BAB IV

HASIL DAN PEMBAHASAN

4.1 Perancangan Sistem

Sistem yang dibuat merupakan usulan perancangan sistem untuk memperbaiki sistem pengolahan *project* (yang selanjutnya akan disebut *plan*) dan *task* yang sedang berjalan sebelumnya. Maka solusi yang diusulkan adalah dengan membuat sistem informasi *Task Management* berbasis web, sistem informasi ini akan memuat fasilitas yang dibutuhkan sebagai berikut :

1. Kemudahan distribusi pekerjaan, memungkinkan para *leader* memberikan *task* kepada timnya melalui sistem informasi *Task Management*.
2. Kemudahan pelaporan pekerjaan, memungkinkan anggota tim melaporkan hasil pekerjaan nya melalui sistem informasi *Task Management*.
3. Kemudahan *monitoring progress plan*, memungkinkan leader dan pihak manajemen lainnya untuk memantau *progress plan*.
4. Analisa dan Evaluasi, dengan visualisasi dalam bentuk grafik maka sistem dapat memberikan informasi yang berguna untuk kebutuhan analisa dan evaluasi. Analisa disini mencakup analisa *progress* suatu *plan* dan juga analisa sumber daya manusia, sehingga dari hasil analisa tersebut pihak menajemen dapat memberikan evaluasi kepada tim nya.

4.1.1 Tujuan Perancangan Sistem

Adapun tujuan perancangan sistem yang diusulkan, yaitu untuk menghasilkan suatu rancangan sistem yang dapat membantu dalam pengolahan *plan* dan *task* yang terintegrasi dengan database dan sistem informasi *Task Management* berbasis web ini bisa mencapai sasaran kebutuhan dan tujuannya untuk membantu PT. eBdesk Teknologi dalam pengolahan *plan* dan *task*.

4.1.2 Gambaran Umum Sistem Yang Diusulkan

Dalam tahap ini gambaran umum sistem yang di usulkan bertujuan untuk menghasilkan perancangan sistem pengolahan *plan* dan *task* terkomputerisasi, lebih mudah di lakukan, bisa dilakukan kapanpun dan dimanapun serta keamanan data yang terjaga.

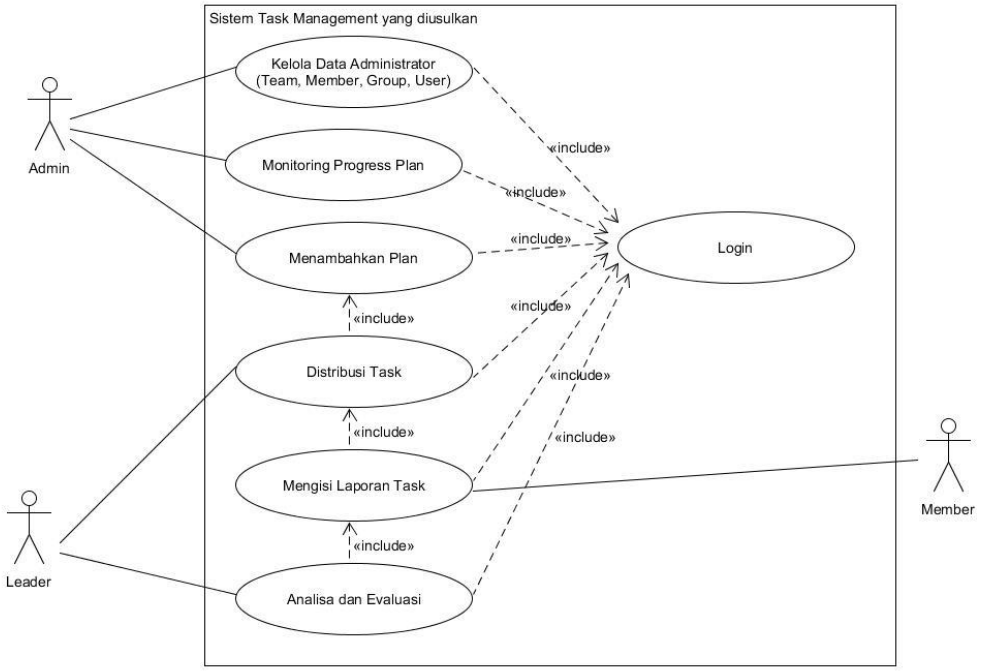
Usulan peracangan yang dilakukan adalah merubah sistem pengolahan *plan* dan *task* yang masih menggunakan cara manual menjadi sistem yang terkomputerisasi. Tahapan perancangan prosedur ini akan dijelaskan dengan menggunakan metode pendekatan sistem berorientasi objek yaitu *Unified Modeling Language* (UML).

4.1.3 Perancangan Prosedur Yang Diusulkan

Perancangan yang diusulkan ini meliputi use case diagram, *activity* diagram, *sequence* diagram, *class* diagram dan *deployment* diagram yang dapat menghasilkan sebuah sistem yang lebih baik. Proses yang dirancang dan diuraikan menjadi beberapa bagian yang dapat membentuk sistem tersebut menjadi satu kesatuan komponen.

4.1.3.1 *Use Case Diagram*

Berikut adalah alur sistem dalam pembuatan sistem informasi *Task Mangement* berbasis web di PT eBdesk Teknologi.



Gambar 4.1 1Use Case Diagram yang Diusulkan

4.1.3.1.1 Definisi Aktor dan Deskripsinya

Berikut ini adalah penjelasan dari aktor yang terlibat pada proses sistem informasi *Task Management* berbasis web yang diusulkan di PT eBdesk Teknologi.

Table 4.1 Definisi Aktor

No.	Aktor	Deskripsi
1.	*Leader*	*Leader* bertugas sebagai manajer yang mana tugasnya adalah mengawasi,

		mendistribusikan pekerjaan serta menganalisa dan mengevaluasi.
2.	*Member*	*Member* bertugas sebagai anggota tim yang melaporkan hasil pekerjaan dalam suatu *plan*.
3.	*Admin*	*Admin* bertugas sebagai *general leader* yang mana dapat mengelola semua *plan* dan *task* yang ada di dalam sistem. Selain itu admin juga memiliki akses untuk mengelola data-data administrator, seperti data *team*, *member*, *group* dan *user*.

4.1.3.1.2 Definisi *Use Case* dan Deskripsinya

Berikut ini adalah penjelasan dari *use case* yang terlibat pada proses sistem informasi *Task Management* berbasis web yang di usulkan di PT eBdesk Teknologi.

Table 4.2 Definisi *Use Case*

No.	*Use Case*	Deskripsi
1.	Menambahkan *Plan*	*Use Case* menambahkan *plan* ini merupakan proses dimana leader membawa permintaan aplikasi baru lalu direalisasikan kedalam sistem informasi *Task Management* dalam bentuk *plan*.

2.	Distribusi *Task*	*Use Case* distribusi *task* ini merupakan proses dimana leader akan merumuskan *task* apa yang harus dikerjakan untuk menyelesaikan suatu *plan* lalu mendistribusikannya kepada anggota tim yang terlibat dalam *plan* tersebut.
3.	Mengisi Laporan *Task*	*Use Case* mengisi laporan *task* merupakan proses dimana anggota tim melaporkan hasil pekerjaannya dari *task* yang telah didistribusikan oleh leader nya.
4.	Analisa dan Evaluasi	*Use Case* analisa dan evaluasi merupakan proses dimana leader menganalisa *progress* suatu *plan*.
5.	Kelola Data Administrator (*Team, Member, Group* dan *User*)	*Use Case* kelola data administrator merupakan proses dimana admin dapat mengelola data-data master yang terkait dengan data administrator, seperti data *team, member, group* dan *user*.
6	*Monitoring Progress Plan*	*Use Case monitoring progress plan* merupakan proses dimana admin memantau progress dari semua plan secara sekaligus. Admin dapat melihat *plan* terbaru, *plan* yang

		progress-nya terhambat serta *plan* yang sudah mendekati masa *deadline*-nya.

4.1.3.2 *Skenario Use Case*

Skenario *Use Case* berfungsi untuk mengetahui alur cerita agar lebih mudah dimengerti. Berdasarkan *Use Case* diagram yang diusulkan, maka sekenario *Use Case* adalah sebagai berikut :

4.1.3.2.1 Skenario *Use Case* Menambahkan *Plan*

Berikut adalah proses menambahkan *plan* yang dilakukan oleh leader di PT. eBdesk Teknologi.

Table 4.3 Skenario *Use Case* Menambahkan *Plan*

Identifikasi	
No *Use Case*	UC-01
Nama	Menambahkan *Plan*
Tujuan	Merealisasikan permintaan aplikasi baru kedalam sistem informasi *Task Management*.
Aktor	Admin
Skenario Utama	
Kondisi Awal	Permintaan aplikasi baru belum direalisasikan ke dalam sistem
Kondisi Akhir	Permintaan aplikasi baru sudah direalisasikan ke dalam sistem

Leader	Sistem
1. Mengisi form *login*	
	2. Memverifikasi data user
	3. Menampilkan halaman utama
4. Menekan tombol tambah *plan* ()	
	5. Menampilkan (*pop up*) form tambah *plan*
6. Mengisi form tambah *plan*	
	7. Menyimpan data *plan* kedalam *database*
	8. Memuat ulang *list plan* di halaman utama
Skenario Alternatif	
	9. Jika data user tidak sesuai
	10. Maka sistem menampilkan peringatan dan kembali ke halaman *login*

4.1.3.2.2 Skenario *Use Case* Distribusi *Task*

Table 4.4 1Skenario *Use Case* Distribusi *Task*

Identifikasi	
No *Use Case*	UC-02
Nama	Distribusi *Task*
Tujuan	Merumuskan dan mendistribusikan *task* yang harus dikerjakan dalam suatu *plan* kepada anggota tim yang terlibat dalam plan tersebut
Aktor	Leader
Skenario Utama	
Kondisi Awal	*Task* belum dirumuskan dan didistribusikan kepada anggota tim
Kondisi Akhir	*Task* sudah dirumuskan dan didistribusikan kepada anggota tim

Leader	**Sistem**
1. Mengisi form *login*	
	2. Memverifikasi data user
	3. Menampilkan halaman utama
4. Memilih salah satu *plan* dihalaman utama	
	5. Menampilkan halaman *board*

	6. Jika *plan* yang dipilih belum memiliki *board*
	7. Maka sistem akan menambahkan *board default* kedalam *plan* tersebut. Secara *default board* terdiri dari : • Backlog • To Do • In Pending • In Progress • In Review • Done
8. Menekan tombol tambah *task* (+) di *board* “Backlog”	
	9. Menampilkan (*pop up*) form tambah *task*
10. Mengisi form tambah *task*	
	11. Menyimpan data *task* kedalam *database*
	12. Memuat ulang *list task* didalam *board*

13. Jika rumusan *task* sudah matang dan siap dikerjakan oleh anggota tim yang ditugaskan	
14. Task dipindahkan dengan cara diseret ke dalam *board* "To Do"	
	15. Meng-*update* data *task* kedalam *database*
Skenario Alternatif	
	16. Jika data user tidak sesuai
	17. Maka sistem menampilkan peringatan dan kembali ke halaman *login*

4.1.3.2.3 Skenario *Use Case* Mengisi Laporan *Task*

Table 4.5 Skenario *Use Case* Mengisi Laporan *Task*

Identifikasi	
No *Use Case*	UC-03
Nama	Mengisi Laporan *Task*
Tujuan	Melaporkan hasil pekerjaan dari *task* yang telah didistribusikan oleh *leader* nya
Aktor	*Member*
Skenario Utama	

Kondisi Awal	*Task* belum diisi dengan laporan pekerjaan
Kondisi Akhir	*Task* sudah diisi dengan laporan pekerjaan

Member	**Sistem**
1. Mengisi form *login*	
	2. Memverifikasi data user
	3. Menampilkan halaman utama
4. Memilih salah satu *plan* dihalaman utama	
	5. Menampilkan halaman *board*
6. Memilih salah satu *task* yang akan diisi dengan laporan	
	7. Menampilkan (*pop up*) properti *task* yang dipilih
8. Mengisi kolom komentar dengan laporan pekerjaan atau meng-*upload* lampiran	
	9. Menyimpan data laporan kedalam *database*
10. Jika status *task* berubah	
11. Maka *task* dipidahkan dengan cara diseret ke *board* yang sesuai dengan status barunya	

	12. Meng-*update* status *task* kedalam *database*
Skenario Alternatif	
	13. Jika data user tidak sesuai
	14. Maka sistem menampilkan peringatan dan kembali ke halaman *login*

4.1.3.2.4 Skenario *Use Case* Analisa dan Evaluasi

Table 4.6 2Skenario *Use Case* Analisa dan Evaluasi

Identifikasi	
No *Use Case*	UC-04
Nama	Analisa dan Evaluasi
Tujuan	Menampilkan grafik *User Repartition*, *Task Distribution* dan *Cumulative Flow Diagram* dari suatu *plan*
Aktor	*Leader*
Skenario Utama	
Kondisi Awal	Grafik belum ditampilkan
Kondisi Akhir	Grafik sudah ditampilkan
Member	**Sistem**
1. Mengisi form *login*	
	2. Memverifikasi data user

	3. Menampilkan halaman utama
4. Memilih salah satu *plan* dihalaman utama	
	5. Menampilkan halaman *board*
6. Untuk menampilkan grafik masuk kedalam submenu "Analytics"	
	7. Menampilkan grafik *User Repartition*, *Task Distribution* dan *Cumulative Flow Diagram*
Skenario Alternatif	
	8. Jika data user tidak sesuai
	9. Maka sistem menampilkan peringatan dan kembali ke halaman *login*

4.1.3.2.5 Skenario *Use Case* Kelola Data Administrator

Tabel 4.7. Skenario *Use Case* Kelola Data Administrator

Identifikasi	
No *Use Case*	UC-05
Nama	Kelola Data Administrator

Tujuan	Mengelola (tambah, lihat, *edit*, hapus) data-data yang bersifat administrator
Aktor	*Admin*
Skenario Utama	
Kondisi Awal	Data belum dapat dikelola
Kondisi Akhir	Data dapat dikelola

Member	**Sistem**
1. Mengisi form *login*	
	2. Memverifikasi data user
	3. Menampilkan halaman utama
4. Memilih salah satu menu *team*, *member*, *group* atau *user*.	
	5. Menampilkan list data yang dipilih dan form tambah data.
6. Mengelola data (tambah, lihat, *edit* atau hapus)	
	7. Menambahkan / meng-*update* / menghapus data yang ada di dalam *database*.
Skenario Alternatif	
	8. Jika data user tidak sesuai

	9. Maka sistem menampilkan peringatan dan kembali ke halaman *login*

4.1.3.2.6 Skenario *Use Case Monitoring Progress Plan*

Tabel 4.8. Skenario *Use Case Monitoring Progress Plan*

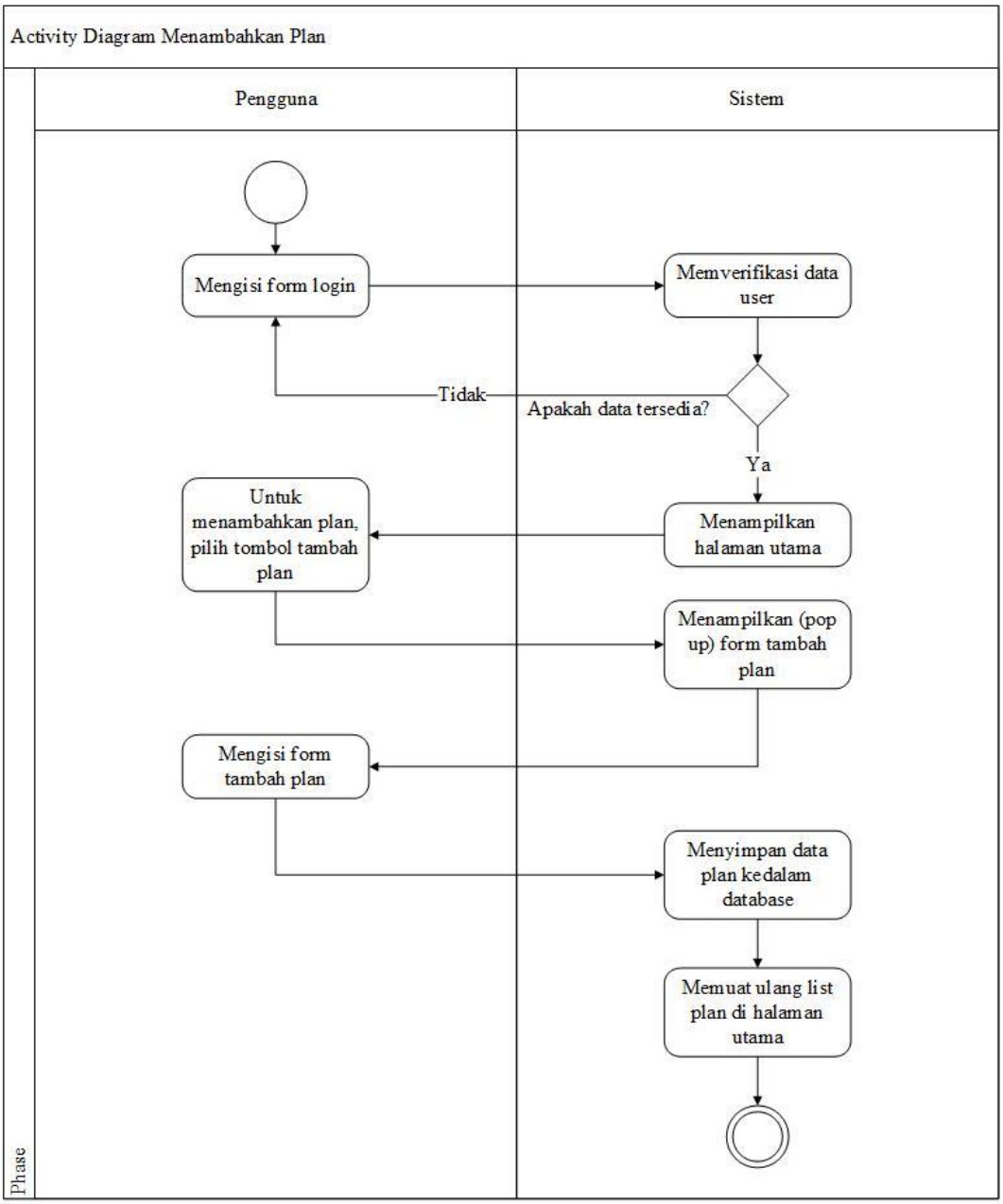
Identifikasi	
No *Use Case*	UC-06
Nama	*Monitoring Progress Plan*
Tujuan	Memantau *progress* semua *plan* yang ada di dalam sistem informasi *Task Management*
Aktor	*Admin*
Skenario Utama	
Kondisi Awal	*Admin* tidak mengetahui *plan* mana saja yang *progress*-nya terhambat atau *plan* yang mendekati tanggal *deadline*
Kondisi Akhir	*Admin* dapat mengetahui *plan* mana saja yang *progress*-nya terhambat atau *plan* yang mendekati tanggal *deadline*
Member	**Sistem**
1. Mengisi form *login*	
	2. Memverifikasi data user
	3. Menampilkan halaman utama

4. Memilih salah satu menu *monitoring*	
	5. Menampilkan list *plan* terbaru, *plan* yang *progress*-nya terhambat dan *plan* yang mendekati tanggal *deadline*
Skenario Alternatif	
	6. Jika data user tidak sesuai
	7. Maka sistem menampilkan peringatan dan kembali ke halaman *login*

