UNIVERSITAS INDONESIA

JUDUL SKRIPSI/THESIS/DISERTASI

TUGAS AKHIR

TIDAK DIKETAHUI
XXXXXXXXXX

FAKULTAS ??
PROGRAM STUDI ??
DEPOK
JANUARI 2010

UNIVERSITAS INDONESIA

JUDUL SKRIPSI/THESIS/DISERTASI

TUGAS AKHIR
Diajukan sebagai salah satu syarat untuk memperoleh gelar
Sarjana ??

TIDAK DIKETAHUI
XXXXXXXXXX

FAKULTAS ??
PROGRAM STUDI ??
DEPOK
JANUARI 2010

HALAMAN PERSETUJUAN

Judul : Judul Skripsi/Thesis/Disertasi
Nama : Tidak Diketahui
NPM : XXXXXXXXXX

Laporan Tugas Akhir ini telah diperiksa dan disetujui.

XX Januari 2010

Prof. XXXX
Pembimbing Tugas Akhir

HALAMAN PERNYATAAN ORISINALITAS

Tugas Akhir ini adalah hasil karya saya sendiri,
dan semua sumber baik yang dikutip maupun dirujuk
telah saya nyatakan dengan benar.

Nama	:	**Tidak Diketahui**
NPM	:	**XXXXXXXXXX**
Tanda Tangan	:	
Tanggal	:	**XX Januari 2010**

HALAMAN PENGESAHAN

Tugas Akhir ini diajukan oleh	:	
Nama	:	Tidak Diketahui
NPM	:	XXXXXXXXXX
Program Studi	:	??
Judul Tugas Akhir	:	Judul Skripsi/Thesis/Disertasi

Telah berhasil dipertahankan di hadapan Dewan Penguji dan diterima sebagai bagian persyaratan yang diperlukan untuk memperoleh gelar Sarjana ?? pada Program Studi ??, Fakultas ??, Universitas Indonesia.

DEWAN PENGUJI

Pembimbing	:	Prof. XXXX	( )
Penguji	:	Prof. XXX	( )
Penguji	:	Prof. XXXX	( )
Penguji	:	Prof. XXXXXX	( )

@todo

Jangan lupa mengisi nama para penguji.

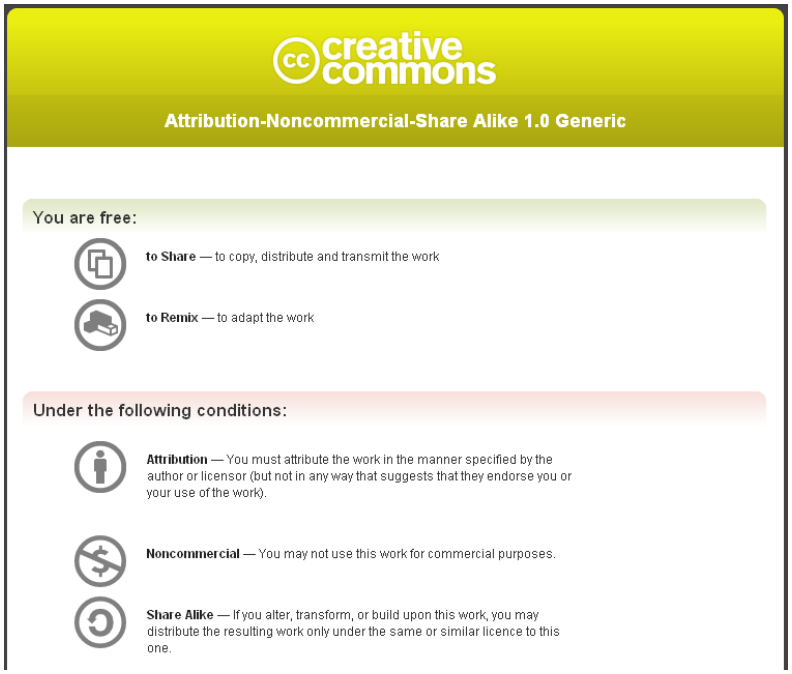
Ditetapkan di	:	Depok
Tanggal	:	XX Januari 2010

KATA PENGANTAR

Template ini disediakan untuk orang-orang yang berencana menggunakan LaTeX untuk membuat dokumen tugas akhirnya. Mengapa LaTeX? Ada banyak hal mengapa menggunakan LaTeX, diantaranya:

1. LaTeX membuat kita jadi lebih fokus terhadap isi dokumen, bukan tampilan atau halaman.
2. LaTeX memudahkan dalam penulisan persamaan matematis.
3. Adanya automatis dalam penomoran caption, bab, subbab, subsubbab, referensi, dan rumus.
4. Adanya automatisasi dalam pembuatan daftar isi, daftar gambar, dan daftar tabel.
5. Adanya kemudahan dalam memberikan referensi dalam tulisan dengan menggunakan label. Cara ini dapat meminimalkan kesalahan pemberian referensi.

Template ini bebas digunakan dan didistribusikan sesuai dengan aturan *Creative Common License 1.0 Generic*, yang secara sederhana berisi:



Gambar 1: *Creative Common License 1.0 Generic*

Gambar 1 diambil dari http://creativecommons.org/licenses/by-nc-sa/1.0/deed.en_CA. Jika ingin mengentahui lebih lengkap mengenai *Creative Common License 1.0 Generic*, silahkan buka http://creativecommons.org/licenses/by-nc-sa/1.0/legalcode. Seluruh dokumen yang dibuat dengan menggunakan template ini sepenuhnya menjadi hak milik pembuat dokumen dan bebas didistribusikan sesuai dengan keperluan masing-masing. Lisensi hanya berlaku jika ada orang yang membuat template baru dengan menggunakan template ini sebagai dasarnya.

Dokumen ini dibuat dengan LaTeX juga. Untuk meyakinkan Anda, coba lihat properti dari dokumen ini dan Anda akan menemukan bagian seperti Gambar 2. Dokumen ini dimaksudkan untuk memberikan gambaran kepada Anda seperti apa mudahnya menggunakan LaTeX dan juga memperlihatkan betapa bagus dokumen yang dihasilkan. Seluruh url yang Anda temukan dapat Anda klik. Seluruh referensi yang ada juga dapat diklik. Untuk mengerti template yang disediakan, Anda tetap harus membuka kode LaTeX dan bermain-main dengannya. Penjelasan dalam PDF ini masih bersifat gambaran dan tidak begitu mendetail, dapat dianggap sebagai pengantar singkat. Jika Anda merasa kesulitan dengan template ini, mungkin ada baiknya Anda belajar sedikit dasar-dasar LaTeX.

