi yang berjudul "Pengaruh Perubahan Tarif Pajak Penghasilan (PP No 46 Tahun 2013) terhadap Profit Margin pada Koperasi Binaan Bumitama

Agri Ltd." oleh Chaerullah (2015), Pajak Penghasilan Pasal 4 ayat 2 (PPh Pasal 4 ayat 2) merupakan pajak yang bersifat final, di mana pelunasan pajak tersebut menyelesaikan kewajiban pajak tersebut, dan tidak digabungkan dengan jenis penghasilan lain yang terkena pajak penghasilan yang bersifat tidak final.

2.3 Sistem yang akan digunakan

Dalam pembuatan serta implementasi sistem baru untuk Kerja Praktek ini, berikut adalah *software*, *tools* ataupun *framework* yang akan digunakan dalam pengembangan sistem pelaporan pajak.

2.3.1 Python

Python adalah bahasa pemrograman yang bersifat *dynamically-typed* serta memiliki sintaks yang jelas dan ekspresif, dan dapat dipakai dalam berbagai ragam aplikasi, mulai dari sebuah *script*, sampai *program* berskala besar (Blomqvist, Dulak, Friis, & Hargus, 2017). Desain dari bahasa pemrograman Python merupakan pendekatan yang *simple* namun efektif terhadap paradigma *Object-oriented Programming* (OOP), dan efektif dalam pengembangan aplikasi dengan cepat. *Interpreter* dari Python serta berbagai ragam *library* tersedia gratis dari beragam sumber untuk beragam sistem operasi. Python mencakup berbagai *library* yang dikhususkan untuk berbagai keperluan ilmiah seperti kalkulasi matematis, visualisasi vektor tiga dimensi, pemrosesan data gambar serta kecerdasan buatan, di mana ragam *library* tersebut menarik perhatian dari para ilmuwan dalam riset ilmiah ataupun analisis data berskala besar seperti yang terdapat pada sebuah bisnis (Campos-Rozo & Domínguez, 2016).

Keunggulan dari sebuah bahasa pemrograman yang bersifat *interpreted* membuat Python dapat di-diagnosa secara fleksibel dan cepat; nilai dari setiap variabel yang terdapat pada *source code* jika dijalankan pada sebuah IDE (Short, Bayer, & Burns, 2018). Sebagai bahasa yang bersifat *open source*, implementasi dari berbagai aplikasi Python dapat ditemukan pada mayoritas *platform* komputer yang termasuk *Windows, Linux,* dan *Mac OS* (Zhang, Moynihan, Ernest, & Gutenson, 2017).

2.3.2 eXtensible Markup Language (XML)

XML merupakan bahasa yang digunakan untuk mendefinisikan arsitektur data yang menghubungkan data dengan metadata. Arsitektur dari bahasa XML merupakan jaringan *node* yang bersifat hierarkial. Setiap *node* pada XML terhubung dengan *node* yang lain sebagai *child*, atau *parent*, membentuk struktur seperti pohon. Sintaks dari XML berfungsi untuk mendefinisikan data ataupun dokumen, dan mirip dengan bahasa HTML (*HyperText Markup Language*).

Jenis dari *node* sebuah XML merupakan "*element*". Element dari XML memiliki dua bentuk sebagai berikut:

1. Sepasang *tag*, mengapit *element* dengan *tag* pembuka dan penutup
   Contoh: <field name="model">account.invoice</field>
2. *Tag* tunggal, di mana *element* kosong, hanya ada *attribute*.
   Contoh: <field name="price" required="True"/>

Karena XML merupakan struktur di mana setiap *node* saling terhubung satu dengan yang lainnya, beberapa *tool query* seperti XPath dan XQuery dikembangkan untuk keperluan ekstraksi *element* pada XML. XPath merupakan

sebuah metode untuk meng-*query* dokumen XML, di mana *element* yang ingin diambil dari dokumen XML didefinisikan sebagai kriteria pencarian berdasarkan value yang terdapat pada *attribute* di dalam nama *element* yang dicari. Contoh dari sintaks XPath adalah: //field[@name=”price”]. Sintaks tersebut mendefinisikan pencarian terhadap seluruh *element* yang bernama “field” pada dokumen XML, yang mempunyai *attribute* bernama “name” dengan *value* yang merupakan “price” (Rühlemann & Gee, 2017).