4.1.3.3 *Acivity Diagram*

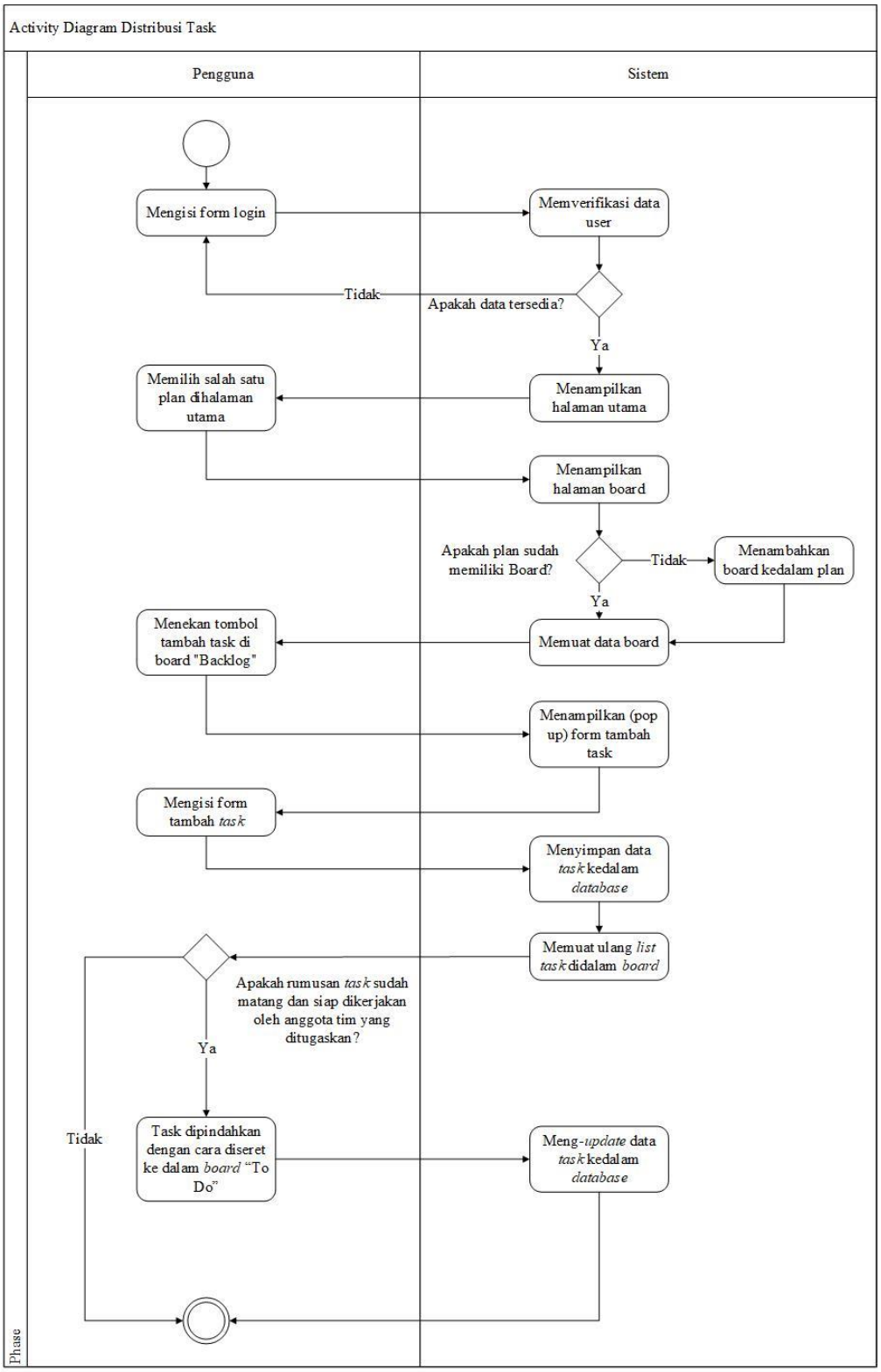
Activity Diagram berfungsi untuk menggambarkan alur aktivitas sistem yang sedang dibuat, bagaimana alur aktivitas itu berawal, dan bagaimana alur aktivitas tersebut berakhir.

4.1.3.3.1 *Activity* Diagram Menambahkan *Plan*



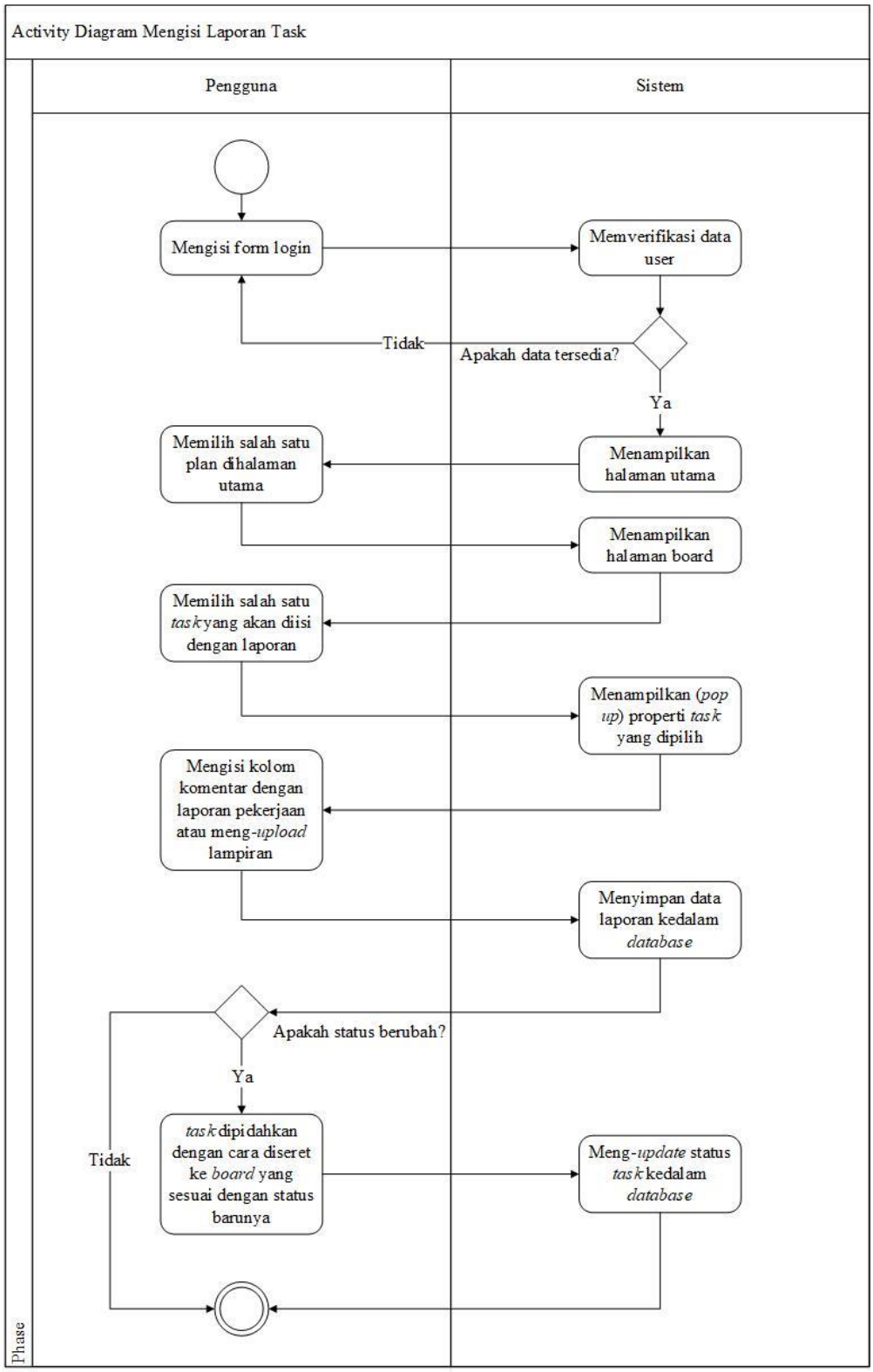
Gambar 4.2 Activity Diagram Menambahkan Plan

4.1.3.3.2 *Activity* Diagram Distribusi *Task*



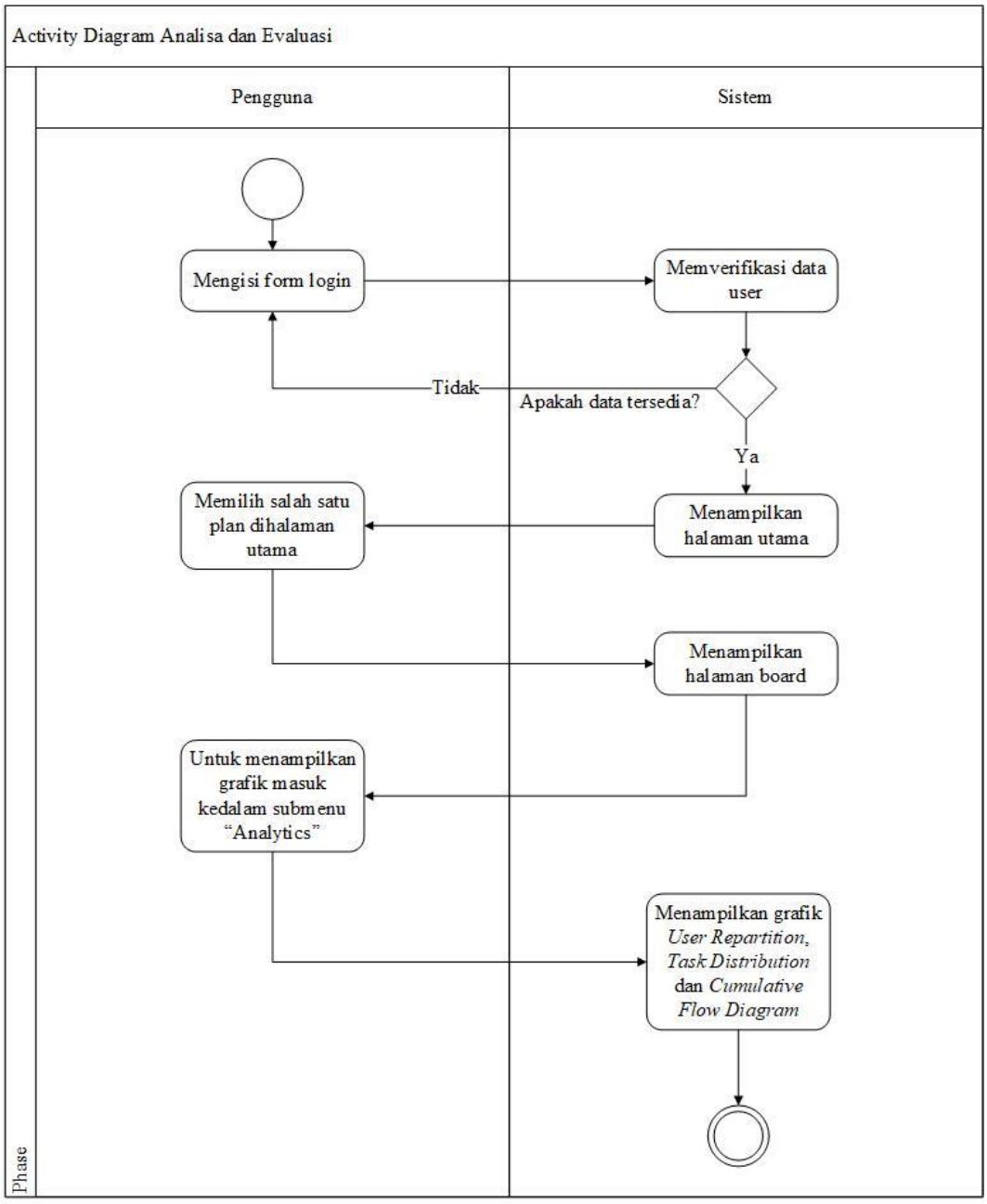
Gambar 4.3 Activity Diagram Distribusi Task

4.1.3.3.3 *Activity* Diagram Mengisi Laporan *Task*



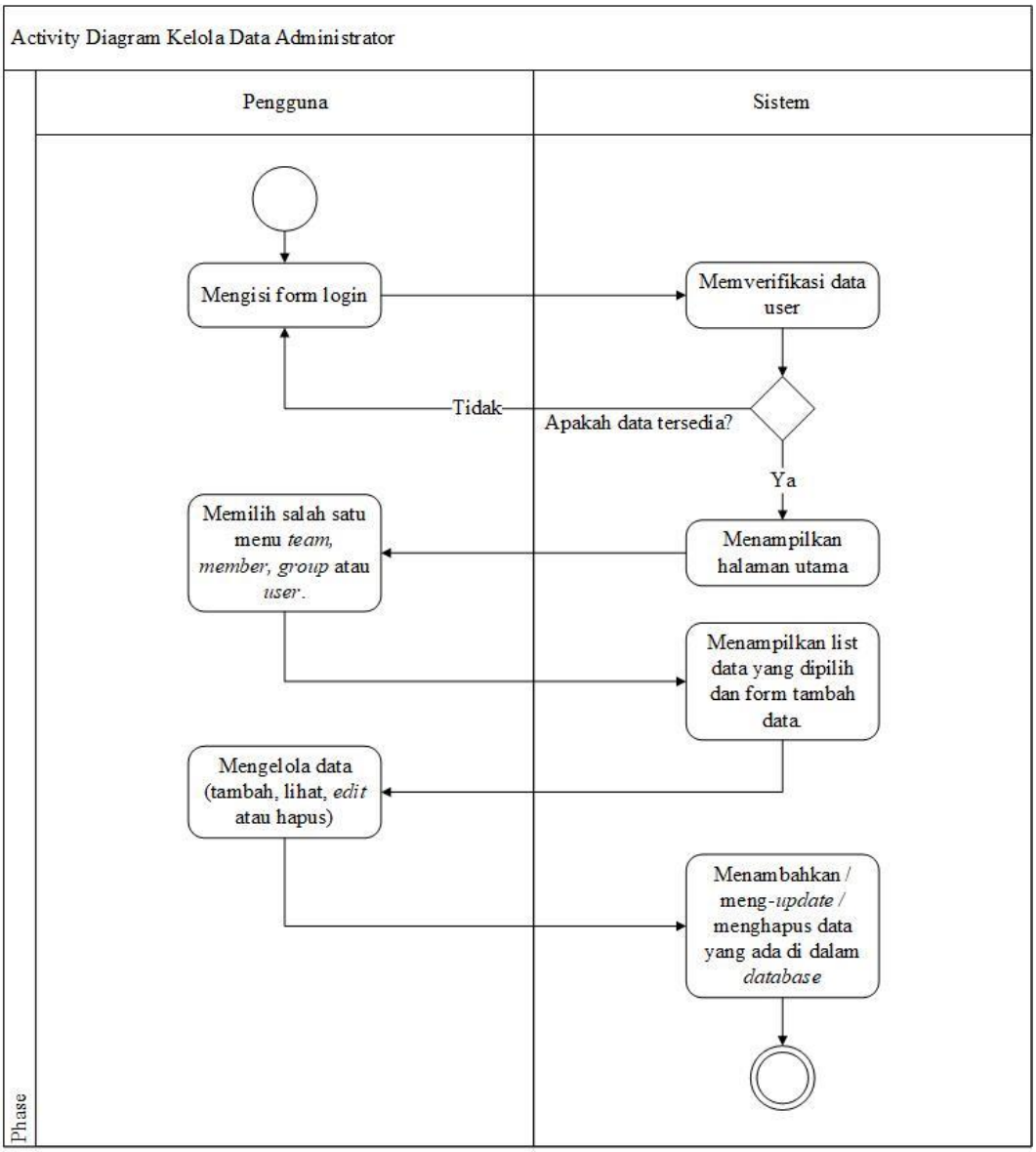
Gambar 4.4 Activity Diagram Mengisi Laporan Task

4.1.3.3.4 *Activity* Diagram Analisa dan Evaluasi



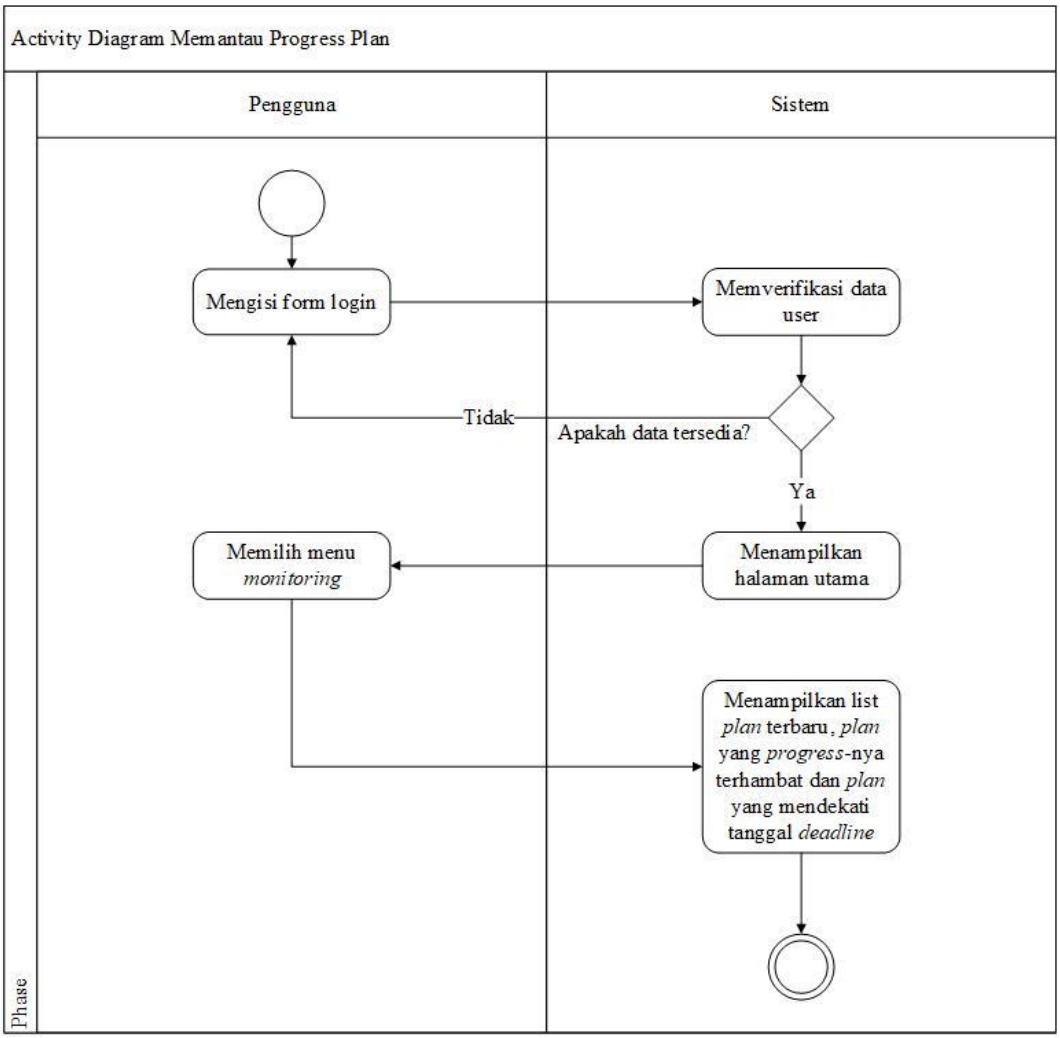
Gambar 4.5 Activity Diagram Analisa dan Evaluasi