Gambar 2: Dokumen Dibuat dengan PDFLatex

Semoga template ini dapat membantu orang-orang yang ingin mencoba menggunakan LaTeX. Semoga template ini juga tidak berhenti disini dengan ada kontribusi dari para penggunanya. Kami juga ingin berterima kasih kepada Andreas Febrian, Lia Sadita, Fahrurrozi Rahman, Andre Tampubolon, dan Erik Dominikus atas kontribusinya dalam template ini.

Depok, 30 Desember 2009

Tidak Diketahui

HALAMAN PERNYATAAN PERSETUJUAN PUBLIKASI TUGAS AKHIR UNTUK KEPENTINGAN AKADEMIS

Sebagai sivitas akademik Universitas Indonesia, saya yang bertanda tangan di bawah ini:

Nama	:	Tidak Diketahui
NPM	:	XXXXXXXXXX
Program Studi	:	??
Fakultas	:	??
Jenis Karya	:	Tugas Akhir

demi pengembangan ilmu pengetahuan, menyetujui untuk memberikan kepada Universitas Indonesia **Hak Bebas Royalti Noneksklusif (Non-exclusive Royalty Free Right)** atas karya ilmiah saya yang berjudul:

Judul Skripsi/Thesis/Disertasi

beserta perangkat yang ada (jika diperlukan). Dengan Hak Bebas Royalti Noneksklusif ini Universitas Indonesia berhak menyimpan, mengalihmedia/formatkan, mengelola dalam bentuk pangkalan data (database), merawat, dan memublikasikan tugas akhir saya selama tetap mencantumkan nama saya sebagai penulis/pencipta dan sebagai pemilik Hak Cipta.

Demikian pernyatan ini saya buat dengan sebenarnya.

Dibuat di : Depok
Pada tanggal : XX Januari 2010

Yang menyatakan

(Tidak Diketahui)

ABSTRAK

Nama	:	Tidak Diketahui
Program Studi	:	??
Judul	:	Judul Skripsi/Thesis/Disertasi

@todo

Tuliskan abstrak laporan disini.

Kata Kunci:

@todo

Tuliskan kata kunci yang berhubungan dengan laporan disini

ABSTRACT

Name : Tidak Diketahui

Program : ??

Title : Unknown Title for Final Report/Thesis/Disertation

@todo

Write your abstract here.

Keywords:

@todo

Write up keywords about your report here.

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB 1
PENDAHULUAN

Bab ini menjelaskan latar belakang, permasalahan, tujuan dan ruang lingkup penelitian, serta sistematika penulisan tugas akhir penelitian

1.1 Latar Belakang

Perjalanan kehidupan manusia dipenuhi dengan berbagai kejadian. Manusia memerlukan usaha untuk dapat memenuhi kebutuhan dan menjaga eksistensi dari ancaman, baik itu dari dalam dan dari luar. Salah satu ancaman tersebut adalah penyakit. Sepanjang sejarah, penyakit tidak dapat terlepas dari manusia.Perkembangan ilmu dan daya pikir manusia menyebabkan manusia dapat bertahan dari serangan penyakit, namun penyakit tidak serta merta menghilang. Muncul jenis penyakit baru yang juga memerlukan pengetahuan baru dalam mencegah penyebaran penyakit tersebut.Berbagai pengobatan dan kepecayaan dalam menyembuhkan penyakit bermunculan. Dunia timur muncul dengan pengobatan herbal dan teknik akupuntur. Arkeolog menemukan bukti bahwa pemanfaatan tanaman obat herbal telah ada sejak 60000 tahun silam (masa*Paleolithic*). Tulisan berumur 5000 tahun tentang daftar tanaman obat juga ditemukan. Tulisan ini merupakan list herbal yang dibuat oleh penduduk Sumeria. Tidak hanya Sumaria, dalam sejarah Mesir, Yunani maupun Cina terdaftar sebagai negara yang mengembangkan pengobatan herbal. Cina, tidak hanya mengembangkan herbal, mereka juga mengembangkan teknik akupuntur dalam menyembuhkan penyakit.

Seiring dengan berkembang ilmu pengetahuan dan teknologi manusia khususnya *proteomics* dan *genomics*, manusia dapat mempelajari penyakit pada tingkat molekul. Hal ini memberikan pengetahuan dan kepastian dalam pengembangan obat secara modern. *Drug discovery* merupakan runtutan skenario panjang dalam menemukan calon obat yang berpotensi. Sebelum masuk dalam *drug discovery*, dilakukan *pre-drug discovery*. Dalam tahap ini peneliti mempelajari penyakit tersebut, karaktersitik, bagaimana penyakit tersebut mempengaruhi gen penderita,bagimana gen yang terkena tersebut memproduksi protein, bagaimana protein tersebut berinteraksi dengan lingkungan sekitar, bagaimana protein tersebut mempengaruhi lapisan dimana protein tersebut berperan , dan pada akhirnya bagaimana penyakit tersebut mempengaruhi *host*(tubuh yang terkena penyakit). De-

ngan berbekal pengetahuan dari proses *pre-drug discovery* tersebut, peneliti perlu memastikan bagian yang berupa molekul tunggal (gen / protein) yang berpengaruh dalam penyakit tersebut. Setelah hal tersebut, perlu dilakukan validasi dengan cara menyuntikan ”target” tersebut ke dalam sel hidup atau *speciment* percobaan. Hal ini untuk memastikan bahwa molekul target yang diambil adalah valid. Setelah molekul target valid, peneliti perlu mencari molekul yang dapat membalikan kerja molekul target. Diperlukan waktu yang cukup lama untuk mengidentifikasi molekul pembalik tersebut dan memastikan bahwa molekul tersebut dapat digunakan sebagai obat. Berdasarkan survey, waktu yang diperlukan dalam suatu *drug discovery* sekitar 10-15 tahun. Selain itu biaya yang dikeluarkan juga sangat besar (800 juta - 1 milyar dollar) dan kemungkinan gagal dalam menemukan obat baru yang berpotensi pun tetap ada.

