SKRIPSI

IMPROVISASI *TASK* PADA *SOFTWARE* MANAJEMEN PROYEK TAIGA DI PT JAVAN CIPTA SOLUSI

Disusun Oleh:

N a m a : Yasmin Aulia Ramadhini

NIM : 18523032

PROGRAM STUDI INFORMATIKA – PROGRAM SARJANA

FAKULTAS TEKNOLOGI INDUSTRI

UNIVERSITAS ISLAM INDONESIA

2022

HALAMAN PENGESAHAN DOSEN PEMBIMBING

IMPROVISASI *TASK* PADA *SOFTWARE* MANAJEMEN PROYEK TAIGA DI PT JAVAN CIPTA SOLUSI

TUGAS AKHIR JALUR MAGANG

Disusun Oleh:

N a m a : Yasmin Aulia Ramadhini

NIM : 18523032

Yogyakarta, 4 Nopember 2021

Pembimbing,

22/8/2022

(Irving Vitra Paputungan, S.T., M.Sc., Ph.D.)

HALAMAN PENGESAHAN DOSEN PENGUJI

IMPROVISASI *TASK* PADA *SOFTWARE* MANAJEMEN PROYEK TAIGA DI PT JAVAN CIPTA SOLUSI

TUGAS AKHIR JALUR MAGANG

Telah dipertahankan di depan sidang penguji sebagai salah satu syarat untuk memperoleh gelar Sarjana Komputer dari Program Studi Informatika – Program Sarjana di Fakultas Teknologi Industri Universitas Islam Indonesia

Yogyakarta, 4 Nopember 2021

Tim Penguji

Irving Vitra Paputungan, S.T., M.Sc., Ph.D. 22/8/2022

Anggota 1

Fayruz Rahma, S.T., M.Eng.

Anggota 2

Rahadian Kurniawan, S.Kom., M.Kom.

Mengetahui,
Ketua Program Studi Informatika – Program Sarjana
Fakultas Teknologi Industri
Universitas Islam Indonesia

(Dr. Raden Teduh Dirgahayu, S.T., M.Sc.)

HALAMAN PERNYATAAN KEASLIAN TUGAS AKHIR

Yang bertanda tangan di bawah ini:

Nama : Yasmin Aulia Ramadhini

NIM : 18523032

Tugas akhir dengan judul:

IMPROVISASI *TASK* PADA *SOFTWARE* MANAJEMEN PROYEK TAIGA DI PT JAVAN CIPTA SOLUSI

Menyatakan bahwa seluruh komponen dan isi dalam tugas akhir ini adalah hasil karya saya sendiri. Apabila di kemudian hari terbukti ada beberapa bagian dari karya ini adalah bukan hasil karya sendiri, tugas akhir yang diajukan sebagai hasil karya sendiri ini siap ditarik kembali dan siap menanggung risiko dan konsekuensi apapun.

Demikian surat pernyataan ini dibuat, semoga dapat dipergunakan sebagaimana mestinya.

Yogyakarta, 4 Nopember 2021

(Yasmin Aulia Ramadhini)

HALAMAN PERSEMBAHAN

Laporan Tugas Akhir ini adalah bagian dari ibadahku kepada Allah Swt., serta persembahan ucapan terima kasihku kepada kedua orangtua, Bapak Ferdi Firmansyah dan Ibu Lies Istiqomah, serta nenek Huryah yang selalu memberikan doa, dukungan dan motivasi dalam hidupku.

HALAMAN MOTO

"Allah tidak membebani seseorang melainkan sesuai dengan kesanggupannya."
(*QS. Al-Baqarah: 286*).

"Tidak harus mengikuti arah jalan, kamu bisa membuat jalanmu sendiri dan tinggalkan jejak di sana."
(*Ralph Waldo Emerson*).

"Work hard in silence, let success be your noise."
(*Frank Ocean*).

KATA PENGANTAR

Assalamualaikum Warahmatullahi Wabarakatuh

Alhamdulillah, segala puji dan syukur dihaturkan kepada Allah Swt., yang telah melimpahkan rahmat, taufiq serta hidayat-Nya sehingga dapat menyelesaikan Tugas Akhir ini yang berjudul "Improvisasi *Task* pada *Software* Manajemen Proyek Taiga di PT Javan Cipta Solusi". Laporan ini disusun sebagai bukti pelaksanaan dalam penjaluran magang dan memenuhi Tugas Akhir sebagai syarat untuk memperoleh sarjana pada Jurusan Informatika Universitas Islam Indonesia.