4.1.3.3.5 *Activity* Diagram Kelola Data Administrator



Gambar 4.6 Activity Diagram Kelola Data Administrasi

4.1.3.3.6 *Activity* Diagram *Monitoring Progress Plan*

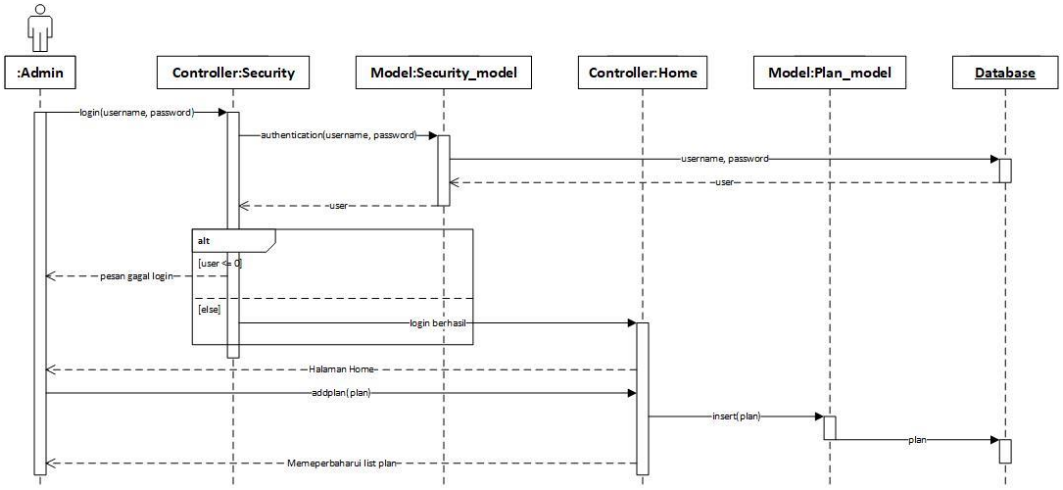


Gambar 4.7. ***Activity Diagram Monitoring Progress Plan***

4.1.3.4 *Sequence Diagram*

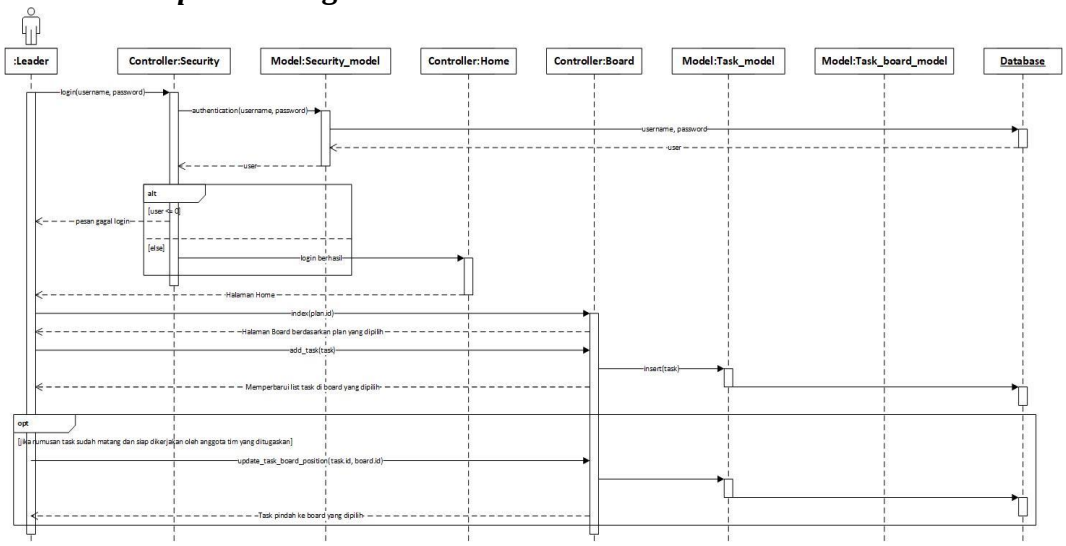
Sequence diagram atau diagram urutan adalah diagram interaksi yang menekankan pada pengiriman pesan (*message*) dalam suatu waktu tertentu. Berikut ini adalah sequence diagram yang diusulkan :

4.1.3.4.1 *Sequence* Diagram Menambahkan *Plan*



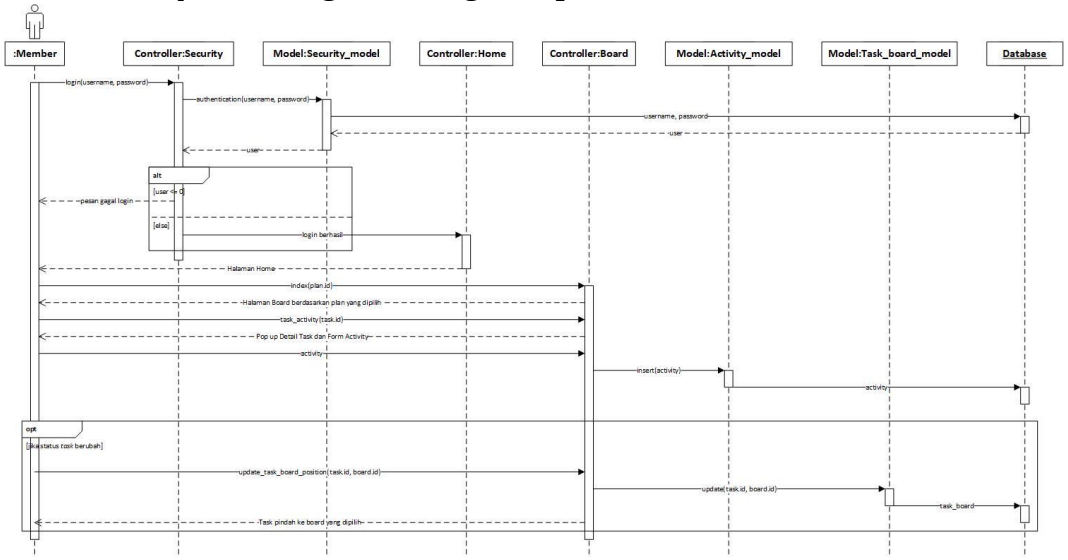
Gambar 4.7 Sequence Diagram Menambahkan Plan

4.1.3.4.2 *Sequence* Diagram Distribusi *Task*



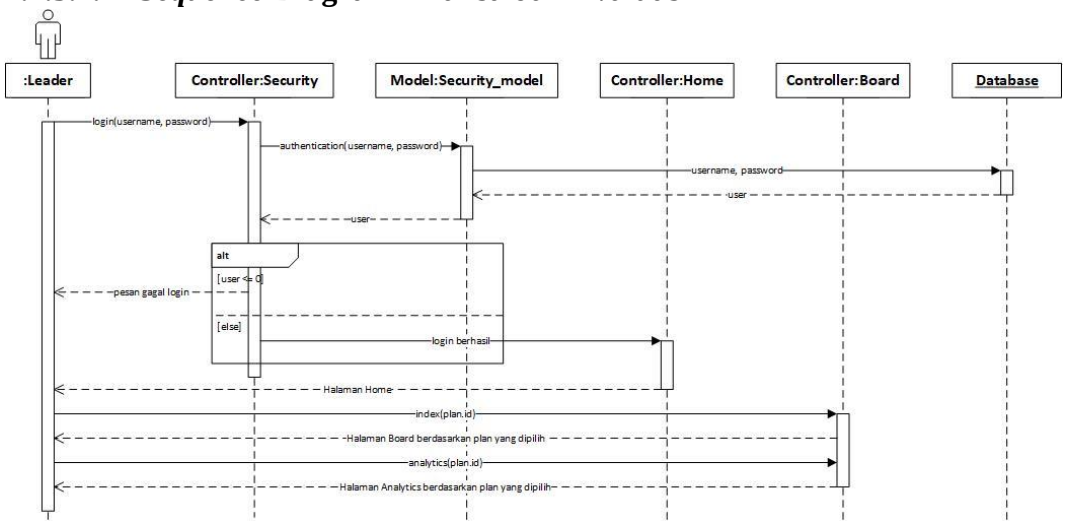
Gambar 4.8 Sequence Diagram Distribusi Task

4.1.3.4.3 *Sequence* Diagram Mengisi Laporan *Task*



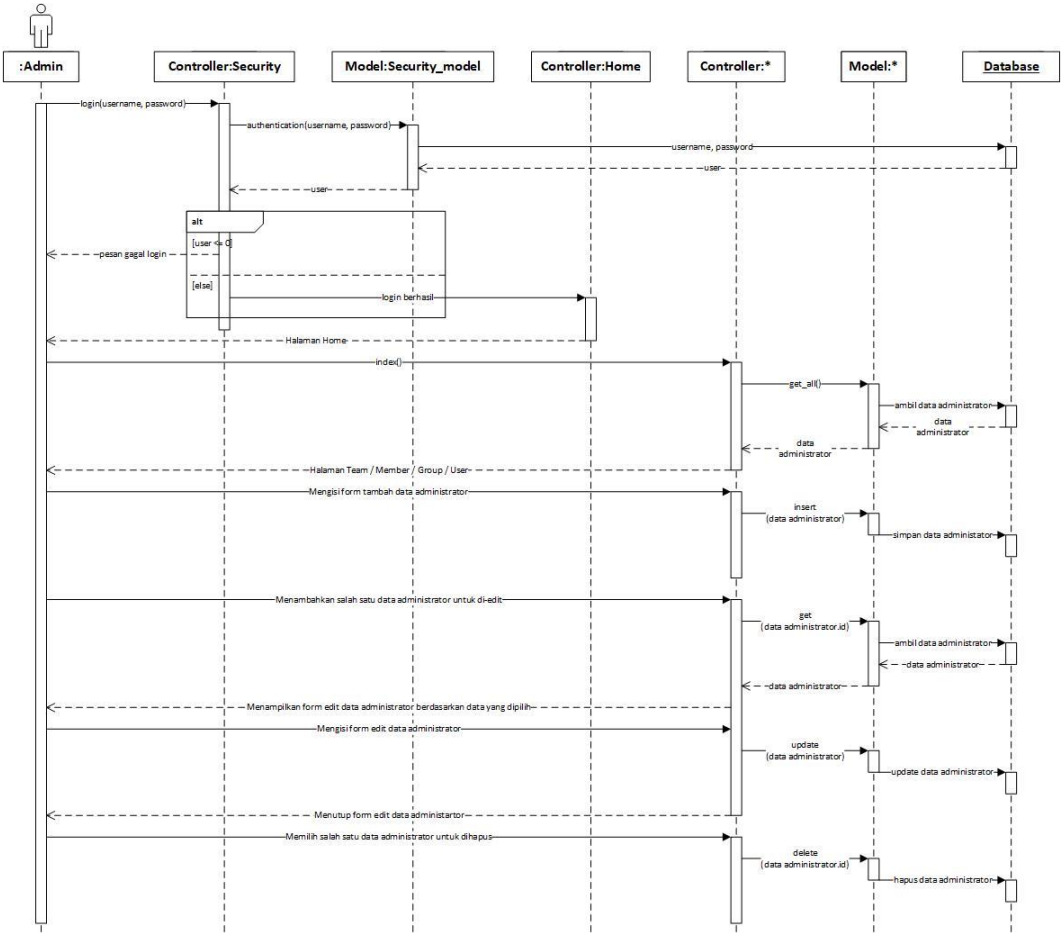
Gambar 4.9 Sequence Diagram Mengisi Laporan Task

4.1.3.4.4 *Sequence* Diagram Analisa dan Evaluasi



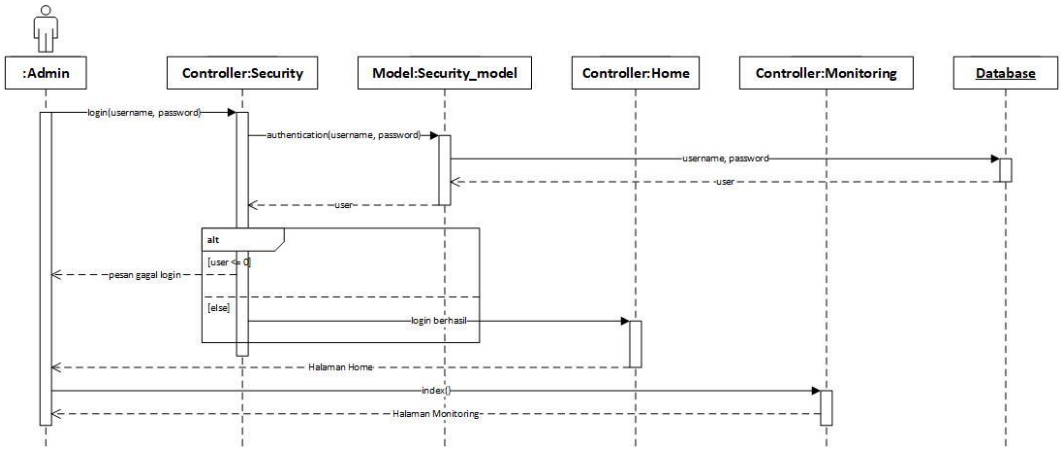
Gambar 4.10 Sequence Diagram Analisa dan Evaluasi

4.1.3.4.5 *Sequence* Diagram Kelola Data Administrator



Gambar 4.11 Sequence Diagram Data Administrator

4.1.3.4.6 *Sequence* Diagram *Monitoring Progress Plan*

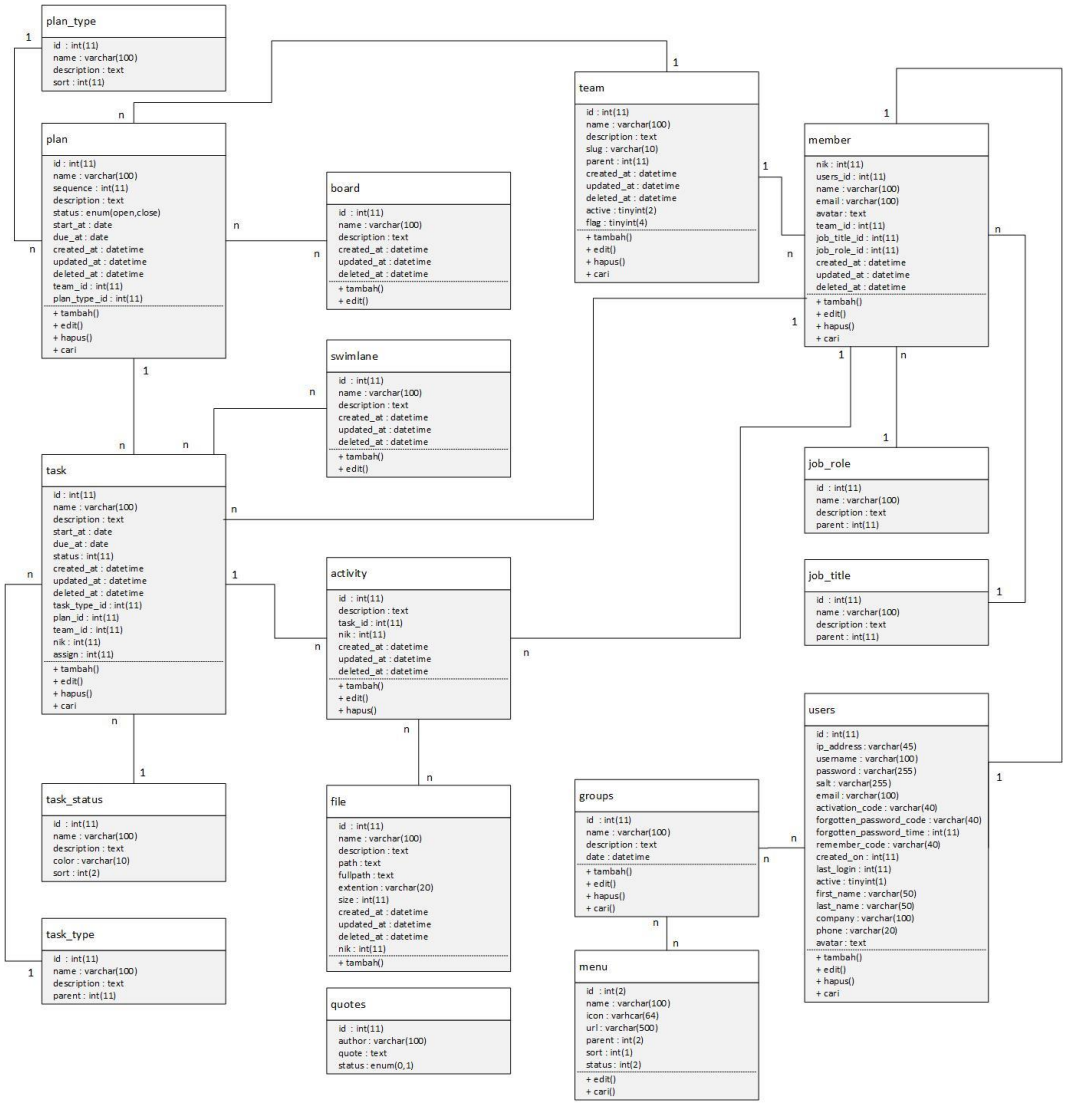


Gambar 4.13. *Sequence Diagram Monitoring Progress Plan*

4.1.4 Perancangan Data

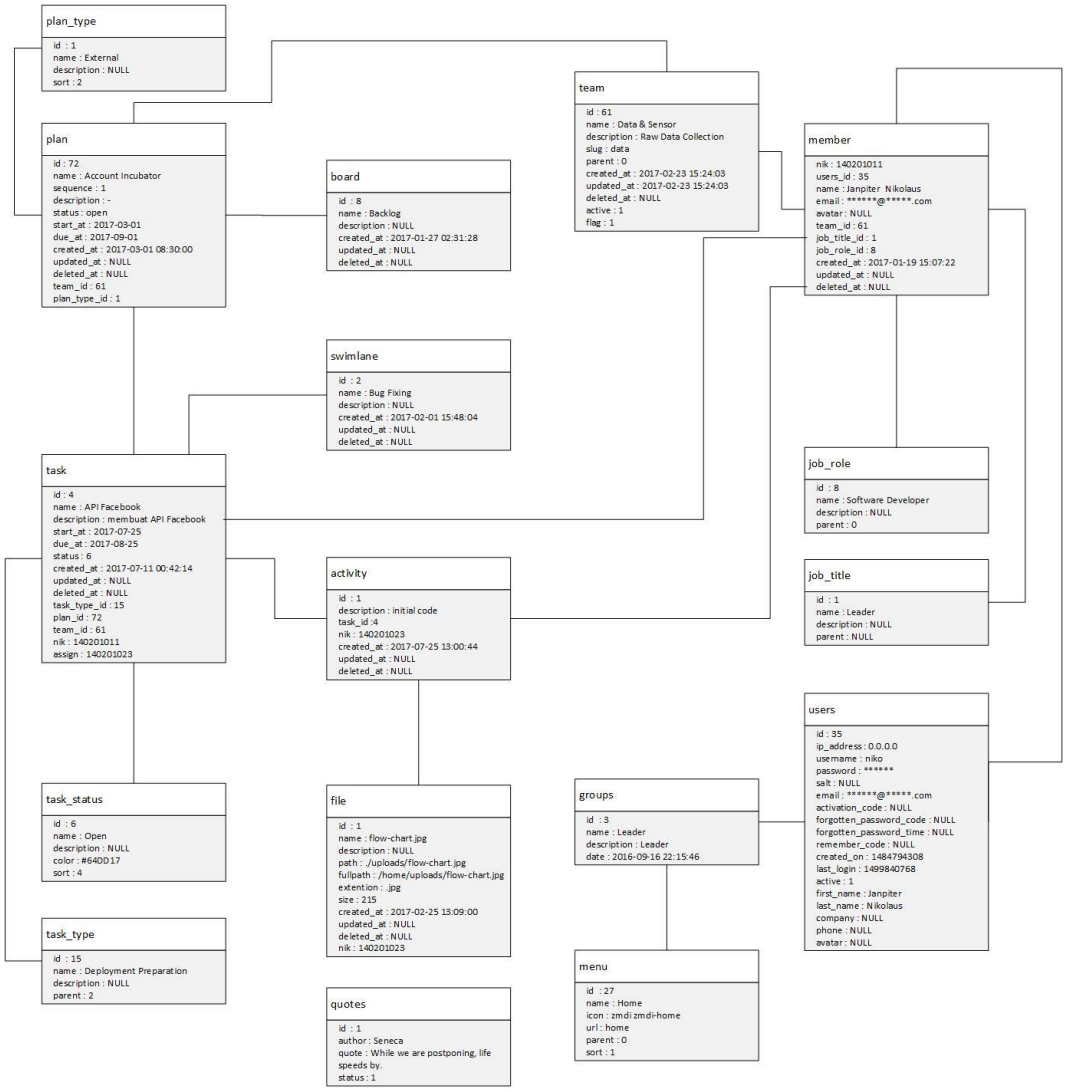
4.1.4.1 *Class Diagram*

Berikut ini adalah relasi antar kelas dari *class diagram* sistem informasi *Task Management* berbasis web. Berikut ini adalah relasi antar kelas dari *class diagram* yang diusulkan :