Waktu yang dibutuhkan, biaya yang dikeluarkan, serta kemungkinan gagal dalam menemukan obat baru yang berpotensi menjadi perhatian utama dalam *drug discovery* mula mula. Komputer turut mengambil andil dalam *drug discovery*modern. Pemanfaatan komputer tersebut menggantikan proses uji coba pencarian molekul yang dapat membalikan efek dari molekul target yang sebelumnya dilakukan dengan eksperimen terhadap makhluk hidup digantikan dengan simulasi.*Virtual screening*, yaitu mencari molekul yang dapat mengikat (*binding*) dengan molekul target dari *database* molekul yang tersimpan. Untuk memastikan molekul terikat tersebut, *virtual screening* memanfaatkan perhitungan fungsi / *scoring* untuk menentukan mana molekul yang ”*best-fit*” dengan molekul target. Pada umumnya *virtual screening* tersebut berjalan secara otomatis dengan menggunakan program komputer. Dengan cara tersebut, para peneliti dapat memangkas waktu dan biaya yang dibutuhkan dalam *drug discovery*. Beberapa aplikasi *virtual screening* yang dapat digunakan antara lain : Auto Dock, DOCK , Gold, V Life MDS, Flex X kalo udah jelas, pasti perlu resource, jelasin tentang resource, kendala nya juga

Virtual screening yang mencakup evaluasi seluruh koleksi *database* molekul terhadap molekul target membutuhkan tenaga komputasi yang tidak sedikit. Tidak hanya itu, kapasitas memori penyimpanan untuk koleksi data molekul maupun hasil *virtual screening*, sumber daya listrik yang digunakan untuk menjaga komputer tetap bekerja. Kemampuan komputasi pada komputer terletak pada *processor*. Komputer akan mengenali *Virtual screening* yang dijalankan sebagai kumpulan instruksi yang harus dikerjakan oleh *processor*. Semakin cepat *processor*, semakin banyak instruksi yang dapat dikerjakan setiap detiknya. *Virtual screening* dapat dilakukan pada sebuah komputer yang memiliki koleksi data molekul yang besar, namun masih terdapat kendala, seberapa lama waktu yang dibutuhkan untuk mem-

proses semua koleksi data molekul tersebut. Untuk memenuhi kebutuhan tenaga komputasi tersebut, dibutuhkan infrastruktur komputer yang dapat saling bekerja sama untuk melakukan suatu komputasi. Infrastruktur ini dikenal sebagai *supercomputer*. *Supercomputer* terdiri dari beberapa *processor* yang memiliki kemampuan komputasi yang tinggi.Komputer tersebut dapat tersebar diberbagai tempat dan dihubungkan dengan jaringan atau kumpulan komputer yang diletakan saling berdekatan pada suatu tempat. Proses kerja *supercomputer* adalah *centralization* dimana setiap *processor* mengerjakan *task* yang sama dan hasilnya akan kembali diolah oleh suatu *processor* untuk menghasilkan output akhir. Konsumsi energi yang dibutuhkan oleh *supercomputer* tergolong sangat besar dan sebagian besar hasil dari eksekusi *task* adalah panas. Dibutuhkan biaya tambahan untuk pemeliharaan *supercomputer* agar dapat terus bekerja pada suhu yang terjaga.

Besarnya biaya yang diperlukan oleh *supercomputer* merupakan kendala utama. *Grid computing* memberikan alternatif lain untuk permasalahan tersebut. Teknologi ini memanfaatkan sekumpulan komputer biasa secara fisik yang saling terhubung melalui suatu jaringan sebagai suatu kesatuan komputer (*super virtual computer*). Tidak seperti *supecomputer*, setiap komputer bekerja sama untuk menyelesaikan sebuah *task*, yang kemudian *task* tersebut akan hilang jika selesai dan tergantikan dengan *task* yang baru. Masing masing komputer penyusun tersebut akan bekerja secara parallel. Permasalahan kembali muncul ketika pengadaan komputer secara fisik yang banyak dan pengaturan jaringan yang perlu dipahami sebelum digunakan untuk *virtual screening*

Terlepas dari pengadaan komputer secara fisik, teknologi *cloud computing* hadir sebagai solusi. Pengguna *cloud computing* tidak perlu memikirkan bagaimana biaya dan perawatan dari teknologi tersebut. Pada saat ini, pemanfaatan aplikasi bersifat *cloud*, dimana pengguna hanya disuguhkan dengan tampilan saja, sedangkan proses komputasi berada di ”awan”. Sifat virtualisasi yang menjadi andalan *cloud computing* membuat teknologi ini sangat murah untuk menambah komputer secara virtual. *Cloud computing* seolah olah mampu membuat suatu kumpulan komputer yang saling bekerja sama menyelesaikan *task* seperti *grid computing*. Memaksimalkan virtualisasi merupakan kunci dalam teknologi *cloud computing*

Teknologi virtualisasi yang ada saat ini terbentur dengan abstraksi komputer pada level *hardware*. Pembuatan komputer virtual tersebut membutuhkan *resource* memori penyimpanan dan kemampuan komputasi yang telah ditentukan sebelumnya. Dengan begitu jumlah komputer virtual akan terbatas dengan *resource* yang dimiliki oleh *host*. Disatu sisi, kinerja virtual komputer virtual tidak akan optimal ketika terdapat komputer virtual yang tidak melakukan apa apa, sehingga alokasi

resource komputer virtual tersebut dapat dibagi secara merata kepada komputer virtual lainnya. Docker, sebagai *platform* virtualisasi memberikan pandangan baru dalam mengoptimalkan proses virtualisasi tersebut.

1.2 Permasalahan

1.2.1 Definisi Permasalahan

Berdasarkan latar belakang yang telah dijelaskan, penulis berusaha untuk menganalisis apakah pemanfaatan teknologi *cloud computing*dengan menggunakan *platform* Docker dapat diajukan sebagai solusi alternatif dari *supercomputer* dan apakah performa yang diberikan tidak berbeda jauh dengan *grid computing*. Untuk itu, penulis akan mencoba *virtual screening* dengan menggunakan aplikasi Autodock dan Autodock Vina yang telah terinstall pada Docker.

1.2.2 Batasan Permasalahan

Pada penelitian ini, penulis lebih berfokus kepada performa Docker dalam menjalankan aplikasi Autodock dan Autodock Vina untuk *virtual screening*. Penulis tidak akan membahas segi *virtual screening* dikarenakan keterbatasan pengetahuan yang dimiliki.

1.3 Tujuan

Penelitian ini secara umum bertujuan untuk mengenalkan pemanfaatan *cloud computing* dengan *platform* Docker dalam virtualisasi komputer yang digabungkan dengan teknologi *cloud computing* untuk menghasilkan performa komputasi dengan harga terjangkau dibandingkan dengan pengadaan *supercomputer* maupun *grid computing*.