2.3.3 PostgreSQL

PostgreSQL adalah *object-relational database management system* (ORDBMS) yang bersifat *open source*, mendukung standar dari SQL serta memiliki berbagai fitur *database* modern, seperti *query* kompleks, *trigger*, *updatable view*, serta *auto-vacuum* (He, Zheng, Deng, & Pan, 2017).

Dikembangkan selama lebih dari 17 tahun, PostgreSQL dapat berjalan pada berbagai macam sistem operasi seperti Windows, Linux serta beberapa sistem operasi dari keluarga besar UNIX (Mac OS, Solaris, BSD dan AIX). PostgreSQL merupakan ORDBMS yang ACID *compliant* serta dapat mendukung berbagai macam tipe data seperti *integer*, *numeric*, *char*, *boolean*, *date*, serta data biner atau disebut *binary lage objects* (BLOB) seperti data gambar, suara maupun video. Sebagai *database* kelas *enterprise*, PostgreSQL mendukung fitur-fitur modern seperti *asynchronous replication*, *character encoding Unicode*, dan sangat *scalable* (Sinaga et al., 2016).

2.3.4 Odoo Framework

Odoo adalah sistem ERP *open source* berbasis web, yang memiliki berbagai aplikasi bisnis dalam bentuk *module* untuk perusahaan berbagai skala. Lebih dari 200 *module* resmi terdapat dalam sistem ERP Odoo, tidak termasuk 4000 *module* kustom yang dikembangkan oleh komunitas Odoo (Murillo & Arévalo, 2015).

Odoo awalnya dikenal dengan nama "OpenERP". Odoo menggunakan bahasa pemrograman Python serta *database* PostgreSQL, dan memiliki 3 bagian utama yaitu *database*, *client* dan *server*. Ketiga bagian tersebut berkorespondensi dengan arsitektur MVC (Model-View-Controller) dan menggunakan *interface* XML-RPC sebagai bentuk komunikasi antara *client* dengan *server*. Karena Odoo merupakan aplikasi yang bersifat *open-source*, fitur-fitur kustom dapat dikembangkan dan diintegrasi ke dalam sistem utama Odoo dengan mudah dalam bentuk *module* kustom yang dikembangkan sesuai kebutuhan perusahaan (Ganesh et al., 2016).

2.3.5 Linux & Ubuntu

Linux merupakan OS yang awalnya diciptakan oleh Linus Torvalds, dan kemudian dikembangkan oleh ribuan kontributor yang lain. Pada umumnya bentuk OS Linux terdapat pada distribusi yang disebut "distro", tersedia untuk komputer desktop, ataupun server (Yu et al., 2016).

Dirancang berdasarkan prinsip-prinsip kernel UNIX, Linux merupakan OS yang *free and open source.* Walaupun memiliki kernel yang original dan berbeda dengan UNIX, interface terhadap pengguna ataupun *programmer*

compatible dengan sebuah sistem UNIX, sehingga berbagai aplikasi yang berbasis UNIX dapat berjalan pada Linux. Design dari Linux modular, dan dapat dikatakan sangat baik jika digunakan sebagai sistem sebuah server. Linux mendukung *environment* untuk lebih dari satu pengguna (*multi-user*), *time-sharing scheduler* untuk melindungi berbagai proses ataupun *service* yang sedang berjalan serta *memory management* dengan metode *copy-on-writing* dan *page-sharing* (Sajid, Shah, Kamran, Javaid, & Zhang, 2016).