Gambar 4.12 Class Diagram

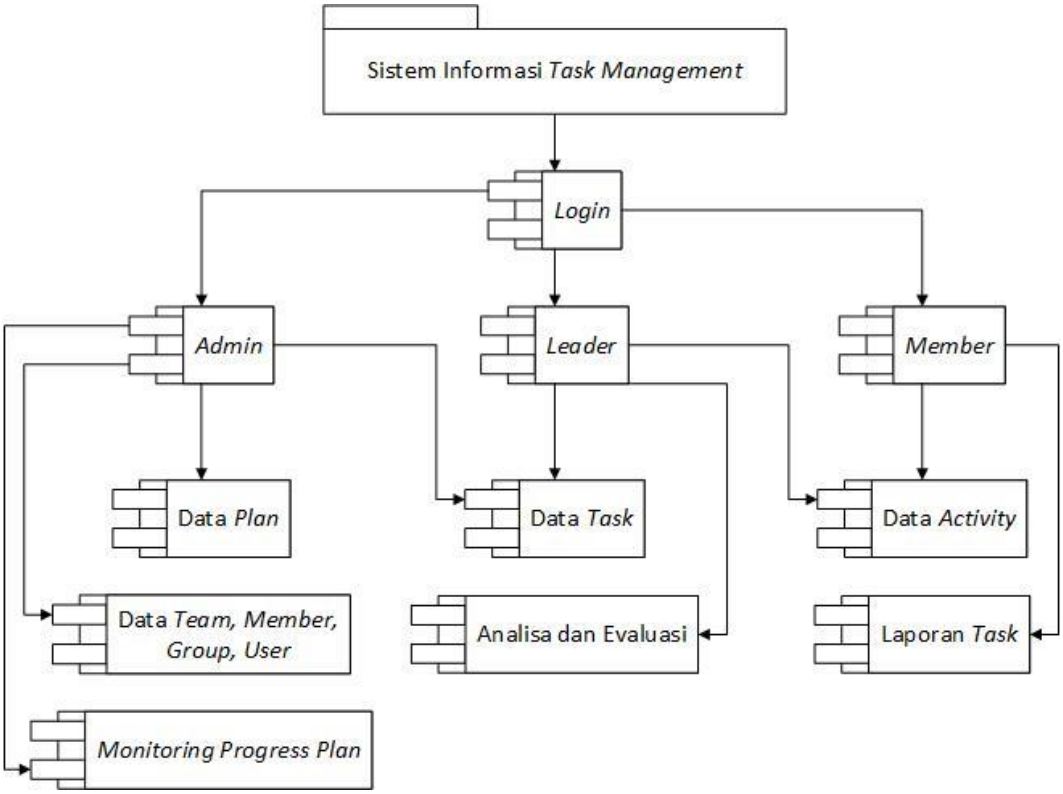
4.1.4.2 *Object Diagram*



Gambar 4.13 Object Diagram

4.1.4.3 *Component Diagram*

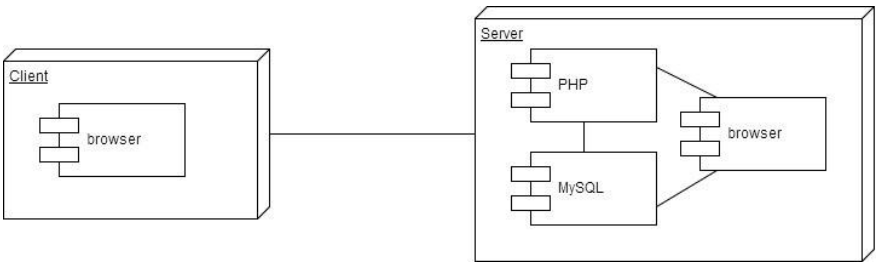
Component diagram dibuat untuk menunjukan organisasi dan ketergantunggan diantara kumpulan komponen dalam sebuah sistem. Com*ponent diagram* fokus pada komponen sistem yang dibutuhkan dan ada di dalam sistem.



Gambar 4.16. *Component Diagram*

4.1.4.4 *Deployment Diagram*

Deployment View berhubungan dengan penyebaran fisik (*deployment*) aplikasi ke komputer-komputer yang sesuai. Berikut adalah *Deployment View* yang diusulkan pada sistem informasi *Task Management* berbasis web di PT eBdesk Teknologi.



Gambar 4.14 Deployment Diagram

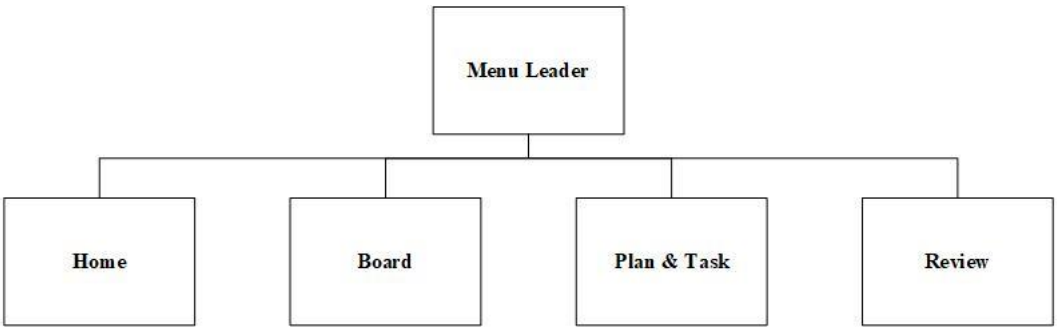
4.2 Perancangan Antar Muka

Perancangan tampilan layar ini bertujuan untuk memberikan interface tentang desain program yang akan dibuat. Dibawah ini terdapat desain template pada tampilan website yang akan dibuat.

4.2.1 Struktur Menu

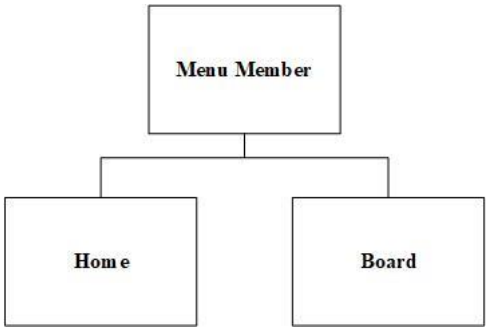
Struktur menu pada system yang diusulkan dibagi 3 yaitu struktur menu untuk *leader*, *member*, dan admin. Struktur tersebut antara lain:

1) Struktur Menu Halaman *Leader*



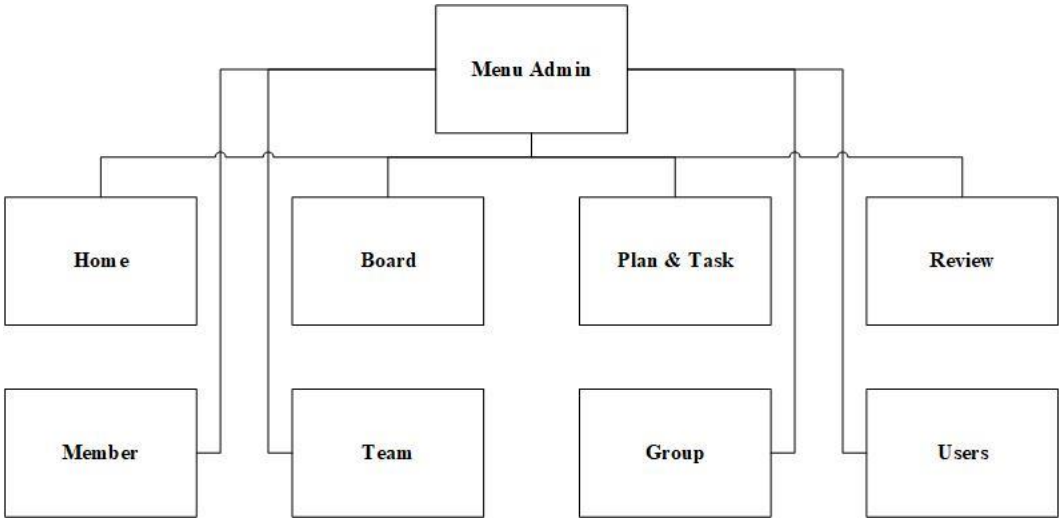
Gambar 4.15 Struktur Menu Halaman Leader

2) Struktur Menu Halaman *Member*



Gambar 4.16 Struktur Menu Halaman Member

3) Struktur Menu Halaman *Admin*



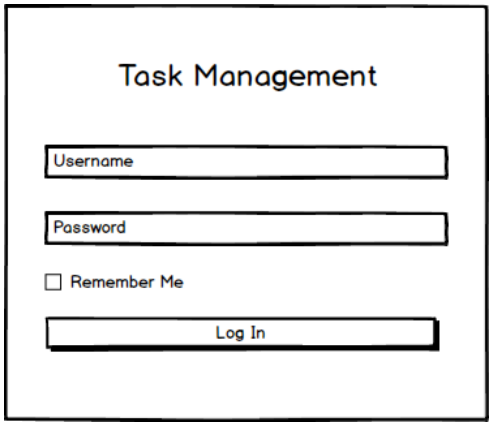
Gambar 4.17 Struktur Menu Halaman Admin

4.2.2 Perancangan *Input*

Perancangan *input* merupakan dimulainya suatu proses informasi. Dalam perancangan *input* ini, data yang dimasukkan akan mempengaruhi hasil yang ditampilkan. Perancangan - perancangan *input* yang ada dalam perancangan ini yaitu :

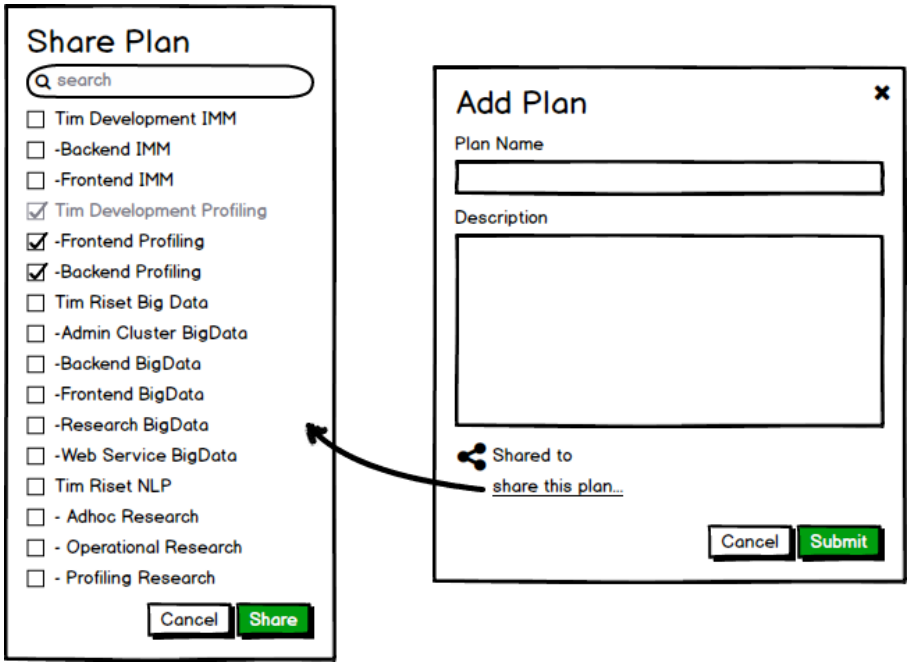
1) Perancangan Tampilan *Login*

Tampilan *Login* ini berlaku untuk semua aktor, baik leader, member maupun admin. Berikut perancangan tampilan login :



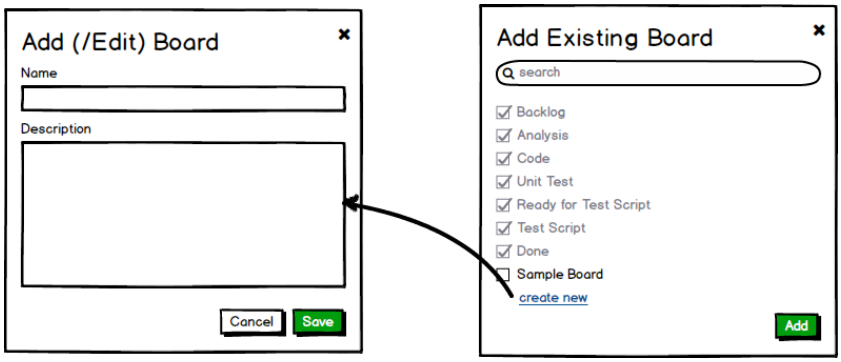
Gambar 4.18 Perancangan Tampilan Login

2) Perancangan Tampilan *Input* Data *Plan*



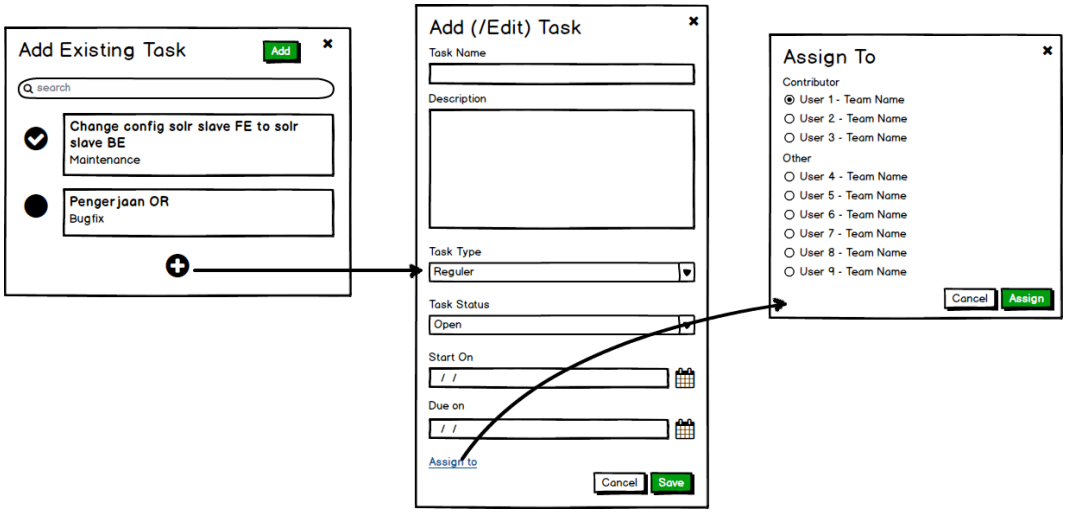
Gambar 4.19 Perancangan Tampilan Input Data Plan

3) Perancangan Tampilan *Input* Data *Board*



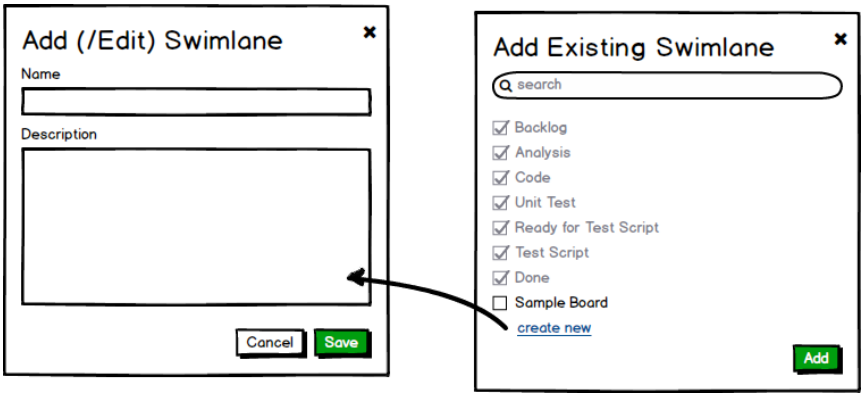
Gambar 4.20 Perancangan Tampilan Input Data Board

4) Perancangan Tampilan *Input* Data *Task*



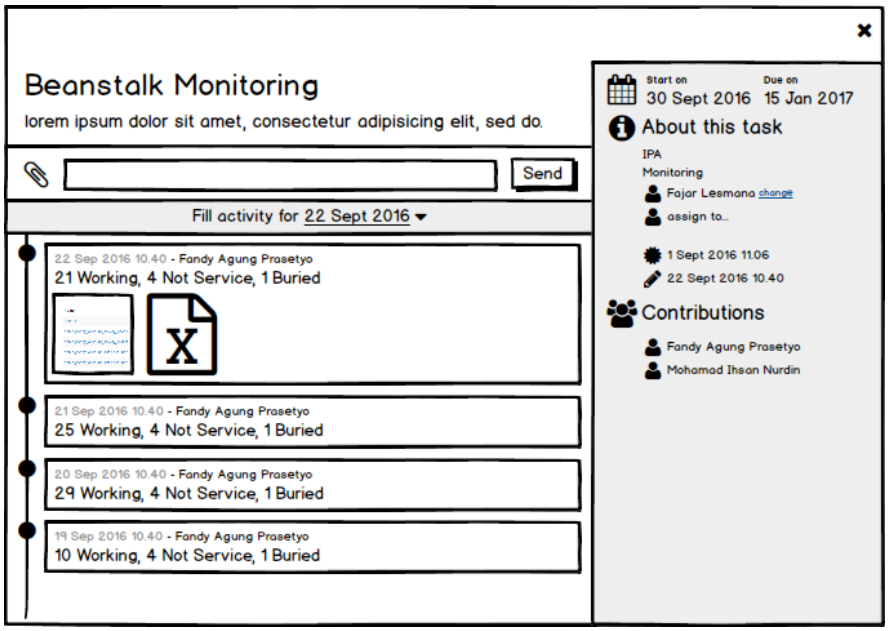
Gambar 4.21 Perancangan Tampilan Input Data Task