1.4 Posisi Penelitian

Penelitian ini mencari alternatif lain dari *supercomputer* dengan memanfaatkan teknologi *cloud computing*. Hasil dari penelitian ini akan dibandingkan dengan hasil penelitian serupa yang telah dikerjakan oleh Bapak Muhammad Hafizhuddin Hilman, S.Kom., M.Kom.

1.5 Sistematika Penulisan

Penulisan ini terbagi dalam 5 bab :

- BAB 1 PENDAHULUAN
  Bagian ini berisikan latar belakang, permasalahan, tujuan dan ruang lingkup penelitian, serta sistematika penulisan tugas akhir penelitian.
- BAB 2 LANDASAN TEORI
  Bagian ini berisikan penjelasan *drug discovery* dengan cara *virtual screening* memanfaatkan teknologi informasi secara umum. Selain itu, *cloud computing* dan Docker akan dijelaskan secara detail.
- BAB 3 METODOLOGI PENELITIAN
  Bagian ini berisikan detail tahap-tahap penelitian dan data yang digunakan.
- BAB 4 HASIL PENELITIAN DAN ANALISIS
  Bagian ini berisikan hasil dari penelitian yang telah dilaksanakan dan analisis dari hasil yang diperoleh.
- BAB 5 KESIMPULAN DAN SARAN
  Bagian ini berisikan kesimpulan penulis terkait dengan penelitian dan saran penulis dalam penelitian ke depannya.

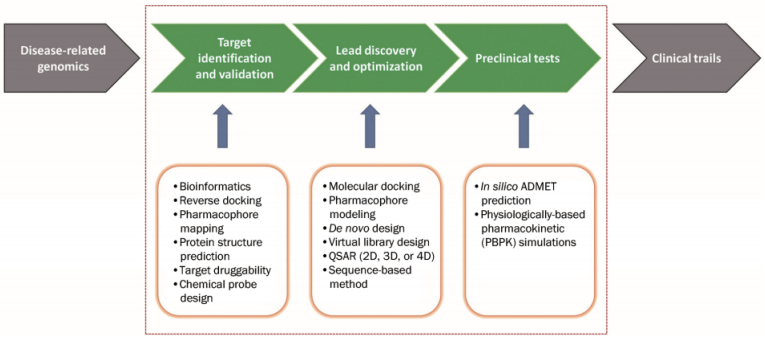
BAB 2
LANDASAN TEORI

Bab ini akan menjelaskan sekilas tentang *computational drug discovery*, apa itu *cloud computing* , dan Docker sebagai platform virtualisasi komputer.

2.1 *Computational Drug discovery*

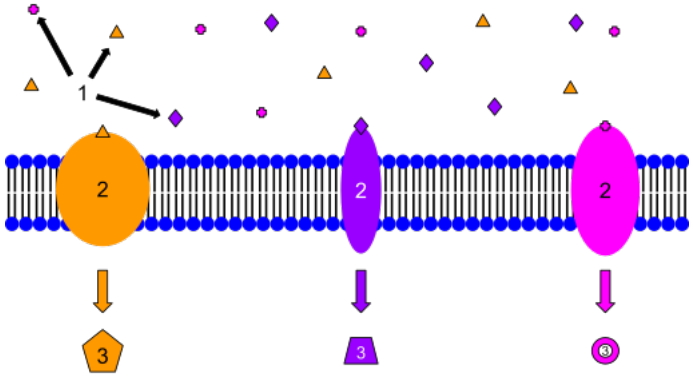
Drug discovery adalah suatu proses menemukan kandidat obat yang berpotensial. Proses ini melibatkan berbagai displin ilmu seperti biologi, kimia, maupun farmakologi. Dalam *drug discovery* dahulu kala, terkadang obat tersebut ditemukan secara tidak sengaja(*serendipity*) atau berdasarkan penyembuhan tradisional. Seiring dengan berkembangnya pengetahuan dan teknologi, khususnya dalam bidang kimia, ekstraksi senyawa kimia dari makhluk hidup merupakan hal yang wajar. Ekstrak tersebut akan disimpan dan akan disimpan sebagai dalam suatu *library* yang nantinya dapat digunakan kembali. Ekstrak tersebut dapat di-*screening*, apakah terdapat senyawa yang berpotensi sebagai penyembuh dari suatu *sympton* penyakit. Hal ini merupakan dasar farmakologi (*classical pharmacology*).

Waktu yang dibutuhkan dalam proses *drug discovery* cukup lama. Tidak hanya itu, proses ini juga sangat beresiko dan membutuhkan dana yang besar [1].Walaupun demikian, investasi dana dalam *drug discovery* juga berkembang pesat. Namun, investasi tersebut tidak dibarengi dengan hasil yang proporsional dengan investasi tersebut. Hal ini disebabkan oleh rendahnya efektivitas dan tingginya kemungkinan kegagalan dalam *drug discovery*. Beberapa pendekatan telah dilakukan dalam meningkatkan efektivitas, salah satu cara tersebut adalah CADD (*Computer Aided Drug Design*).Dengan begitu, penekanan biaya dan siklus waktu penemuan obat juga semakin cepat. Dalam CADD, komputer digunakan sebagai sarana komputasi dan penyimpanan data dalam *drug discovery*. Tidak ketinggalan, CADD menggunakan aplikasi dalam mendesain ikatan kimiawi, memodelkan struktur kimia yang mengandung kandidat calon obat yang berpotensial dan pembuatan *library* yang dapat digunakan untuk pembelajaran selanjutnya.



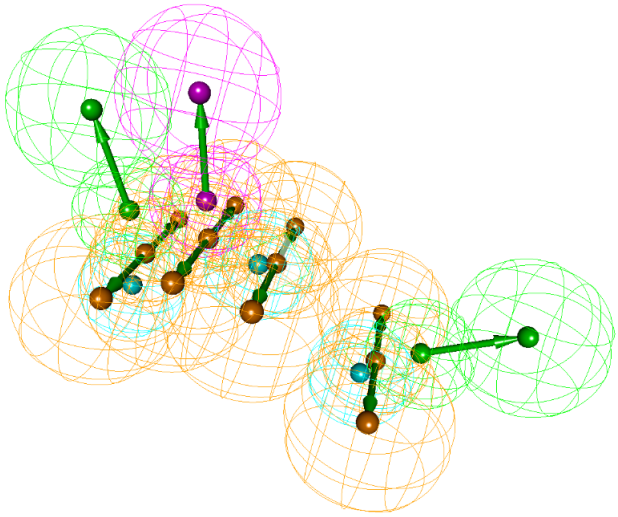
Gambar 2.1: Pemanfaatan komputer sebagai sarana komputasi dalam *drug discovery*

Dalam *computational drug discovery* (penemuan obat dengan memanfaatkan komputer), pendekatan yang dapat dilakukan dibagi menjadi *structure-based drug design*(SBDD), *ligand-based drug design*(LBDD) dan pendekatan berdasarkan sekuens. terdiri dari proses *docking* kandidat *ligand* ke target protein kemudian dilakukan penilaian dengan menggunakan *scoring function* untuk dapat menghitung kemungkinan *ligand*tersebut terikat pada target protein dengan daya tarik yang tinggi.