Penulis menyadari bahwa dalam penyusunan Tugas Akhir ini tidak dapat selesai dengan tepat waktu tanpa bantuan, bimbingan, serta motivasi dari berbagai pihak. Oleh karena itu penulis tidak lupa menyampaikan rasa terima kasih yang sebesar-besarnya kepada:

1. Allah Swt., yang telah memberikan rahmat dan hidayat-Nya yang selalu hadir di setiap proses dalam memberikan kesehatan, semangat, kemudahan, dan kemampuan kepada penulis untuk dapat menyelesaikan tugas akhir ini.
2. Kedua orang tua, yang telah memberikan doa, dukungan, serta dukungan kepada penulis.
3. Bapak Fathul Wahid, S.T., M.Sc., Ph.D., selaku Rektor Universitas Islam Indonesia dan Bapak Dr. Raden Teduh Dirgahayu, S.T., M.Sc., selaku Kepala Program Studi Informatika Universitas Islam Indonesia.
4. Bapak Irving Vitra Paputungan, S.T., M.Sc., Ph.D., selaku dosen pembimbing tugas akhir yang telah memberikan waktu, ilmu, arahan serta bimbingan secara langsung dalam penyelesaian tugas akhir ini.
5. Bapak dan Ibu dosen Jurusan Informatika, yang telah memberikan ilmu bermanfaat kepada penulis, semoga bapak dan ibu selalu diberikan kesehatan serta lindungan dari Allah Swt.
6. Saudara kandung sekaligus adik, Muhammad Yusuf Habibie yang telah memberikan dukungan kepada saya selama proses pengerjaan tugas akhir.
7. Seluruh rekan PT Javan Cipta Solusi yang telah memberikan ilmu, arahan dan pengalaman magang.
8. Nisa dan Dilfa, teman seperjuangan yang memberikan saran serta pencerahan selama penyusunan tugas akhir.
9. Nurul, Cai, Talitha, Egi, dan Lintang, yang selalu memberikan motivasi, dukungan, semangat serta doa bagi saya.
10. Teman-teman seperjuangan Sabil, Putri, Gina, Alya, Laila, yang melaksanakan tugas akhir.

11. Semua pihak terkait yang tidak bisa saya sebutkan satu per satu, terima kasih atas semua bentuk dukungan yang telah diberikan kepada penulis dalam proses pembuatan tugas akhir.

Atas segala bantuan, dukungan, dan bimbingan dari berbagai pihak yang telah diberikan, penulis dapat menyelesaikan Laporan Tugas Akhir dengan baik, semoga mendapatkan balasan kebaikan dari Allah Swt. Namun, penulis memohon maaf apabila selama pelaksanaan Tugas Akhir terdapat kesalahan dan khilafan. Penulis menyadari dalam pembuatan laporan ini masih jauh dari kata sempurna dan memiliki keterbatasan kemampuan yang dimiliki. Kritik maupun saran sangat membantu membangun kesempurnaan Tugas Akhir ini. Semoga laporan ini bermanfaat bagi semua yang membacanya.

Wassalamu'alaikum Warahmatullah Wabarakatuh

Yogyakarta, 4 Nopember 2021

(Yasmin Aulia Ramadhini)

SARI

Pengembangan perangkat lunak yang terus berevolusi memaksa perusahaan atau organisasi untuk mengadopsi metode, alat, atau pendekatan manajemen yang dapat membantu mempercepat pengerjaan *task* pada sebuah proyek. *Task* secara singkat merupakan bagian terkecil dan penting dari manajemen proyek pengembangan perangkat lunak. *Task* dilakukan setelah memastikan *backlog* dan *user story* terdefinisi dengan baik. Laporan ini menyajikan perbandingan beberapa *software project management tools* seperti Taiga dan Jira yang berdasarkan pada beberapa kriteria umum dan aspek kepuasan pengguna. Kemudian didapatkan kriteria spesifik yang dapat digunakan untuk membantu dalam improvisasi *task*, yaitu indikasi dan atribut. Improvisasi *task* dapat memberikan informasi yang berguna bagi tim pengembang, serta membantu mempercepat pengerjaan *task* pada proyek yang sedang berjalan.

Kata kunci: Improvisasi *Task*, Indikasi dan Atribut, *Software Project Management*, *Task*.

GLOSARIUM

Agile Scrum	Kerangka kerja yang digunakan untuk mengimplementasikan pengembangan suatu produk yang menerapkan metode *agile*.
Backlog/Product Backlog/ Backlog User Story	Berisi kumpulan *requirements* yang diberikan atau yang telah terdefinisikan dengan baik atau kumpulan dari *user story* yang telah disusun menjadi kesatuan utuh yang harus diselesaikan dalam pengembangan produk.
Daily Scrum	*Event* yang dilaksanakan setiap hari dengan durasi 15 menit untuk tim *developer*. Dilakukan pada setiap awal memulai pekerjaan untuk melaporkan kendala dan masalah pada pengerjaan *task* proyek.
Improvisasi *Task*	Pembaharuan informasi tambahan secara detail yang ada dalam sebuah *task*.
Issue/*Bugs*	Temuan dari sistem yang bersifat *high priority* dan *urgent* untuk dikerjakan oleh tim *developer*.
Sprint	Teknik iterasi dan bertahap secara dinamis dalam proses pembuatan suatu produk.
Sprint Backlog	Kumpulan *sprint* yang berisi *user story* atau *task*, kemudian akan dikerjakan pada masa *sprint* berlangsung.
Sprint Planning	Berfungsi untuk tim *scrum* dalam memeriksa pekerjaan dari *backlog/product backlog* yang akan dilakukan selanjutnya yang berjalan.
Sprint Review	Pelaksanaannya dijalankan di akhir *sprint*, hasilnya mendapatkan *feedback* dari pengerjaan proyek dan melakukan *planning* untuk *sprint* selanjutnya.
Sprint Retrospective	Dilakukan ketika *sprint review* sudah terlaksana. Mulai dari evaluasi tim, sampai dengan peningkatan yang akan dilakukan untuk *sprint* selanjutnya dengan kondisi yang lebih informal.
SOP (*Standard Operational*	