5) Perancangan Tampilan *Input* Data *Swimlane*



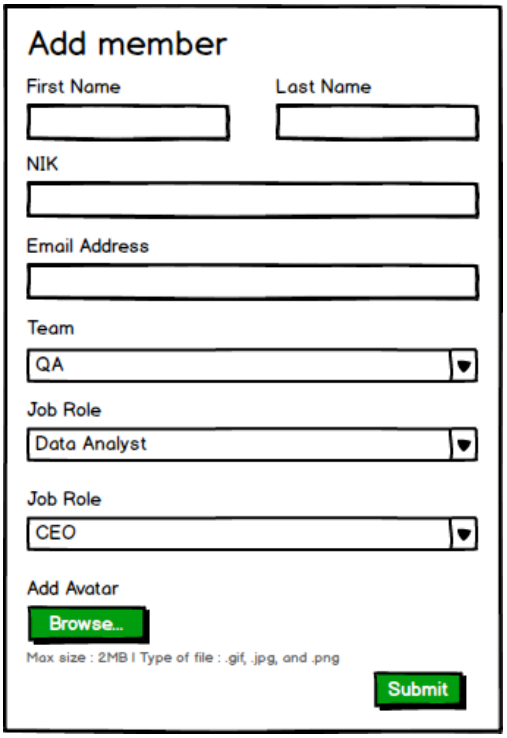
Gambar 4.22 Perancangan Tampilan Input Data Swimlane

6) Perancangan Tampilan *Input* Data *Activity*



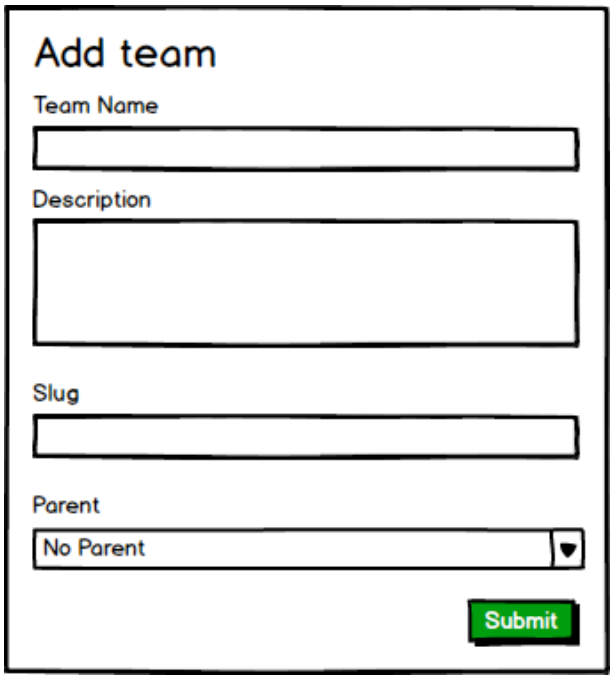
Gambar 4.23 Perancangan Tampilan Input Data Activity

7) Perancangan Tampilan *Input* Data *Member*



Gambar 4.24 Perancangan Tampilan Input Data Member

8) Perancangan Tampilan *Input* Data *Team*



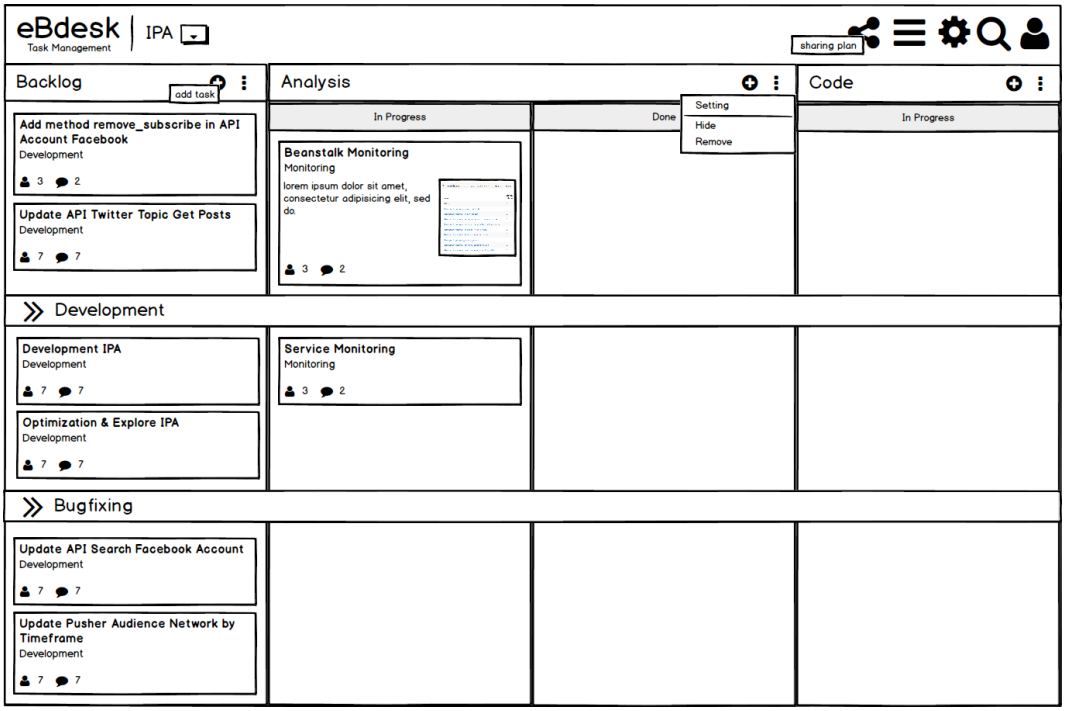
Gambar 4.25 Perancangan Tampilan Input Data Team

4.2.3 Perancangan *Output*

Perancangan *output* merupakan gambaran yang menunjukan hasil dari eksekusi data atau perintah yang telah dimasukan ke dalam sistem. Dimana data yang masih mentah pada saat proses input akan ditampilkan pada proses *output* dalam bentuk informasi. Dan perancangan *output* pada sistem yang diusulkan adalah sebagai berikut:

1) Perancangan Tampilan *Output Monitoring Progress Plan*

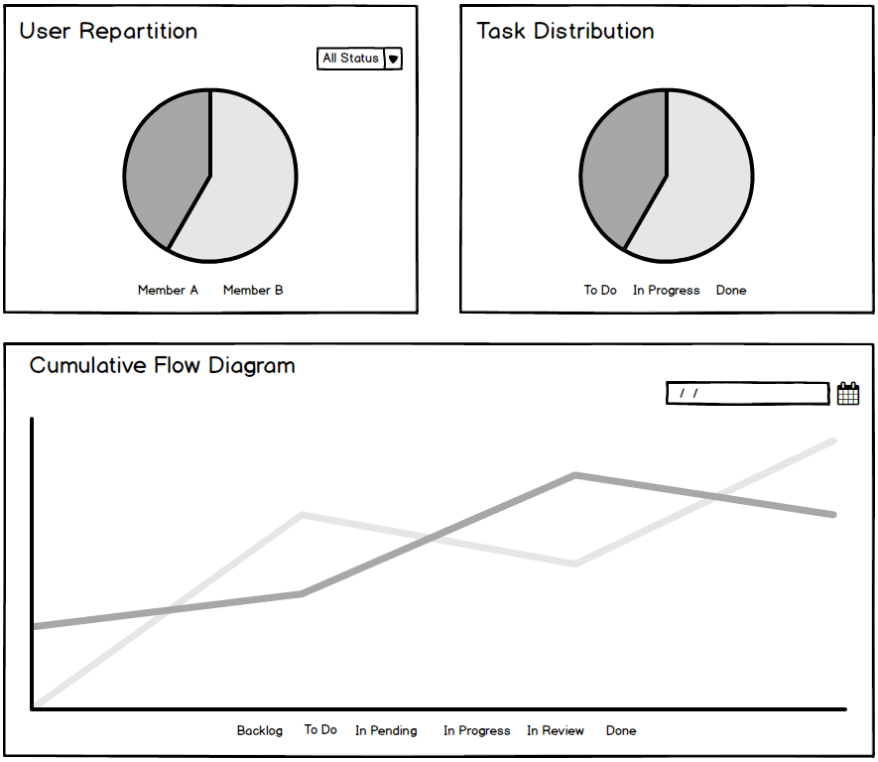
Tampilan *output monitoring progress plan* adalah tampilan dimana pengguna dapat memantau *progress* dari suatu *plan* yang sedang dikerjakan. Dalam tampilan ini pengguna dapat melihat task apa saja yang harus, sedang dan sudah dikerjakan. Selain itu pengguna juga dapat meilhat siapa saja yg terlibat dalam *plan* tersebut dan apa saja aktifitas dari orang-orang yang terlibat tersebut.



Gambar 4.26 2Perancangan Tampilan Output Monitoring Progress Plan

2) Perancangan Tampilan *Output* Analisa dan Evaluasi

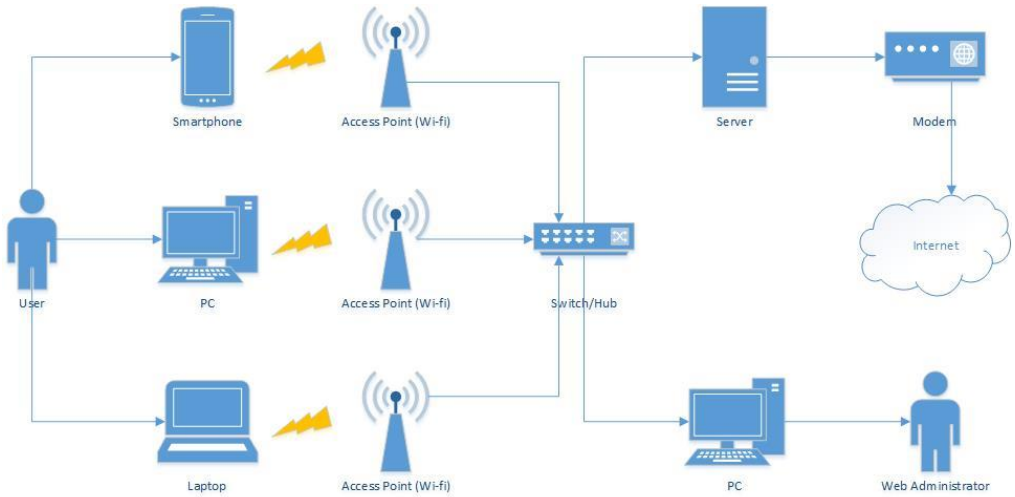
Tampilan *Output* Analisa & Evaluasi ini berupa grafik *User Repartition*, *Task Distribution* dan *Cumulative Flow Diagram* yang mana digunakan untuk menganalisa apa aksi selanjutnya yang harus dilakukan untuk menyelesaikan suatu *plan* yang sedang dikerjakan secara efektif dan efisien.



Gambar 4.27 Perancangan Tampilan Output Monitoring Progress Plan

4.3 Perancangan Arsitektur Jaringan

Arsitektur jaringan merupakan pemilihan topologi jaringan dan protokol yang tepat agar kinerja perangkat lunak yang dibangun berjalan maksimal. Pada aplikasi *dekstop* arsitektur jaringan sangat mempengaruhi kinerja perangkat lunak. Sedangkan untuk aplikasi berbasis *website*, arsitektur jaringan merupakan sesuatu yang optional karena *website* dapat dijalankan selama memiliki koneksi internet dan internet *browser*.



Gambar 4.28 3Perancangan Arsitektur Jaringan

4.4 Pengujian

Pengujian merupakan bagian penting dalam siklus pengembangan perangkat lunak. Pengujian dilakukan untuk menjamin kualitas dan juga mengetahui kelemahan dari perangkat lunak. Tujuan dari pengujian ini adalah untuk menjamin bahwa perangkat lunak yang dibangun memiliki kualitas yang baik. Berikut ini adalah rencana pengujian *input* atau *output* aplikasi sistem informasinya.

4.4.1 Rencana Pengujian

Pengujian terhadap program itu sendiri bertujuan agar program dapat berjalan dengan baik tanpa mengalami gangguan atau *error*, dan memungkinkan untuk dilakukannya pengembangan sistem lebih lanjut. Pengujian perangkat lunak ini menggunakan metode pengujian *Black Box*. Berikut adalah rencana pengujian *input* atau *output* aplikasi sistem informasinya :

Tabel 4.8. Rencana Pengujian

Item Pengujian	Butir Uji	Jenis Pengujian
Login	Verifikasi username	Blackbox
	Verifikasi password	Blackbox
Pengolahan Data Master	Tambah data	Blackbox
	Edit data	Blackbox
	Hapus data	Blackbox
	Cari data	Blackbox
	Lihat data	Blackbox
Pengolahan Board	Lihat Board	Blackbox
	Tambah board	Blackbox
	Edit board	Blackbox
	Sembunyi board	Blackbox
	Urutkan board	Blackbox
	Tambah / Hapus status board	Blackbox
Pengolahan Swimlane	Tambah swimlane	Blackbox
	Edit swimlane	Blackbox
	Sembunyi swimlane	Blackbox
	Urutkan swimlane	Blackbox
Pengolahan Task	Tambah task	Blackbox
	Edit task	Blackbox

	Tambah activity	Blackbox
Pengolahan Plan	Lihat plan	Blackbox
	Tambah plan	Blackbox
	Share plan	Blackbox
Unduh Laporan	Unduh laporan	Blackbox
Analisa & Evaluasi	Lihat grafik	Blackbox
	Lihat aktifitas	Blackbox

4.4.2 Kasus dan Hasil Pengujian

Pengujian Sistem Informasi Task Management di bawah ini menggunakan data uji berupa sebuah data masukan. Berikut ini adalah beberapa pengujian yang telah dilakukan, yaitu :

1. **Pengujian *Login***

Tabel 4.9. Pengujian *Login*

Kasus dan Hasil Uji (Data Normal)			
Data Masuk	**Yang diharapkan**	**Pengamatan**	**Kesimpulan**
Username dan *Password* yang dimasukan benar	*Login* sukses dan masuk ke halaman utama sesuai dengan hak akses	*Login* sukses dan masuk ke halaman utama sesuai dengan hak akses	[X] Diterima [ ] Ditolak
Kasus dan Hasil Uji (Data Salah)			

Data Masuk	Yang diharapkan	Pengamatan	Kesimpulan
Username atau *Password* yang dimasukan salah	*Login* gagal serta menampilkan pesan *username* atau *password* salah	*Login* gagal serta menampilkan pesan *username* atau *password* salah	[ ] Diterima [X] Ditolak

2. Pengujian Pengolahan Data Master

Tabel 4.111. Pengujian Pengelolaan Data Master

Kasus dan Hasil Uji (Data Normal)			
Data Masuk	**Yang diharapkan**	**Pengamatan**	**Kesimpulan**
Klik menu yang ada	Tampil halaman yang telah dipilih	Halaman dan hasil data-data tampil sesuai dengan yang diharapkan	[X] Diterima [ ] Ditolak
Pilih tombol tambah	Tampil form tambah data	Dapat menambah ke dalam *database*	[X] Diterima [ ] Ditolak
Pilih tombol simpan	Data tersimpan ke *database*	Dapat menyimpan ke dalam *database*	[X] Diterima [ ] Ditolak

Pilih tombol edit data	Tampil form edit data	Dapat merubah dan menyimpan ke dalam *database*	[X] Diterima [ ] Ditolak
Pilih tombol hapus data	Tampil pesan penghapusan data berhasil dihapus	Dapat menghapus dan menyimpan ke dalam *database*	[X] Diterima [ ] Ditolak
Masukan kata kunci pencarian	Tampil data pencarian	Dapat mencari data dari *database*	[X] Diterima [ ] Ditolak
Kasus dan Hasil Uji (Data Salah)			
Salah satu *field* dokosongkan	Tampil pesan "harap isi bidang ini" pada *field* yang kosong	Tampil pesan "harap isi bidang ini" pada *field* yang kosong	[ ] Diterima [X] Ditolak

3. Pengujian Pengolahan *Board*

Tabel 4.212. Pengujian Pengelolaan Board

Kasus dan Hasil Uji (Data Normal)			
Data Masuk	**Yang diharapkan**	**Pengamatan**	**Kesimpulan**

Klik salah satu plan yang ada	Tampil halaman *board* dengan plan yang dipilih	Halaman dan hasil data-data tampil sesuai dengan yang diharapkan	[X] Diterima [ ] Ditolak
Pilih tombol kelola *board* ()	Tampil (*pop up*) kelola *board*	Halaman dan hasil data-data tampil sesuai dengan yang diharapkan	[X] Diterima [ ] Ditolak
Pilih tombol Tambah *Board* ()	Tampil form tambah *board*	Dapat mengisi form tambah *board*	[X] Diterima [ ] Ditolak
Pilih tombol *save*	Data tersimpan ke *database*	Dapat menyimpan data ke dalam *database*	[X] Diterima [ ] Ditolak
Klik tombol opsi () pada salah satu *board* lalu pilih opsi "*Setting*"	Tampil form *edit board*	Dapat meng-*edit* data dan tersimpan ke dalam *database*	[X] Diterima [ ] Ditolak
Klik tombol opsi () pada salah satu *board* lalu pilih opsi "*Hide*"	Tampilan *board* yang disembunyikan menghilang	Dapat meng-*edit* data dan tersimpan ke dalam *database*	[X] Diterima [ ] Ditolak

Pilih tombol kelola *board* ()	Tampil (pop up) kelola *board*	Halaman dan hasil data-data tampil sesuai dengan yang diharapkan	[X] Diterima [ ] Ditolak
Urutkan *board* dengan menyeret *board* () ke posisi yang diinginkan	Board pindah ke posisi yang diinginkan	*Board* pindah ke posisi yang diinginkan	[X] Diterima [ ] Ditolak
Pilih tombol *save*	Data tersimpan ke *database*	Dapat meng-*edit* data dan tersimpan ke dalam *database*	[X] Diterima [ ] Ditolak
Pilih tombol kelola *board* ()	Tampil (*pop up*) kelola *board*	Halaman dan hasil data-data tampil sesuai dengan yang diharapkan	[X] Diterima [ ] Ditolak
Centang status yang diinginkan pada masing-masing *board*	Status yang diinginkan tercentang	Status yang diinginkan tercentang	[X] Diterima [ ] Ditolak

Pilih tombol *save*	Data tersimpan ke *database*	Dapat meng-*edit* data dan tersimpan ke dalam *database*	[X] Diterima [ ] Ditolak
Kasus dan Hasil Uji (Data Salah)			
Salah satu *field* dokosongkan	Tampil pesan "harap isi bidang ini" pada *field* yang kosong	Tampil pesan "harap isi bidang ini" pada *field* yang kosong	[ ] Diterima [X] Ditolak

4. Pengujian Pengolahan *Swimlane*

Tabel 4.313. Pengujian Pengelolaan Swimlane

Kasus dan Hasil Uji (Data Normal)			
Data Masuk	**Yang diharapkan**	**Pengamatan**	**Kesimpulan**
Klik salah satu plan yang ada	Tampil halaman *board* sesuai dengan *plan* yang dipilih	Halaman dan hasil data-data tampil sesuai dengan yang diharapkan	[X] Diterima [ ] Ditolak
Pilih tombol kelola *swimlane* ()	Tampil (*pop up*) kelola *swimlane*	Halaman dan hasil data-data tampil sesuai dengan yang diharapkan	[X] Diterima [ ] Ditolak

Pilih tombol Tambah *Swimlane* (+)	Tampil *form* tambah *swimlane*	Dapat mengisi form tambah *swimlane*	[X] Diterima [ ] Ditolak
Pilih tombol *save*	Data tersimpan ke *database*	Dapat menyimpan data ke dalam *database*	[X] Diterima [ ] Ditolak
Klik tombol *edit swimlane* ( )	Tampil form *edit swimlane*	Dapat meng-*edit* data dan tersimpan ke dalam *database*	[X] Diterima [ ] Ditolak
Klik tombol *hide swimlane* ( )	Tampilan *swimlane* yang disembunyikan menghilang	Dapat meng-*edit* data dan tersimpan ke dalam *database*	[X] Diterima [ ] Ditolak
Pilih tombol kelola *swimlane* ( )	Tampil (*pop up*) kelola *swimlane*	Halaman dan hasil data-data tampil sesuai dengan yang diharapkan	[X] Diterima [ ] Ditolak
Urutkan *swimlane* dengan menyeret board ( ) ke posisi yang diinginkan	*Swimlane* pindah ke posisi yang diinginkan	*Swimlane* pindah ke posisi yang diinginkan	[X] Diterima [ ] Ditolak