Gambar 2.2: Ilustrasi *ligand*(nomor 1)yang akan terikat dengan *receptor*(nomor 2)

Dalam LBDD, diberikan sekumpulan *ligand* dan *receptor*, dimana model *receptor*dapat dibuat dengan mengumpulkan informasi dari *ligand*. Model ini dikenal dengan *pharmacophore*. Kemudian, kandidat *ligand* akan dibandingkan dengan *pharmacophore*, apakah kompatibel dengan *pharmacophore* dan dapat terikat.

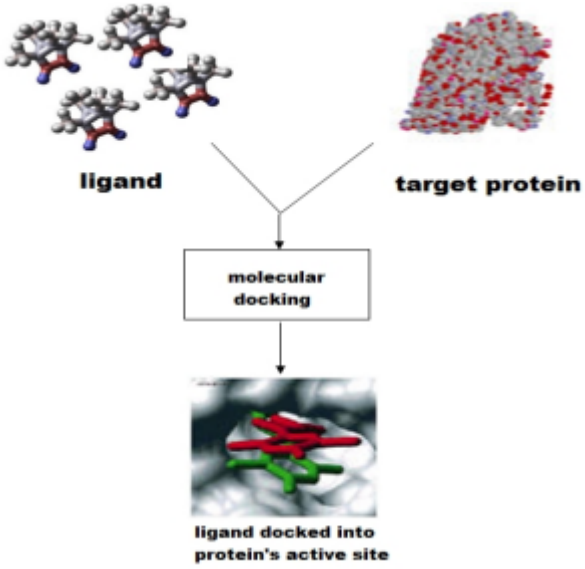


Gambar 2.3: Contoh dari *pharmacophore*

2.2 *Molecular Docking*

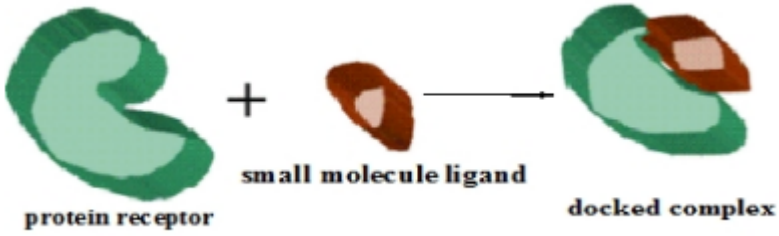
Virtual screening berdasarkan *molecular docking* merupakan metode yang sering digunakan dalam SBDD. Penulis juga akan menggunakan metode tersebut yang terdapat dalam Program Autodock dan Autodock Vina. Pemodelan struktur kimia menggunakan *molecular modeling* untuk mempelajari fenomena dari struktur kimia tersebut. Kemudahan dalam *molecular modeling* sekarang ini dibantu dengan aplikasi komputer yang tersedia, namun masih terdapat kesulitan yaitu bagaiman cara mendapatkan model struktur ikatan kimia yang benar dan interpretasi model yang tepat.Secara umum, *molecular modeling* dapat dikatakan sebagai pemanfaatan komputer dalam mengkonstruksi molekul dan melakukan berbagai macam perhitungan untuk mempelajari karakteristik dan sifat struktur ikatan kimia. Istilah *molecular modeling* terkadang disamakan dengan istilah *computational chemistry*.

Proses *drug discovery* yang semula berupa *trial and error* berubah menjadi proses yang dibantu dengan perhitungan komputer. Perhitungan tersebut digunakan dalam menentukan struktur ikatan kimia baru berdasarkan struktur protein yang sudah diketahui. Pendekatan ini terbagi menjadi dua : *de novo design* dan *docking*. *Docking* merupakan suatu proses untuk menebak struktur *complex* dari struktur *ligand* dan protein. Dalam *molecular modeling*, *docking* dapat dipandang sebagai suatu metode untuk memprediksi orientasi suatu molekul ketika diikat dengan molekul lainnya untuk membentuk *complex* yang stabil.



Gambar 2.4: *Proses molecular docking*

Molecular docking adalah suatu proses komputasi dalam pencarian *ligand* yang paling cocok baik secara geometri maupun energi ketika diikat dengan suatu *receptor*(protein) yang telah diketahui.Aspek utama dalam *molecular docking* adalah perhitungan energi interaksi dan konformasi dengan menggunakan metode dari kuantum mekanik hingga fungsi empiris energi. Permasalahan *molecular docking* sekilas dapat dilihat sebagai suatu permasalahan *lock and key*, dimana *receptor* protein dapat dipandang sebagai *lock* dan *ligand* sebagai *key*. Namun dalam praktiknya, *molecular docking* akan mencari *ligand* yang dapat menyesuaikan dengan *receptor*.karena adaptasi tersebut, *molecular docking* dapat dipandang sebagai permasalahan "*best-fit*" *ligand*. Diperlukan *scoring function* untuk dapat membatasi *best-fit* dari proses *molecular docking*. Fungsi tersebut biasanya berdasarkan *force field* yang digunakan dalam mensimulasikan protein. Beberapa *scoring function* juga menambahkan aspek perhitungan lainnya seperti entropi.



Gambar 2.5: *best-fit docking*

Dalam percobaan ini penulis memanfaatkan aplikasi *molecular docking* Autodock dan Autodock Vina.

2.3 *Autodock dan Autodock Vina*

Aplikasi ini merupakan aplikasi yang dikembangkan oleh *The Scripps Research Institute*, lembaga riset nonprofit yang terletak di California, Amerika Serikat. Terdapat 2 jenis aplikasi, Autodock dan Autodock Vina. Masing masing merupakan aplikasi yang saling berbeda dalam segi *scoring function* yang digunakan dan optimisasi komputasi secara paralel oleh Autodock Vina.