Walaupun sebuah sistem operasi Linux sering disebut sebagai "OS Linux", kata tersebut sebenarnya mengarah kepada sebuah distribusi Linux, bukan merujuk ke sebuah sistem operasi tunggal yang bernama "Linux", karena "Linux" pada dasarnya bukan merupakan sebuah sistem operasi, melainkan sebuah *kernel*, yaitu bagian utama dari sebuah sistem operasi. Distribusi Linux (umum disebut sebagai "distro") adalah kumpulan dari berbagai komponen penting, yang umumnya berupa berbagai *tools*, *package management system* serta sebuah *desktop environment* (jika *distro* tersebut merupakan sistem dengan GUI) dan dokumentasi lengkap, yang dikemas beserta kernel Linux, membentuk sebuah sistem operasi yang lengkap.

Sebagai *software* yang *open source*, kemungkinan mengkustomisasi Linux menjadi OS sesuai preferensi tersendiri tidak terbatas, sehingga melahirkan lebih dari ratusan *distro* yang dapat di-*download* oleh siapapun di seluruh dunia. Siapapun dapat mengembangkan *distro* tersendir berdasarkan kernel Linux, asalakan memiliki ilmu serta motivasi yang cukup untuk pengembangan tersebut (Castro, 2016).

Ubuntu merupakan salah satu distro Linux yang paling populer, yang dikembangkan dari Debian oleh Mark Shuttleworth dan menjadi produk dari perusahaan Canonical, yang ditemukan oleh Shuttleworth. Berbagai versi Ubuntu dibuat untuk berbagai ragam *hardware*, mulai dari *desktop*, *server*, hingga IoT (Internet of Things). Ubuntu merupakan sistem operasi yang sangat stabil. Sebuah versi Ubuntu dapat di-*upgrade* selama beberapa tahun tanpa perlu melakukan instalasi manual versi yang lebih baru. Sebuah tipe rilis Ubuntu jangka panjang, atau disebut sebagai Ubuntu LTS (long-term support), merupakan versi Ubuntu yang jauh lebih stabil dan aman daripada yang versi non-LTS, dan umumnya dipakai untuk pengguna yang ingin instalasi sebuah sistem dengan lancar. Ubuntu non-LTS menerima dukungan dari Canonical selama 9 bulan, sementara versi LTS didukung selama 5 tahun (Castro, 2016).

2.3.6 Apache Subversion (SVN)

Apache Subversion (SVN) adalah *Centralized Version Control System* (CVCS) yang *open source*, berlisensi Apache dan dibuat oleh CollabNet Inc. pada tahun 2000. SVN mengatur versi dari berbagai *file*, direktori serta *metadata*. SVN dapat digunakan untuk memindahkan atau meng-*copy* seluruh direktori, dan tetap menjaga seluruh *history* dari berbagai *revision* yang berada di dalam direktori tersebut (Bhoir & Patil, 2018).

2.3.7 PyCharm

PyCharm merupakan IDE yang dikhususkan untuk pengembangan aplikasi dengan bahasa pemrograman Python. Dikembangkan oleh perusahaan

JetBrains yang berada di Ceko, PyCharm memiliki beberapa fitur utama sebagai berikut (S. D. Kumar, U., C., & Ganesh, 2018):

1. Navigasi proyek serta kode: Tersedia tampilan khusus untuk proyek, struktur direktori dan navigasi cepat ke sebuah *file*, *method*, *class* ataupun penggunaan *method* serta *class*.
2. Asistensi serta analisis kode: *Code auto-complete*, *syntax highligting*, label *error* pada kode, integrasi *linter* Python untuk pengecekan kode, serta perbaikan kode cepat.
3. Refactoring: Perubahan kode, referensi terhadap variabel, nama *class*, nama *method* pada seluruh *file* atau sebagian *file* tanpa mengubah perilaku/fungsi dari kode tersebut.
4. Dukungan untuk berbagai *web framework* berbasis Python seperti Django, web2py dan Flask
5. Debugger Python yang terintegrasi, untuk berbagai keperluan debugging kode.
6. Unit-testing terintegrasi, dengan dukungan pengecekan kode per baris.
7. Dukungan untuk pengembangan Google Apps berbasis Python.
8. Integrasi dari berbagai sistem VCS seperti Git, Apache Subversion, Mercurial dan Perforce, dalam betunk *interface* yang menampilkan daftar perubahaan pada kode.