Pilih tombol *save*	Data tersimpan ke *database*	Dapat meng-*edit* data dan tersimpan ke dalam *database*	[X] Diterima [ ] Ditolak
Kasus dan Hasil Uji (Data Salah)			
Salah satu *field* dokosongkan	Tampil pesan "harap isi bidang ini" pada *field* yang kosong	Tampil pesan "harap isi bidang ini" pada *field* yang kosong	[ ] Diterima [X] Ditolak

5. Pengujian Pengolahan *Task*

Tabel 4.414. Pengujian Pengelolaan Task

Kasus dan Hasil Uji (Data Normal)			
Data Masuk	**Yang diharapkan**	**Pengamatan**	**Kesimpulan**
Klik salah satu *plan* yang ada	Tampil halaman *board* dengan plan yang dipilih	Halaman dan hasil data-data tampil sesuai dengan yang diharapkan	[X] Diterima [ ] Ditolak
Pilih tombol Tambah *Task* (+) disalah satu *board*	Tampil *form* tambah *task*	Dapat mengisi form tambah *task*	[X] Diterima [ ] Ditolak

Pilih tombol *save*	Data tersimpan ke *database*	Dapat menyimpan data ke dalam *database*	[X] Diterima [ ] Ditolak
Klik salah satu task yang akan di-*edit*	Tampil *form edit task*	Dapat meng-*edit* data dan tersimpan ke dalam *database* ketika data dirubah	[X] Diterima [ ] Ditolak
Klik salah satu *task* yang akan diisi *activity*	Tampil form *edit activity*	Dapat mengisi form tambah *activity*	[X] Diterima [ ] Ditolak
Pilih tombol *send*	Data tersimpan ke *database*	Dapat menyimpan data ke dalam *database*	[X] Diterima [ ] Ditolak
Kasus dan Hasil Uji (Data Salah)			
Salah satu *field* dokosongkan	Tampil pesan "harap isi bidang ini" pada *field* yang kosong	Tampil pesan "harap isi bidang ini" pada *field* yang kosong	[ ] Diterima [X] Ditolak

6. Pengujian Pengolahan *Plan*

Tabel 4.515. Pengujian Pengolahan Plan

Kasus dan Hasil Uji (Data Normal)			
Data Masuk	**Yang diharapkan**	**Pengamatan**	**Kesimpulan**
Pilih menu *Home*	Tampil halaman home data list *plan* di panel sebalah kiri	Halaman dan hasil data-data tampil sesuai dengan yang diharapkan	[X] Diterima [ ] Ditolak
Klik tombol tambah *plan* ( )	Tampil (*pop up*) form tambah *plan*	Halaman dan hasil data-data tampil sesuai dengan yang diharapkan	[X] Diterima [ ] Ditolak
Pilih tombol *save*	Data tersimpan ke *database*	Dapat menyimpan data ke dalam *database* dan memuat ulang list plan	[X] Diterima [ ] Ditolak
Pilih salah satu *plan*	Tampil halaman board dengan *plan* yang dipilih	Halaman dan hasil data-data tampil sesuai dengan yang diharapkan	[X] Diterima [ ] Ditolak

Klik tombol *share plan* ()	Tampil *form share plan*	Dapat mengisi *form share plan*	[X] Diterima [ ] Ditolak
Pilih tombol *save*	Data tersimpan ke *database*	Dapat menyimpan data ke dalam *database*	[X] Diterima [ ] Ditolak
Kasus dan Hasil Uji (Data Salah)			
Salah satu *field* dokosongkan	Tampil pesan "harap isi bidang ini" pada *field* yang kosong	Tampil pesan "harap isi bidang ini" pada *field* yang kosong	[ ] Diterima [X] Ditolak

7. Unduh Laporan

Tabel 4.616. Pengujian Unduh Laporan

Kasus dan Hasil Uji (Data Normal)			
Data Masuk	**Yang diharapkan**	**Pengamatan**	**Kesimpulan**
Klik tombol unduh laporan ()	Tampil (*pop up*) *form* unduh laporan	Halaman dan hasil data-data tampil sesuai dengan yang diharapkan	[X] Diterima [ ] Ditolak

Pilih tombol *preview*	Tampil (*pop up*) data laporan	Halaman dan hasil data-data tampil sesuai dengan yang diharapkan	[X] Diterima [ ] Ditolak
Pilih tombol *download*	Sistem mengunduh data	Dapat mengunduh data yang tersimpan di dalam *database*	[X] Diterima [ ] Ditolak
Kasus dan Hasil Uji (Data Salah)			
Salah satu *field* dokosongkan	Tampil pesan "harap isi bidang ini" pada *field* yang kosong	Tampil pesan "harap isi bidang ini" pada *field* yang kosong	[ ] Diterima [X] Ditolak

8. Analisa & Evaluasi

Tabel 4.717. Pengujian Analisa dan Evaluasi

Kasus dan Hasil Uji (Data Normal)			
Data Masuk	**Yang diharapkan**	**Pengamatan**	**Kesimpulan**
Klik salah satu *plan* yang ada	Tampil halaman *board* sesuai dengan *plan* yang dipilih	Halaman dan hasil data-data tampil sesuai dengan yang diharapkan	[X] Diterima [ ] Ditolak

Pilih tombol *analytics* ()	Tampil halaman *Analytics*	Halaman dan hasil data-data tampil sesuai dengan yang diharapkan	[X] Diterima [ ] Ditolak
Pilih menu *review*	Tampil halaman *Review*	Halaman dan hasil data-data tampil sesuai dengan yang diharapkan	[X] Diterima [ ] Ditolak
Kasus dan Hasil Uji (Data Salah)			
Salah satu *field* dokosongkan	Tampil pesan "harap isi bidang ini" pada *field* yang kosong	Tampil pesan "harap isi bidang ini" pada *field* yang kosong	[ ] Diterima [X] Ditolak

4.4.3 Kesimpulan Hasil Pengujian

Berdasarkan hasil tahapan pengujian yang dilakukan, sistem yang diusulkan telah memenuhi standar. Pembangunan perangkat lunak yang diusulkan sudah layak untuk dipergunakan dan dari modul serta fungsionalnya mendapatkan hasil yang sesuai dengan yang diharapkan.

4.5 Implementasi

Tahap implementasi merupakan realisasi sistem berdasarkan pada desain yang telah dibuat pada tahap perancangan sistem. Dibawah ini akan dijelaskan

mengenai batasan implementasi (optimal), implementasi perangkat lunak, implementasi perangkat keras, implementasi basis data, implementasi antar muka, implementasi instalasi program, dan penggunaan program.

4.5.1 Implementasi Perangkat Lunak

Adapun implementasi perangkat lunak yang digunakan untuk merancang dan membangun website *Task Management* di PT eBdesk Teknologi ini, diantaranya :

Tabel 4.18. Implementasi Perangkat Lunak

1.	Operating System	Windows 10 Pro Language 86x/32-Bit
2.	Php Editor	Sublime Text 3
3.	Web Server	XAMPP v5.6.30
4.	Database Server	MySQL
5.	Web Browser	Google Chrome v59.0.2785.143 m

4.5.2 Implementasi Perangkat Keras

Perangkat keras yang digunakan dalam pembuatan aplikasi sistem informasi *Task Management* ini sangat menentukan kecepatan dan ketetapan dalam proses penyimpanan data. Adapun perangkat keras yang dibutuhkan berdasarkan kebutuhan minimal yang harus dipenuhi antara lain :

1. Processor : Intel Core i3 atau setaranya.
2. Memory : 2 GB

3. Graphic Display (VGA) : 2GB
4. Hardisk : 500 GB
5. Alat Pendukung : Monitor, Mouse, Keyboard.

4.5.3 Implementasi Basis Data

Implementasi basis data dilakukan dengan menggunakan aplikasi pemrograman *MySQL*, implementasi basis datanya dalam bahasa *SQL* adalah sebagai berikut :

1) Struktur dari Tabel Activity

```
CREATE TABLE IF NOT EXISTS `activity` (
 `id` int(11) NOT NULL AUTO_INCREMENT,
 `description` text,
 `created_at` datetime DEFAULT NULL,
 `updated_at` datetime DEFAULT NULL,
 `deleted_at` datetime DEFAULT NULL,
 `task_id` int(11) DEFAULT NULL,
 `nik` int(11) DEFAULT NULL,
 PRIMARY KEY (`id`),
 KEY `fk_progress_1_idx` (`task_id`),
 KEY `fk_progress_2_idx` (`nik`),
 KEY `created_at` (`created_at`),
 KEY `created_at_2` (`created_at`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 ;
```

2) Struktur dari Tabel Activity File

```
CREATE TABLE IF NOT EXISTS `activity_file` (
  `activity_id` int(11) NOT NULL,
  `file_id` int(11) NOT NULL,
  PRIMARY KEY (`activity_id`,`file_id`),
  KEY `fk_progress_file_2_idx` (`file_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

3) Struktur dari Tabel Board

```
CREATE TABLE IF NOT EXISTS `board` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `name` varchar(100) DEFAULT NULL,
  `description` text,
  `created_at` datetime DEFAULT NULL,
  `updated_at` datetime DEFAULT NULL,
  `deleted_at` datetime DEFAULT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB  DEFAULT CHARSET=utf8 ;
```

4) Struktur dari Tabel Board Cumulative Flow

```
CREATE TABLE IF NOT EXISTS `board_cumulative_flow` (
  `plan_id` int(11) NOT NULL,
  `board_id` int(11) NOT NULL,
  `total` int(11) DEFAULT NULL,
  `d_day` char(8) NOT NULL,
  `d_month` char(6) DEFAULT NULL,
```

```
  `d_year` char(4) DEFAULT NULL,
  PRIMARY KEY (`plan_id`,`board_id`,`d_day`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

5) Struktur dari Tabel Board Task Status

```
CREATE TABLE IF NOT EXISTS `board_task_status` (
  `board_id` int(11) NOT NULL,
  `task_status_id` int(11) NOT NULL,
  PRIMARY KEY (`board_id`,`task_status_id`),
  KEY `task_status_id` (`task_status_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

6) Struktur dari Tabel File

```
CREATE TABLE IF NOT EXISTS `file` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `name` varchar(100) DEFAULT NULL,
  `description` text,
  `path` text,
  `fullpath` text,
  `extension` varchar(20) DEFAULT NULL,
  `size` int(11) DEFAULT NULL,
  `created_at` datetime DEFAULT NULL,
  `updated_at` datetime DEFAULT NULL,
  `deleted_at` datetime DEFAULT NULL,
  `nik` int(11) DEFAULT NULL,
```

```
  PRIMARY KEY (`id`),
  KEY `index2` (`nik`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 ;
```

7) Struktur dari Tabel Groups

```
CREATE TABLE IF NOT EXISTS `groups` (
  `id` mediumint(8) unsigned NOT NULL AUTO_INCREMENT,
  `name` varchar(20) NOT NULL,
  `description` varchar(100) NOT NULL,
  `date` datetime DEFAULT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB  DEFAULT CHARSET=utf8 ;
```

8) Struktur dari Tabel Job Role

```
CREATE TABLE IF NOT EXISTS `job_role` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `name` varchar(100) DEFAULT NULL,
  `description` text,
  `parent` int(11) DEFAULT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB  DEFAULT CHARSET=utf8 ;
```

9) Struktur dari Tabel Job Title

```
CREATE TABLE IF NOT EXISTS `job_title` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `name` varchar(100) DEFAULT NULL,
```

```
  `decription` text,
  `parent` int(11) DEFAULT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB  DEFAULT CHARSET=utf8 ;
```

10) Struktur dari Tabel Member

```
CREATE TABLE IF NOT EXISTS `member` (
  `nik` int(11) NOT NULL,
  `users_id` int(11) DEFAULT NULL,
  `name` varchar(100) DEFAULT NULL,
  `email` varchar(100) DEFAULT NULL,
  `avatar` text,
  `team_id` int(11) DEFAULT NULL,
  `job_title_id` int(11) DEFAULT NULL,
  `job_role_id` int(11) DEFAULT NULL,
  `created_at` datetime DEFAULT NULL,
  `updated_at` datetime DEFAULT NULL,
  `deleted_at` datetime DEFAULT NULL,
  PRIMARY KEY (`nik`) USING BTREE,
  UNIQUE KEY `users_id` (`users_id`),
  KEY `fk_member_1_idx` (`team_id`),
  KEY `fk_member_2_idx` (`job_role_id`),
  KEY `fk_member_3_idx` (`job_title_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

11) Struktur dari Tabel Menu

```
CREATE TABLE IF NOT EXISTS `menu` (
  `id` int(2) unsigned NOT NULL AUTO_INCREMENT,
  `name` varchar(100) DEFAULT NULL,
  `icon` varchar(64) DEFAULT NULL,
  `url` varchar(500) DEFAULT NULL,
  `parent` int(2) NOT NULL DEFAULT '0',
  `sort` int(1) NOT NULL DEFAULT '1',
  `status` int(2) DEFAULT '1' COMMENT '1 : Active | 0 : Inactive',
  `show_in_dashboard` tinyint(4) NOT NULL DEFAULT '0',
  PRIMARY KEY (`id`)
) ENGINE=InnoDB  DEFAULT CHARSET=utf8 ;
```

12) Struktur dari Tabel Menu Groups

```
CREATE TABLE IF NOT EXISTS `menu_groups` (
  `id` int(11) unsigned NOT NULL AUTO_INCREMENT,
  `menu_id` int(11) unsigned NOT NULL,
  `group_id` mediumint(8) unsigned NOT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `uc_users_groups` (`menu_id`,`group_id`),
  KEY `fk_users_groups_users1_idx` (`menu_id`),
  KEY `fk_users_groups_groups1_idx` (`group_id`)
) ENGINE=InnoDB  DEFAULT CHARSET=utf8 ;
```

13) Struktur dari Tabel Plan

```
CREATE TABLE IF NOT EXISTS `plan` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `name` varchar(100) DEFAULT NULL,
  `sequence` int(11) DEFAULT NULL,
  `description` text,
  `created_at` datetime DEFAULT NULL,
  `updated_at` datetime DEFAULT NULL,
  `deleted_at` datetime DEFAULT NULL,
  `team_id` int(11) DEFAULT NULL,
  `plan_type_id` int(11) DEFAULT NULL,
  PRIMARY KEY (`id`),
  KEY `fk_project_1_idx` (`team_id`)
) ENGINE=InnoDB  DEFAULT CHARSET=utf8 ;
```

14) Struktur dari Tabel Plan Board

```
CREATE TABLE IF NOT EXISTS `plan_board` (
  `plan_id` int(11) NOT NULL,
  `board_id` int(11) NOT NULL,
  `sort` int(11) NOT NULL DEFAULT '0',
  `hide` tinyint(1) NOT NULL DEFAULT '0',
  PRIMARY KEY (`plan_id`,`board_id`),
  KEY `fk_plan_board_1_idx` (`board_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

15) Struktur dari Tabel Plan Shared

```
CREATE TABLE IF NOT EXISTS `plan_shared` (
  `plan_id` int(11) NOT NULL,
  `team_id` int(11) NOT NULL,
  PRIMARY KEY (`plan_id`,`team_id`),
  KEY `fk_project_shared_2_idx` (`team_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

16) Struktur dari Tabel Plan Status

```
CREATE TABLE IF NOT EXISTS `plan_status` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `name` varchar(100) DEFAULT NULL,
  `description` text,
  `color` varchar(10) DEFAULT NULL,
  `sort` int(2) DEFAULT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB  DEFAULT CHARSET=utf8 ;
```

17) Struktur dari Tabel Plan Swimlane

```
CREATE TABLE IF NOT EXISTS `plan_swimlane` (
  `plan_id` int(11) NOT NULL,
  `swimlane_id` int(11) NOT NULL,
  `sort` int(11) NOT NULL DEFAULT '0',
  `hide` tinyint(1) NOT NULL DEFAULT '0',
  PRIMARY KEY (`plan_id`,`swimlane_id`),
  KEY `fk_plan_swimlane_1_idx` (`swimlane_id`)
```

```
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

18) Struktur dari Tabel Plan Type

```
CREATE TABLE IF NOT EXISTS `plan_type` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `name` varchar(100) DEFAULT NULL,
  `description` text,
  `sort` int(2) DEFAULT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB  DEFAULT CHARSET=utf8 ;
```

19) Struktur dari Tabel Quotes

```
CREATE TABLE IF NOT EXISTS `quotes` (
  `id` int(10) NOT NULL AUTO_INCREMENT,
  `author` varchar(100) DEFAULT 'Unknown',
  `quotes` text NOT NULL,
  `status` enum('0','1') NOT NULL DEFAULT '1' COMMENT '0 =
disactive, 1 = active',
  PRIMARY KEY (`id`)
) ENGINE=InnoDB  DEFAULT CHARSET=utf8 ;
```

20) Struktur dari Tabel Swimlane

```
CREATE TABLE IF NOT EXISTS `swimlane` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `name` varchar(100) DEFAULT NULL,
  `description` text,
```

```
  `created_at` datetime DEFAULT NULL,
  `updated_at` datetime DEFAULT NULL,
  `deleted_at` datetime DEFAULT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB  DEFAULT CHARSET=utf8 ;
```

21) Struktur dari Tabel Task

```
CREATE TABLE IF NOT EXISTS `task` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `name` varchar(100) DEFAULT NULL,
  `description` text,
  `start_at` date DEFAULT NULL,
  `due_at` datetime DEFAULT NULL,
  `status` int(11) DEFAULT NULL,
  `created_at` datetime DEFAULT NULL,
  `updated_at` datetime DEFAULT NULL,
  `deleted_at` datetime DEFAULT NULL,
  `task_type_id` int(11) DEFAULT NULL,
  `plan_id` int(11) DEFAULT NULL,
  `team_id` int(11) DEFAULT NULL,
  `nik` int(11) DEFAULT NULL,
  `assign` int(11) DEFAULT NULL,
  PRIMARY KEY (`id`),
  KEY `fk_task_1_idx` (`plan_id`),
```

```
  KEY `index3` (`nik`),
  KEY `team_id` (`team_id`),
  KEY `task_type_id` (`task_type_id`),
  KEY `status` (`status`),
  KEY `assign` (`assign`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 ;
```

22) Struktur dari Tabel Task Board

```
CREATE TABLE IF NOT EXISTS `task_board` (
  `task_id` int(11) NOT NULL,
  `board_id` int(11) DEFAULT NULL,
  `swimlane_id` int(11) DEFAULT NULL,
  PRIMARY KEY (`task_id`),
  KEY `index2` (`board_id`),
  KEY `index3` (`swimlane_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

23) Struktur dari Tabel Task Board Log

```
CREATE TABLE IF NOT EXISTS `task_board_log` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `task_id` int(11) DEFAULT NULL,
  `board_id` int(11) DEFAULT NULL,
  `swimlane_id` int(11) DEFAULT NULL,
  `nik` int(11) DEFAULT NULL,
  `created_at` datetime DEFAULT NULL,
```

```
  PRIMARY KEY (`id`),
  KEY `index2` (`task_id`),
  KEY `index3` (`board_id`),
  KEY `index4` (`swimlane_id`),
  KEY `index5` (`nik`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 ;
```

24) Struktur dari Tabel Task Status

```
CREATE TABLE IF NOT EXISTS `task_status` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `name` varchar(100) DEFAULT NULL,
  `description` text,
  `color` varchar(10) DEFAULT NULL,
  `sort` int(2) DEFAULT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB  DEFAULT CHARSET=utf8 ;
```