Paket aplikasi Autodock terdiri dari 2 program, Autodock4 dan Autogrid4. AutoGrid akan menghasilkan *pre-calculated map* dari suatu *ligand*.Selain itu juga akan menghasilkan *map* tambahan, ”d” untuk *desolvation* dan ”e” untuk *electrostatic*.Pengoperasian Autodock membutuhkan beberapa file, yaitu : *.dpf untuk pengaturan *docking* parameter , hasil *map* berupa *.gpf dari AutoGrid , dan *.pdbqt untuk *receptor* dan *ligand* yang akan diujicoba. Keluaran yang dihasilkan oleh Autodock berupa *.dlg yang berisi hasil perhitungan konformasi posisi dan energi pada setiap *ligand* yang diujicoba. Paket aplikasi Autodock Vina hanya datang dengan sebuah program saja dikarenakan proses *pre-processing docking* dibuat transparan sehingga pengguna tidak perlu tahu dan memahami bagaimana aplikasi tersebut bekerja. Autodock Vina membutuhkan 2 file, *.pdbqt dari *receptor* dan *.pdbqt dari tiap *ligand* yang akan diujicobakan. Keluaran yang dihasilkan juga beruapa *.pdbqt yang beisikan konformitas posisi dan energi.

Gambar 2.6: Logo Autodock

2.4 *Cloud Computing*

Seperti yang telah dijelaskan sebelumnya, pemanfaatan komputer dalam riset mengalami kendala di negara berkembang. Biaya yang harus dikeluarkan dan sumber daya listrik yang dibutuhkan untuk pengadaan *supercomputer* terbilang besar. Selain itu, dibutuhkan pengetahuan lebih terhadap perangkat keras yang akan digunakan tersebut. Hal ini bertujuan supaya pemeliharaan dan penggunaan *supercomputer* lebih baik. Dengan perkembangan teknologi jaringan dan internet yang pesat, muncul teknologi baru yang dikenal dengan *cloud computing*. Teknologi menjawab permasalahan pengadaan perangkat komputasi dan tidak peneliti tidak perlu memahami secara dalam dibalik layanan yang disediakan oleh teknologi tersebut.

Teknologi *cloud computing* sudah berkembang sejak tahun 1990-an. Pada awalnya teknologi ini dikembangkan sebagai solusi dari permasalahan yang dihadapi oleh perusahaan dengan bertambahnya kapasitas data yang perlu disimpan dan pengeluaran yang cukup besar untuk pengadaan perangkat keras beserta konsumsi daya listrik yang dibutuhkan. Definisi dari *cloud computing* dapat dikatakan sebagai virtualisasi dari perangkat komputasi, dimana pengguna mengakses perangkat tersebut dengan menggunakan jaringan internet. Selain itu, teknologi ini memberikan kemudahan dalam penggunaan secara *multiuser*, dimana dapat diakses secara bersamaan.

Terdapat 3 karakteristik dari teknologi *cloud computing*, yaitu virtualisasi, terdistribusi, dan kemudahan dalam pengembangan selanjutnya (*dynamically extendibility*). Virtualisasi merupakan faktor yang utama. Virtualisasi dapat membagi suatu perangkat komputasi secara fisik menjadi beberapa ”virtual” perangkat komputasi. Dengan begitu, kinerja dari perangkat akan dioptimisasi. Alokasi sumber daya perangkat keras tidak ada yang diam secara percuma. Disamping itu, dengan adanya virtualisasi dapat menekan biaya pengeluaran pengadaan perangkat komputasi secara fisik. Yang dimaksud dengan terdistribusi adalah perangkat keras (*physical node*) yang digunakan dalam *cloud computing* digunakan secara terdistribusi. Dalam tahap pengembangan teknologi ini memberikan kemudahan. Kemudahan tersebut merupakan efek dari virtualisasi. Ketika pengguna ingin mengikuti perkembangan yang ada, tidak perlu merubah / membuat kembali dari awal. Pengguna dapat merubah dari level virtual (virtualisasi).

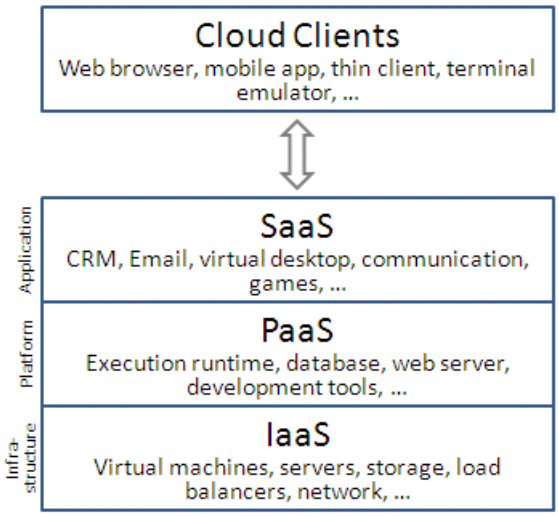
Terdapat 3 jasa yang disediakan dalam *cloud computing* :

- **Software as a Service (SaaS)**
  Layanan jasa ini menyediakan aplikasi dan penyimpanan data (*database*) yang langsung dapat digunakan. Pengguna dapat mengakses aplikasi secara bersamaan dengan pengguna lainnya. SaaS dikenal juga sebagai *on-demand software* dan biaya yang dikenakan oleh penyedia jasa ini merupakan *pay-per-use*. Umumnya, pengguna akan dikenakan biaya bulanan atau tahunan. Pengguna tidak perlu susah - susah untk menginstall aplikasi pada *cloud* dikarenakan penyedia jasa tersebut akan melakukan hal tersebut. Saas memberikan keuntungan bagi perusahaan dengan cara mengurangi biaya operasional IT (Information Technology). Hal ini dikarenakan biaya untuk pemeliharaan dan pengadaan perangkat komputasi ditanggungkan kepada pihak penyedia jasa (*outsourcing*). Contoh dari Saas adalah layanan *e-mail*, media sosial.

- **Platform as a Service (PaaS)**

Berbeda dengan Saas, layanan ini menyediakan perangkat komputasi yang dapat digunakan untuk menjalankan aplikasi dari pengguna. Layanan ini dapat menyediakan sistem operasi atau *framework* yang dibutuhkan dalam menjalankan apliksi tersebut. Contoh dari layanan jasa ini adalah : Windows Azure, Amazon Web Service, dan Google App Engine.

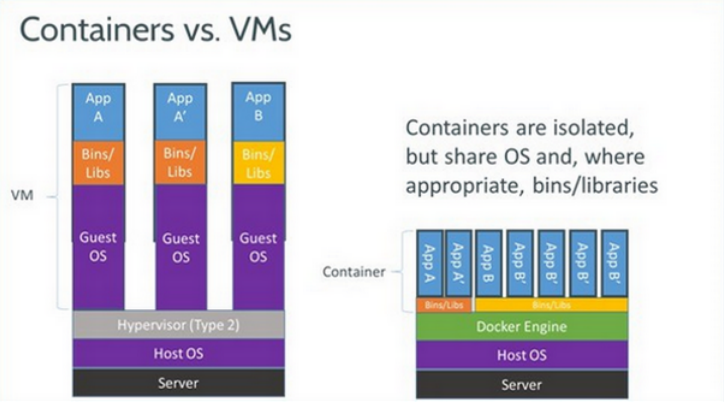
- **Infrastructure as a Service (IaaS)**
  layanan ini menyediakan infrastruktur IT kepada pengguna sesuai dengan permintaan yang dapat diakses dengan jaringan internet. Infrstruktur tersebut meliputi perangkat keras seperti *harddisk, memory, firewall*, tipe server, alamat IP, *virtual local area networks*(VLANs), maupun aplikasi yang harus terinstall dalam server



Gambar 2.7: Layanan dalam *cloud computing*

2.5 Docker

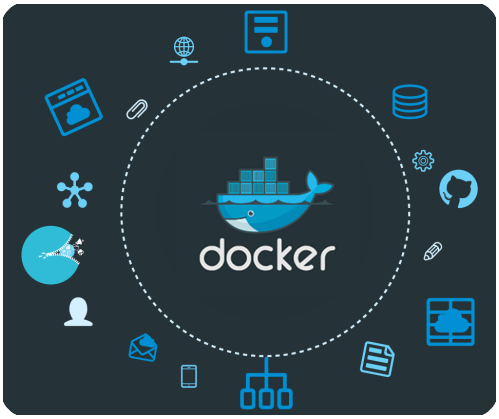
Virtualisasi merupakan faktor utama pada teknologi *cloud computing* dalam mengoptimisasikan kinerja server sehingga tidak ada sumber daya yang tidak terpakai. Virtualisasi yang dimaksud adalah virtualisasi perangkat keras (*hardware*). Virtualisasi tersebut memungkinkan ”virtual” server yang dapat dibuat sesuai dengan kebutuhan yang diperlukan dan berjalan pada server fisik. Spesifikasi untuk masing masing virtual server dapat disesuaikan dengan kebutuhan aplikasi yang akan dijalankan diatasnnya dan tidak melebihi spesifikasi server fisik. Masing - masing virtual server akan saling terisolasi satu dengan yang lainnya. Virtual



Gambar 2.8: Perbandingan virtualisasi pada VM dan container pada Docker

Permasalahan virtualisasi tersebut muncul ketika hendak mengoptimisasikan sumber daya yang ada. Ketika suatu ”virtual” server dibuat untuk menjalankan suatu aplikasi yang spesifik, maka keseluruhan sistem perlu dibuat (Sistem operasi, alokasi sumber daya memori, *processor*) termasuk *library* yang dibutuhkan dalam menjalankan sistem / aplikasi tersebut. ”virtual” server tersebut akan diatur oleh suatu program *hypervisor* yang menjembatani komunikasi dengan server fisik. Dengan begitu, kapasitas memori dan alokasi sumber daya pada server tidak optimal dalam menjalankan ”virtual” server secara bersamaan.

Docker, *platform*virtualisasi yang dikembangkan oleh Perusahaan Docker menggunakan teknik berbeda dengan virtualisasi *hardware*. Docker menggunakan *Docker engine* sebagai ganti dari *hypervisor*.*Docker engine* dikembangkan berdasarkan *Linux containers* (LXC), dimana level virtualisasi terletak pada sistem operasi. Dengan begitu suatu server linux dapat menjalankan beberapa sistem linux yang saling terisolasi satu sama lain. Kernel pada ”virtual” server, dalam hal ini disebut *container* yang dibentuk dengan LXC akan menggunakan kernel yang sama dengan server fisik. Oleh karena itu, *container* yang dibentuk akan lebih kecil dan padat tanpa perlu menginstall sistem secara keseluruhan dari awal. *container* tidak perlu mengalokasikan sumber daya secara tetap dikarenakan *Docker engine* akan mengalokasikan sumber daya berdasarkan aplikasi yang berjalan pada *container*.



Gambar 2.9: Logo Docker

Walaupun begitu, untuk masing masing virtualisasi, baik itu yang menggunakan *hypervisor* maupun *Docker engine* pada Docker memiliki kelebihan dan kekurangan masing - masing. *Container* yang dibentuk pada Docker harus menggunakan sistem operasi yang sama dengan server fisik. Untuk saat ini, Docker baru mendukung sistem operasi Linux sebagai *container* dan berjalan pada sistem operasi Linux. Walaupun dapat berjalan pada sistem operasi lainnya, namun penggunaan Docker masih dijalankan dalam ”virtual” komputer yang menggunakan sistem operasi Linux pada *Virtual Machine*. Dengan alokasi sumber daya yang tidak tetap (sesuai dengan proses yang berjalan), maka *container* yang terbentuk akan lebih banyak dari ”virtual” server yag dihasilkan dengan menggunakan ”hypervisor” pada umumnya terikat dengan batasan sumber daya. Namun ”virtual” server tersebut lebih beragam dan tidak terikat dengan sistem operasi yang berjalan pada server fisik.

BAB 3
METODOLOGI PENELITIAN

@todo
tambahkan kata-kata pengantar bab 1 disini

3.1 Satu Persamaan

$$\frac{y-y_1}{y_2-y_1}=\frac{x-x_1}{x_2-x_1} \tag{3.1}$$

Persamaan 3.1 diatas adalah persamaan garis. Persamaan 3.1 dan 3.2 sama-sama dibuat dengan perintah \align. Perintah ini juga dapat digunakan untuk menulis lebih dari satu persamaan.

$$\underbrace{|\overline{ab}|}_{\text{pada bola } |\overline{ab}|=r} = \sqrt[2]{(x_b-x_a)^2+(y_b-y_a)^2+||(z_b-z_a)^2} \tag{3.2}$$

3.2 Lebih dari Satu Persamaan

$$|\overline{a}*\overline{b}| = |\overline{a}||\overline{b}|\sin\theta \tag{3.3}$$

$$\overline{a}*\overline{b} = \begin{vmatrix} \hat{i} & x_1 & x_2 \\ \hat{j} & y_1 & y_2 \\ \hat{k} & z_1 & z_2 \end{vmatrix}$$

$$= \hat{i}\begin{vmatrix} y_1 & y_2 \\ z_1 & z_2 \end{vmatrix} + \hat{j}\begin{vmatrix} z_1 & z_2 \\ x_1 & x_2 \end{vmatrix} + \hat{k}\begin{vmatrix} x_1 & x_2 \\ y_1 & y_2 \end{vmatrix}$$