25) Struktur dari Tabel Task Type

```
CREATE TABLE IF NOT EXISTS `task_type` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `name` varchar(100) DEFAULT NULL,
  `description` text,
  `parent` int(11) NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB  DEFAULT CHARSET=utf8 ;
```

26) Struktur dari Tabel Task Type Status

```
CREATE TABLE IF NOT EXISTS `task_type_status` (
  `task_type_id` int(11) NOT NULL,
  `task_status_id` int(11) NOT NULL,
  PRIMARY KEY (`task_type_id`,`task_status_id`),
  KEY `task_status_id` (`task_status_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_unicode_ci;
```

27) Struktur dari Tabel *Team*

```
CREATE TABLE IF NOT EXISTS `team` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `name` varchar(100) DEFAULT NULL,
  `decription` text,
  `slug` varchar(10) DEFAULT NULL,
  `parent` int(11) DEFAULT '0',
  `created_at` datetime DEFAULT NULL,
  `updated_at` datetime DEFAULT NULL,
  `deleted_at` datetime DEFAULT NULL,
  `active` tinyint(2) NOT NULL DEFAULT '1' COMMENT '0 : In Active |
1 : Active',
  `flag` tinyint(4) DEFAULT NULL COMMENT '0 : Resource Pool | 1 :
Research Group',
  PRIMARY KEY (`id`)
```

```
) ENGINE=InnoDB  DEFAULT CHARSET=utf8;
```

28) Struktur dari Tabel Users

```
CREATE TABLE IF NOT EXISTS `users` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `ip_address` varchar(45) NOT NULL,
  `username` varchar(100) DEFAULT NULL,
  `password` varchar(255) NOT NULL,
  `salt` varchar(255) DEFAULT NULL,
  `email` varchar(100) NOT NULL,
  `activation_code` varchar(40) DEFAULT NULL,
  `forgotten_password_code` varchar(40) DEFAULT NULL,
  `forgotten_password_time` int(11) unsigned DEFAULT NULL,
  `remember_code` varchar(40) DEFAULT NULL,
  `created_on` int(11) unsigned NOT NULL,
  `last_login` int(11) unsigned DEFAULT NULL,
  `active` tinyint(1) unsigned DEFAULT NULL,
  `first_name` varchar(50) DEFAULT NULL,
  `last_name` varchar(50) DEFAULT NULL,
  `company` varchar(100) DEFAULT NULL,
  `phone` varchar(20) DEFAULT NULL,
  `avatar` text,
  PRIMARY KEY (`id`),
  UNIQUE KEY `email` (`email`),
```

```
  UNIQUE KEY `username` (`username`)
) ENGINE=InnoDB  DEFAULT CHARSET=utf8 ;
```

29) Struktur dari Tabel Users Groups

```
CREATE TABLE IF NOT EXISTS `users_groups` (
  `id` int(11) unsigned NOT NULL AUTO_INCREMENT,
  `user_id` int(11) unsigned NOT NULL,
  `group_id` mediumint(8) unsigned NOT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `uc_users_groups` (`user_id`,`group_id`),
  KEY `fk_users_groups_users1_idx` (`user_id`),
  KEY `fk_users_groups_groups1_idx` (`group_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8
COLLATE=utf8_unicode_ci ;
```

4.5.4 Implementasi Antar Muka

Antar muka sebuah perangkat lunak adalah hal yang sangat penting, karena program yang baik adalah program yang dapat menciptakan media yang baik pula antara pengguna dan sistem.

4.5.4.1 *Implementasi Halaman Utama*

Form halaman utama merupakan sentral dari semua halaman form. Form ini berisi menu-menu utama yang dapat mengakses dan menjadi penghubung antara pengguna dengan sistem. Berikut ini implementasi struktur menu pada halaman utama Sistem Informasi *Task Management* PT eBdesk Teknologi :

Tabel 4.19. Kerangka Halaman Utama

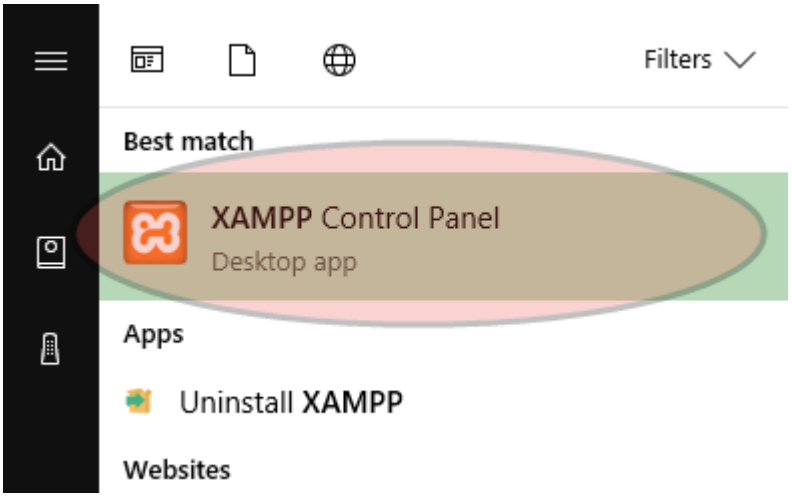
Menu	**Deskripsi**	**Nama Menu**
Login	Menu untuk masuk ke dalam sistem sesuai dengan kategori *user* masing-masing	/security/index
Home	Halaman utama yang berisi aktivitas terakhir dari *user* yang sedang *login*	/home/index
Board	Halaman utama yang berfungsi sebagai *main stage* distribusi dan pelaporan pekerjaan.	/board/index
Analytics	Halaman dimana ditampilkan visualisasi *board* dalam bentuk grafik sebagai acuan analisa.	/board/analytics
Review	Halaman dimana ditampilkan aktivitas *member*.	/home/review

Plan & Task	Halaman dimana pengelolaan data master *Plan* & *Task*.	/home/managerial
Team	Halaman dimana pengelolaan data master *team*.	/home/team
Member	Halaman dimana pengelolaan data master *member*.	/home/employee
Group	Halaman dimana pengelolaan data master *group*.	/group
User	Halaman dimana pengelolaan data master *user*.	/user

4.5.5 Implementasi Instalasi Program

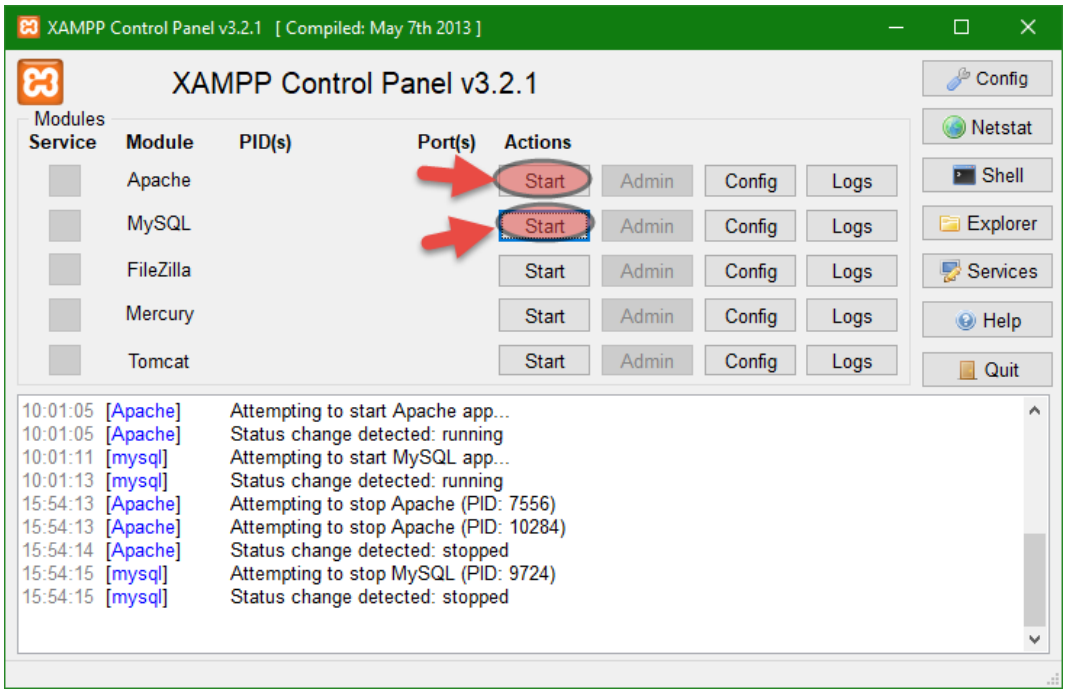
Berikut merupakan implementasi instalasi program bagaimana cara menggunakan program Aplikasi Sistem Informasi *Task Management* di PT eBdesk Teknologi. Langkah-langkahnya adalah sebagai berikut:

a) Setelah terlebih dahulu menginstal Xampp lalu pilih menu pencarian pada komputer “XAMPP”, atau pilih icon “Xampp Control” yang tersedia di desktop komputer.



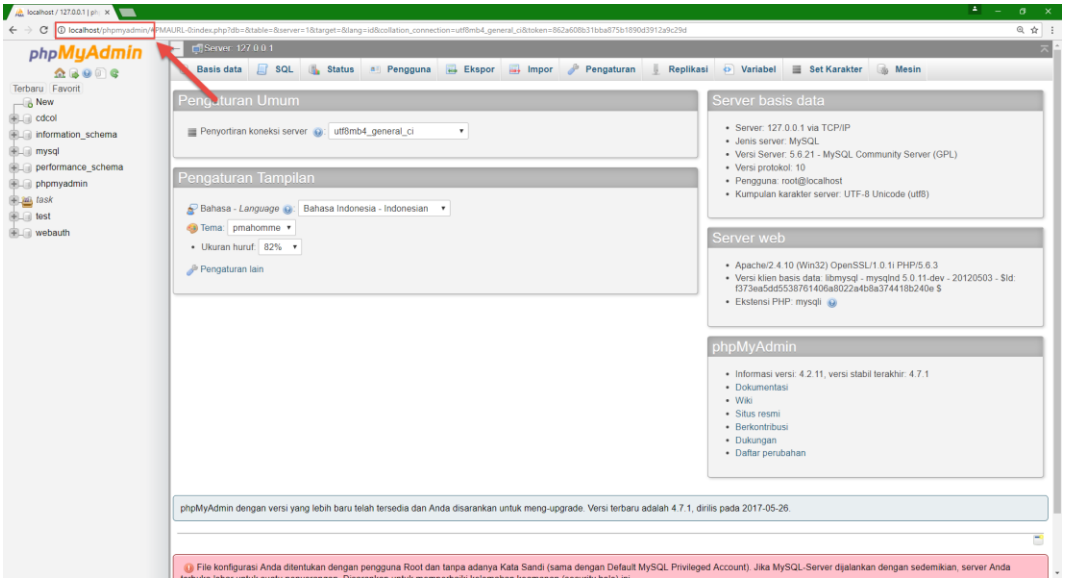
Gambar 4.29 Icon Xampp Control Panel

b) Setelah tampil xampp control panel klik tombol start pada *Apache* dan *MySQL*.



Gambar 4.30 Xampp Control Panel

c) Pilih Google Chrome, lalu masukan alamat *https://localhost/phpmyadmin/* pada URL.



Gambar 4.31 Phpmyadmin

Maka akan tampil halaman Phpmyadmin yang berguna untuk mengelola *database* seperti gambar diatas.

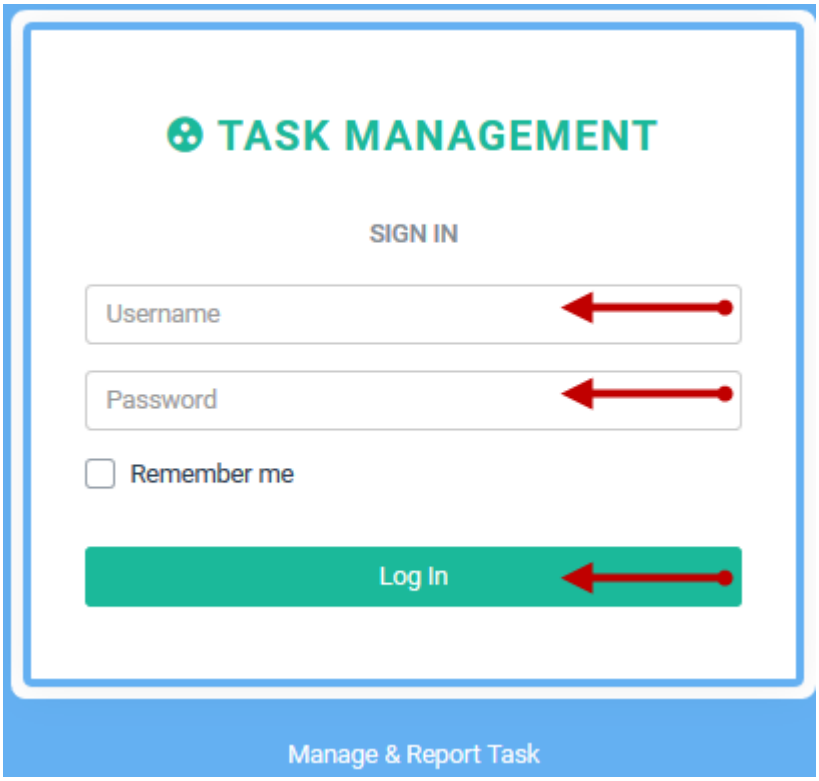
4.5.6 Penggunaan Program

Penggunaan program bertujuan untuk memberikan informasi tentang bagaimana tata cara dalam mengelola program Sistem Informasi *Task Management* di PT eBdesk Teknologi. Berikut adalah petunjuk penggunaan program aplikasi *Task Management* di PT eBdesk Teknologi :

1) Halaman *Login*

Untuk masuk kedalam aplikasi *Task Management* pengguna harus melakukan *login* terlebih dahulu dengan meng-*input*-kan *username* dan *password* masing-masing.

Login ke IP *ebtask.com* dengan ***user*** dan ***password*** yg sudah disediakan oleh administrator.

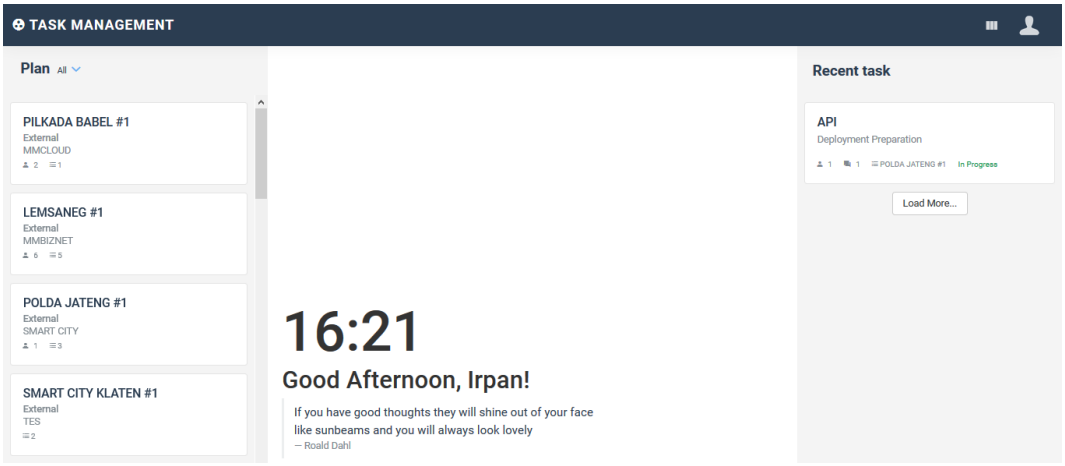


Gambar 4.32 Halaman Login

2) Halaman *Home*

Di halaman *home* ini pengguna dapat melihat *list plan* yang sedang dikerjakan (sebelah kiri) dan *list task* yang terakhir dikerjakan (sebalah kanan).

Jika login sudah benar, tampilan akan diarahkan ke halaman *Dashboard* adalah halaman seperti berikut :

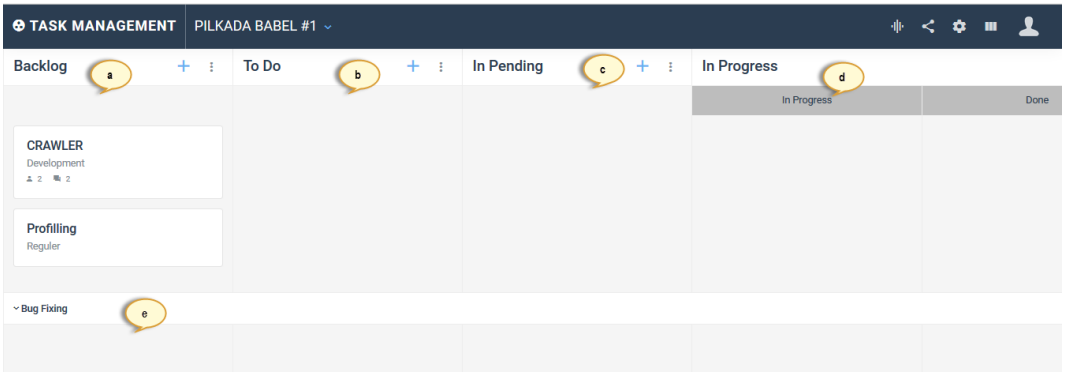


Gambar 4.33 Halaman Home

3) Halaman *Board*

Untuk melihat halaman *board* pengguna harus memilih salah satu *plan* yang tersedia. Seperti yang sudah dijelaskan di halaman *home*, *list plan* tersebut berada disebelah kiri.

Pilih nama (**Plan**) untuk menampilkan detil *Board* dan catatan penugasan dalam job terkait.



Gambar 4.34 Halaman Board

a. Pilih *Backlog* untuk membuat task

b. Lalu *To Do* untuk meng-*approve task* yang sudah diberikan

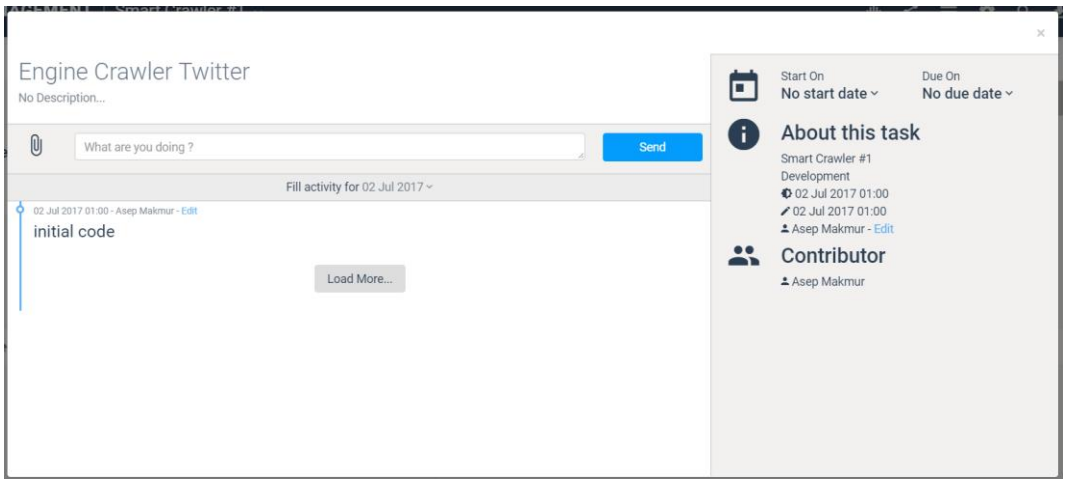
c. Kemudian *In Pending* bila task yang sudah diambil belum dikerjakan dikarenakan ada suatu hal.

d. Setelah itu Kolom *In Progress* yang artinya task tersebut sedang dikerjan oleh pegawai, apabila sudah selesai maka *task* tersebut berpindah tempat ke kolom *done*.

e. Kemudian ada kolom In Review apabila task yang sudah dikerjan dan sudah lolos unit test staging, maka task segera dicek oleh QA

f. Apabila ada bug maka QA akan mengeceknya sampai tidak lagi terdapat bug dan siap di-*deploy*.

Untuk melihat detail *task* maka klik salah satu *task*, maka akan muncul *pop-up* detail *task*.



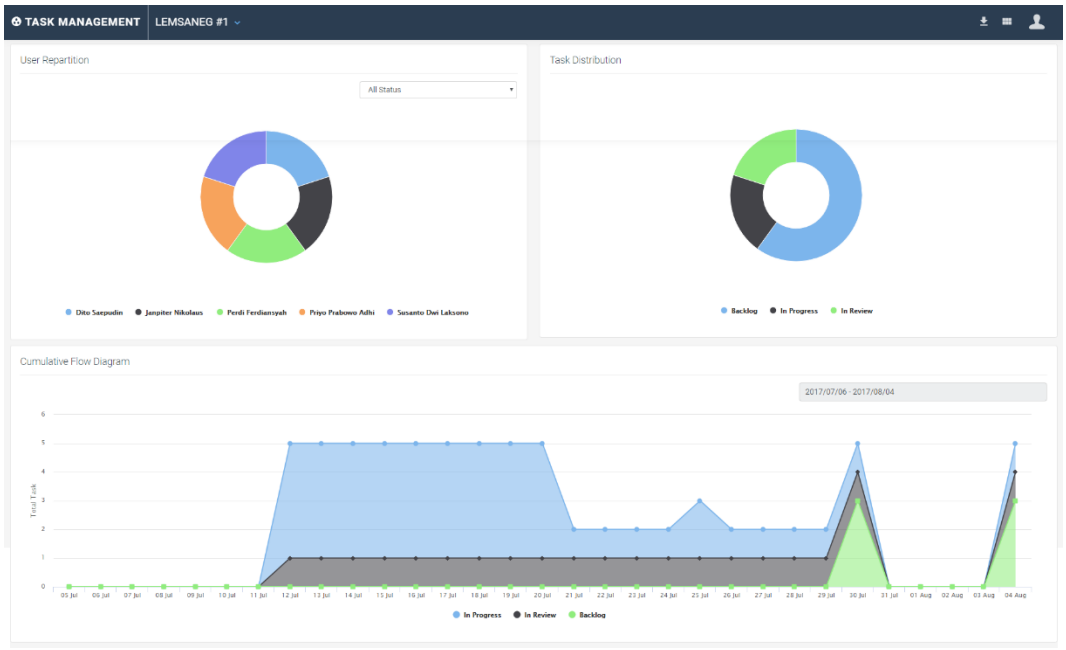
Gambar 4.35 Detail Task

4) Halaman *Analytics*

Setelah berada di halaman *board*, untuk masuk ke halaman *analytics* pengguna tinggal memilih tombol "Analytic" ( ) maka pengguna akan langsung dialihkan ke halaman *Analytics* sesuai dengan plan yang sedang aktif. Di halaman *analytics* ini pengguna akan disuguhkan dengan visualisasi *board*

dalam bentuk grafik sebagai acuan untuk melakukan proses analisa terhadap *plan* yang sedang aktif.

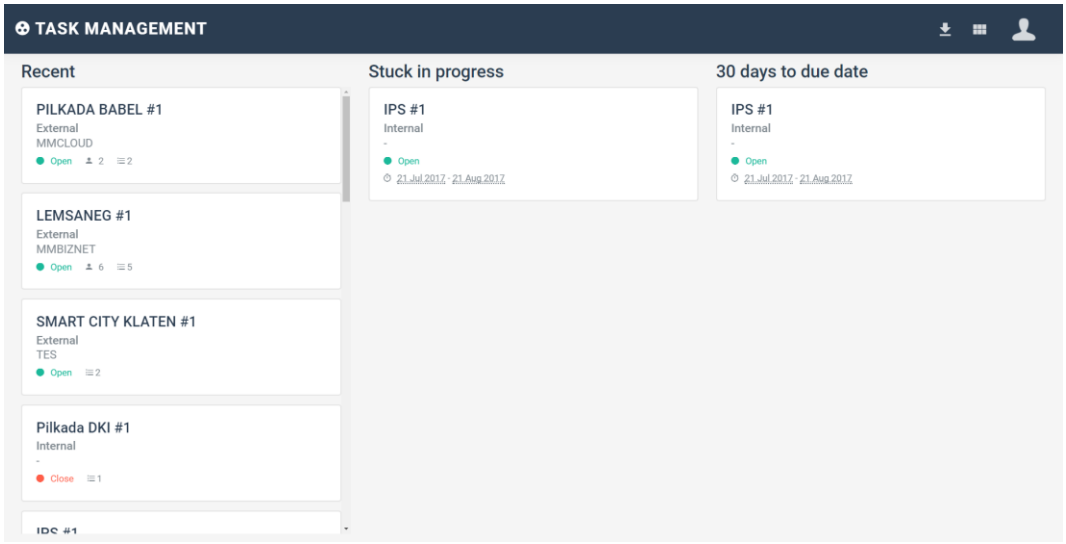
Berikut merupakan tampilan halaman *analytics*.



Gambar 4.36 Halaman Analytics

5) Halaman *Monitoring*

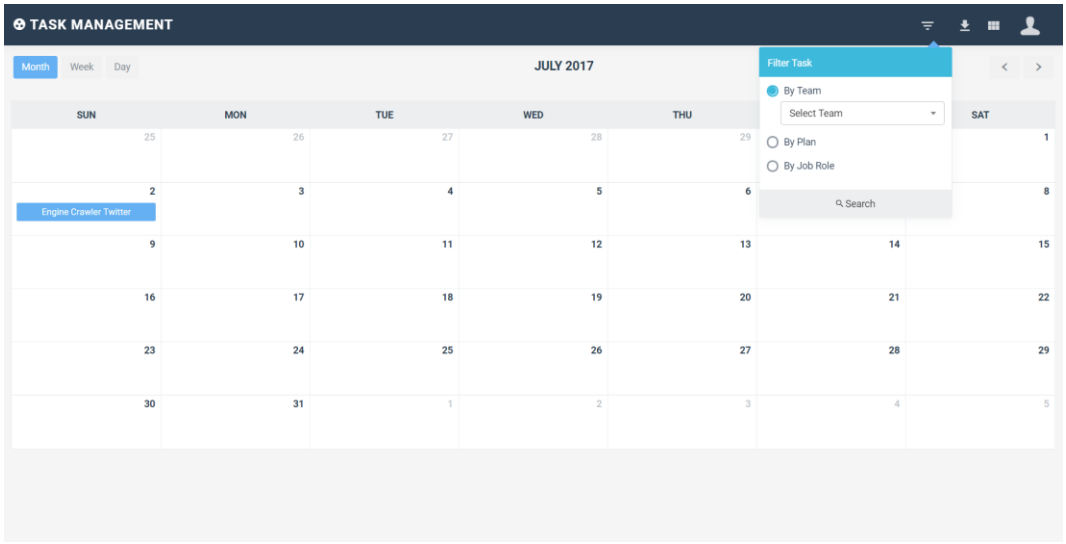
Untuk masuk ke halaman monitoring maka pilih menu “Monitoring”, maka akan tampil list plan terbaru, plan yang progress-nya terhambat dan plan yang mendekati tanggal deadline



6) Halaman *Review*

Untuk masuk ke halaman *review* maka pilih menu “Review”, maka akan tampil daftar aktifitas dalam bentuk kalender. Aktifitas tersebut dalam di filter berdasarkan *Team*, *Plan* dan *Job Role*.

Berikut merupakan tampilan halaman *review*.

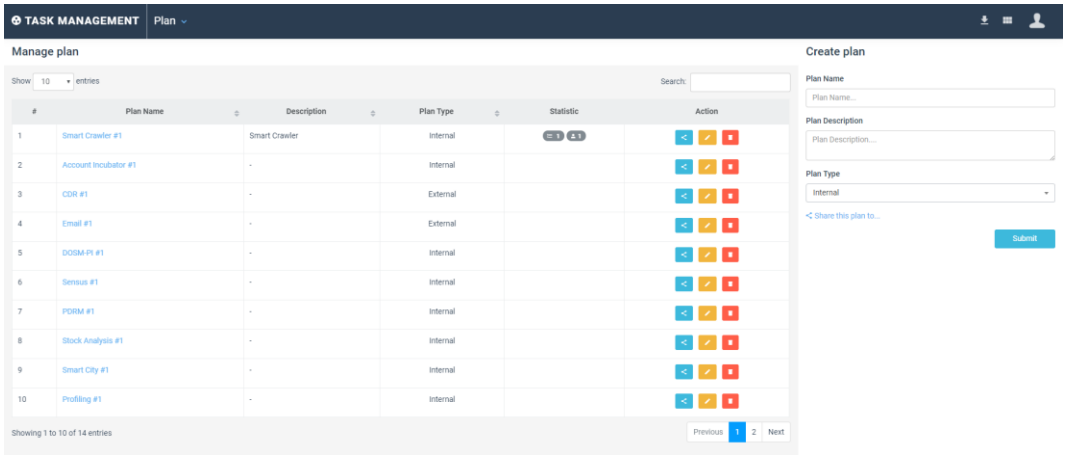


Gambar 4.37 Halaman Review

7) Halaman *Plan* & *Task*

Untuk masuk ke halaman *plan* & *task* maka pilih menu “Plan & Task”. Di halaman ini pengguna dapat mengelola data master *plan* & *task*. Pada saat membuka halaman *Plan* & *Task* pengguna akan melihat *list plan* dan *form create plan* yang berfungsi untuk mengelola data master *plan*.

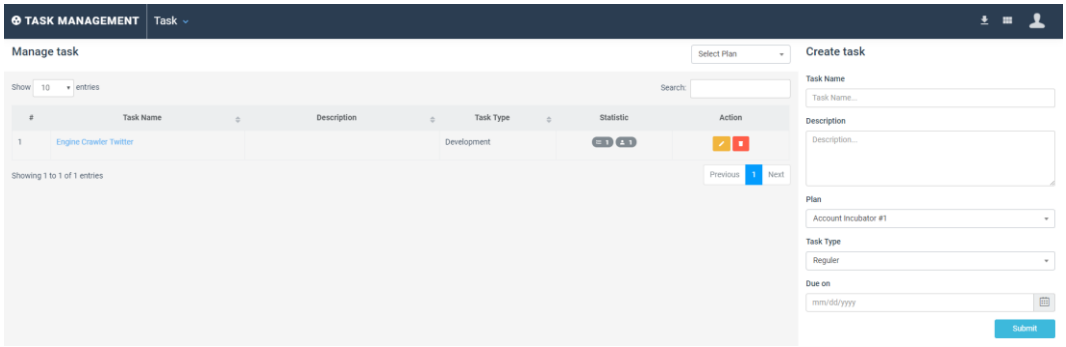
Berikut merupakan tampilan halaman *Plan* & *Task* pada saat memasuki halaman tersebut.



Gambar 4.38 Halaman Plan & Task

Untuk mengelola data master *task* maka pengguna tinggal memilih submenu “Task”, maka akan tampil *list task* dan *form create task*.

Berikut merupakan tampilan halaman *Plan* & *Task* submenu *Task*.

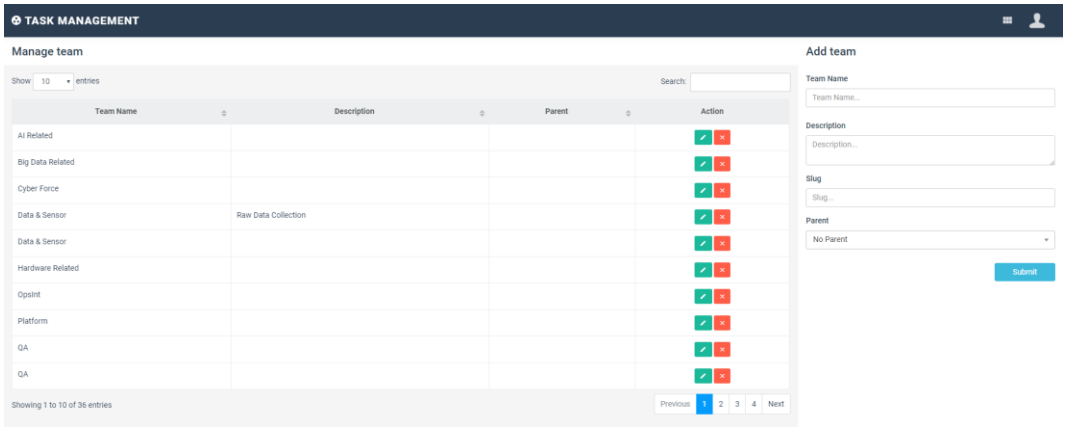


Gambar 4.39 Halaman Plan & Task submenu Task

8) Halaman *Team*

Untuk masuk ke halaman *team* maka pilih menu “Team”, kemudian akan tampil *list team* di sebelah kiri dan *form* tambah *team* disebelah kanan. Di halaman ini pengguna dapat mengelola data master *team*.

Berikut merupakan tampilan halaman *team.*

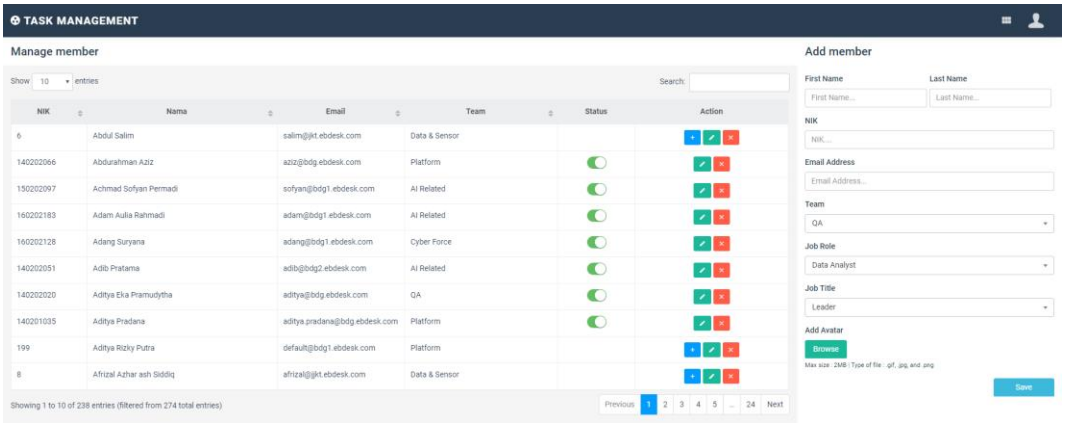


Gambar 4.40 4Halaman Team

9) Halaman *Member*

Untuk masuk ke halaman *member* maka pilih menu “Member”, kemudian akan tampil *list member* di sebelah kiri dan *form* tambah *member* disebelah kanan. Di halaman ini pengguna dapat mengelola data master *member*.

Berikut merupakan tampilan halaman *member.*

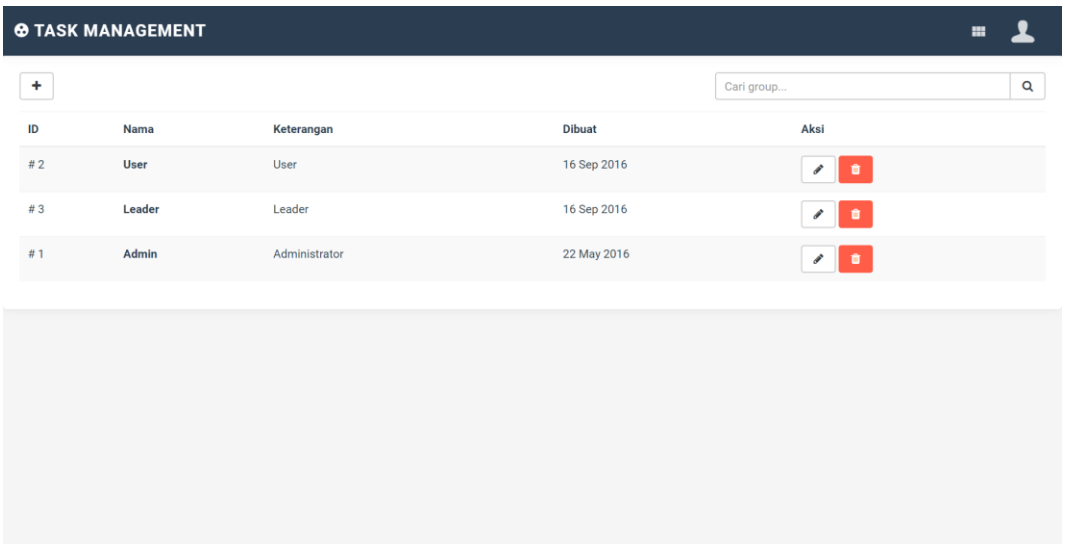


Gambar 4.41 5Halaman Member

10) Halaman *Group*

Untuk masuk ke halaman *group* maka pilih menu “Group”, kemudian akan tampil *list group*. Di halaman ini pengguna dapat mengelola data master *group*.

Berikut merupakan tampilan halaman *group.*

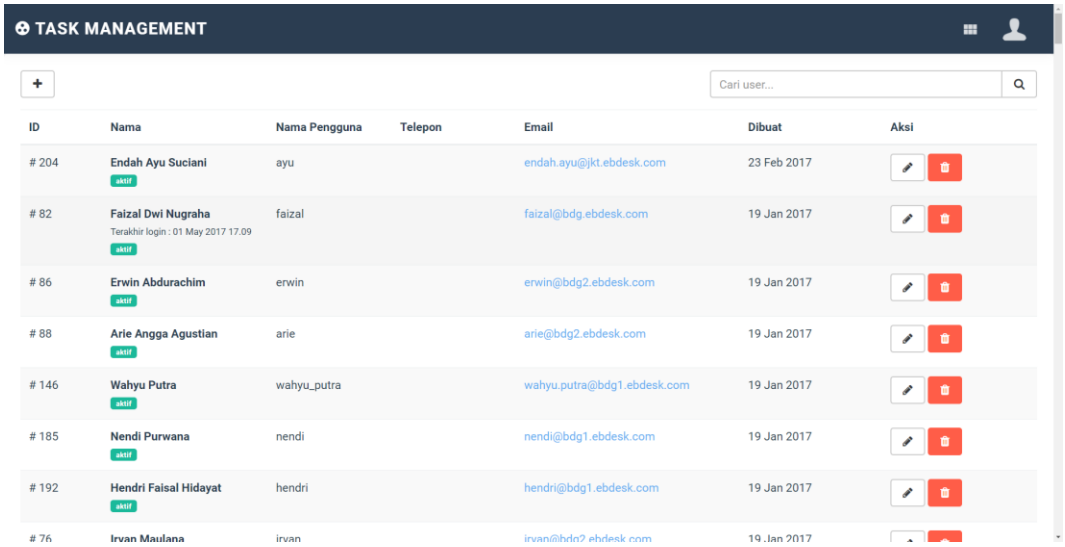


Gambar 4.42 Halaman Group

11) Halaman *User*

Untuk masuk ke halaman *user* maka pilih menu “User”, kemudian akan tampil *list user*. Di halaman ini pengguna dapat mengelola data master *user*.

Berikut merupakan tampilan halaman *user*.



Gambar 4.43 Halaman User