Pada Persamaan 3.3 dapat dilihat beberapa baris menjadi satu bagian dari Persamaan 3.3. Sedangkan dibawah ini dapat dilihat bahwa dengan cara yang sama, Persamaan 3.4, 3.5, dan 3.6 memiliki nomor persamaannya masing-masing.

$$\int_a^b f(x)\,dx + \int_b^c f(x)\,dx = \int_a^c f(x)\,dx \tag{3.4}$$

$$\lim_{x\to\infty} \frac{f(x)}{g(x)} = 0 \qquad \text{jika pangkat } f(x) < \text{pangkat } g(x) \tag{3.5}$$

$$a^{m^{a^n \log b}} = b^{\frac{m}{n}} \tag{3.6}$$

BAB 4
HASIL PENELITIAN DAN ANALISIS

@todo
tambahkan kata-kata pengantar bab 1 disini

4.1 thesis.tex

Berkas ini berisi seluruh berkas Latex yang dibaca, jadi bisa dikatakan sebagai berkas utama. Dari berkas ini kita dapat mengatur bab apa saja yang ingin kita tampilkan dalam dokumen.

4.2 laporan_setting.tex

Berkas ini berguna untuk mempermudah pembuatan beberapa template standar. Anda diminta untuk menuliskan judul laporan, nama, npm, dan hal-hal lain yang dibutuhkan untuk pembuatan template.

4.3 istilah.tex

Berkas istilah digunakan untuk mencatat istilah-istilah yang digunakan. Fungsinya hanya untuk memudahkan penulisan. Pada beberapa kasus, ada kata-kata yang harus selalu muncul dengan tercetak miring atau tercetak tebal. Dengan menjadikan kata-kata tersebut sebagai sebuah perintah LaTeX tentu akan mempercepat dan mempermudah pengerjaan laporan.

4.4 hype.indonesia.tex

Berkas ini berisi cara pemenggalan beberapa kata dalam bahasa Indonesia. LaTeX memiliki algoritma untuk memenggal kata-kata sendiri, namun untuk beberapa kasus algoritma ini memenggal dengan cara yang salah. Untuk memperbaiki pemenggalan yang salah inilah cara pemenggalan yang benar ditulis dalam berkas hype.indonesia.tex.

4.5 pustaka.tex

Berkas pustaka.tex berisi seluruh daftar referensi yang digunakan dalam laporan. Anda bisa membuat model daftar referensi lain dengan menggunakan bibtex. Untuk mempelajari bibtex lebih lanjut, silahkan buka http://www.bibtex.org/Format. Untuk merujuk pada salah satu referensi yang ada, gunakan perintah \cite, e.g. \cite{latex.intro} yang akan akan memunculkan [1]

4.6 bab[1 - 6].tex

Berkas ini berisi isi laporan yang Anda tulis. Setiap nama berkas e.g. bab1.tex merepresentasikan bab dimana tulisan tersebut akan muncul. Sebagai contoh, kode dimana tulisan ini dibaut berada dalam berkas dengan nama bab4.tex. Ada enam buah berkas yang telah disiapkan untuk mengakomodir enam bab dari laporan Anda, diluar bab kesimpulan dan saran. Jika Anda tidak membutuhkan sebanyak itu, silahkan hapus kode dalam berkas thesis.tex yang memasukan berkas LaTeX yang tidak dibutuhkan; contohnya perintah \include{bab6.tex} merupakan kode untuk memasukan berkas bab6.tex kedalam laporan.

BAB 5
PERINTAH DALAM UITHESIS.STY

@todo

Tambahkan kata-kata pengantar bab 5 disini.

5.1 Mengubah Tampilan Teks

Beberapa perintah yang dapat digunakan untuk mengubah tampilan adalah:

- \f
  Merupakan alias untuk perintah \textit, contoh *contoh hasil tulisan*.
- \bi
  Contoh hasil tulisan.
- \bo
  Contoh hasil tulisan.
- \m
  $Contohhasiltulisan$.
- \mc

  $$Contohhasiltulisan$$

  .
- \code
  `Contoh hasil tulisan`.

5.2 Memberikan Catatan

Ada dua perintah untuk memberikan catatan penulisan dalam dokumen yang Anda kerjakan, yaitu:

- \todo

  Contoh:

  @todo

  Contoh bentuk todo.

- \todoCite

  Contoh:

  @todo

  Referensi

5.3 Menambah Isi Daftar Isi

Terkadang ada kebutuhan untuk memasukan kata-kata tertentu kedalam Daftar Isi. Perintah \addChapter dapat digunakan untuk judul bab dalam Daftar isi. Contohnya dapat dilihat pada berkas thesis.tex.

5.4 Memasukan PDF

Untuk memasukan PDF dapat menggunakan perintah \inpdf yang menerima satu buah argumen. Argumen ini berisi nama berkas yang akan digabungkan dalam laporan. PDF yang dimasukan degnan cara ini akan memiliki header dan footer seperti pada halaman lainnya.

Untitled

Ini adalah berkas pdf yang dimasukan dalam dokumen laporan.

Cara lain untuk memasukan PDF adalah dengan menggunakan perintah \putpdf dengan satu argumen yang berisi nama berkas pdf. Berbeda dengan perintah sebelumnya, PDF yang dimasukan dengan cara ini tidak akan memiliki footer atau header seperti pada halaman lainnya.

Ini adalah berkas pdf yang dimasukan dalam dokumen laporan.

5.5 Membuat Perintah Baru

Ada dua perintah yang dapat digunakan untuk membuat perintah baru, yaitu:

- \Var
  Digunakan untuk membuat perintah baru, namun setiap kata yang diberikan akan diproses dahulu menjadi huruf kapital. Contoh jika perintahnya adalah \Var{adalah} makan ketika perintah \Var dipanggil, yang akan muncul adalah ADALAH.

- \var
  Digunakan untuk membuat perintah atau baru.

BAB 6

??

@todo
tambahkan kata-kata pengantar bab 6 disini

BAB 7
KESIMPULAN DAN SARAN

@todo
Tambahkan kesimpulan dan saran terkait dengan perkerjaan yang dilakukan.

7.1 Kesimpulan

7.2 Saran

DAFTAR REFERENSI

[1] Jeff Clark. (n.d). *Introduction to LaTeX*. 26 Januari 2010. http://frodo.elon.edu/tutorial/tutorial/node3.html.

LAMPIRAN

LAMPIRAN 1