Procedure)	Standar atau panduan yang berkaitan dengan prosedur yang harus dijalankan.
Task	Bagian terkecil dan penting dari manajemen proyek pengembangan perangkat lunak.
Tim *Developer*	Pihak yang menjalankan proyek dan berperan penting dalam keberlangsungan sebuah proyek.
Task Management Tools	Alat yang digunakan untuk melakukan pengorganisasian, penentuan prioritas, dan penjadwalan *task* dengan cara menghasilkan pencapaian dari berbagai tujuan secara efisien, efektif, dan produktif.
User Story	Deskripsi atau penjelasan sederhana mengenai kebutuhan sistem dalam bentuk bahasa yang mudah dipahami oleh *user*.

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB I
PENDAHULUAN

1.1 Latar Belakang

Manajemen proyek merupakan tata cara pengelolaan sumber penghasilan dalam penyelesaian proyek dari awal sampai proyek selesai dikerjakan (Dimyati & Nurjaman, 2014). Konsep manajemen proyek pada dasarnya dapat dipakai pada jenis proyek apapun, dan dipakai secara luas untuk menyelesaikan proyek besar dan kompleks (Sitanggang, Simarmata, & Luthan, 2019). Manajemen proyek diperlukan oleh tim *developer* atau pengembang untuk mengimbangi dampak tingginya permintaan pasar akan pengembangan proyek teknologi informasi. Fungsi alat manajemen proyek dalam hal ini adalah sebagai alat bantu dalam melakukan pengelolaan dan dokumentasi pengembangan perangkat lunak (Suharno, Gunantara, & Sudarma, 2020).

Terdapat beberapa alat bantu manajemen proyek, salah satunya adalah Taiga. Taiga merupakan perangkat lunak *open source* yang cocok untuk mendukung proses pelacakan *issue, multiplatform importers*, dan kustomisasi pada sebuah proyek (Özkan & Mishra, 2019). Taiga juga mendukung *scrum, Kanban,* dan *user story* secara mudah ketika digunakan. Taiga sangat tepat digunakan pada proyek kecil dan proyek yang tidak terlalu kompleks (Manole & Avramescu, 2017). Alat bantu yang lain adalah Jira, yang merupakan perangkat lunak *issue* dan *bug tracking.* Jira mendukung fitur pelaporan, *mapping* alur kerja untuk *issue*, dan pengorganisasian proyek (Makhija & Goyal, 2014). Jira juga mendukung kustomisasi *scrum Boards* dan *Kanban Board* yang fleksibel. Fitur penting Jira adalah dapat melakukan kustomisasi *developer tool integrations*, memperhitungkan alur diagram, pelaporan, *issue* dan *bug tracking* (Özkan & Mishra, 2019). Jira dipilih karena memiliki persamaan dalam mendukung pengembangan proyek yang menggunakan pendekatan atau *best practice scrum* dan *agile* yang digunakan oleh PT Javan Cipta Solusi.

Salah satu bagian kecil namun penting dari sebuah manajemen proyek adalah *Task. Task* merupakan tugas yang harus diselesaikan dalam tenggat waktu yang ditentukan dan harus memiliki kontribusi terhadap tujuan yang berhubungan dengan pekerjaan (Setyawati, Santamoko, Handoko, & Setiawan, 2021). *Task* berfungsi sebagai sarana untuk membedakan berbagai komponen lain dalam sebuah proyek. Sebuah *task* dapat dipecah menjadi beberapa *subtask* yang juga memiliki tanggal mulai dan akhir sesuai tenggat waktu penyelesaiannya (Lam & Maheshwari, 2001). *Task* diatur oleh seorang manajer. Manajer *task* yang akan

menentukan tenggat waktu, memprioritaskan tugas, memilih anggota tim untuk ditetapkan, melacak performa dan kemajuan proyek, dan memastikan tidak terjadinya penundaan yang menyebabkan keterlambatan proyek.

Laporan ini menyajikan analisis dari perbandingan dua perangkat manajemen proyek yang telah disebutkan. Analisis ini dimaksudkan untuk mencari cara improvisasi yang tepat terhadap suatu *task* pada manajemen proyek. Cara improvisasi yang tepat diperlukan karena pada tahap pembuatan *task* di Javan belum bisa memenuhi kebutuhan informasi yang jelas untuk *engineer* pada saat pembuatan *task*. Analisis ini dapat memberikan manfaat dan pengetahuan baru, serta memiliki gambaran dalam pembuatan *task* pada suatu proyek.

Laporan ini mencakup rangkuman dari aktivitas selama magang yang dilaksanakan di PT Cipta Javan Cipta Solusi, sebagai bentuk laporan tugas akhir penjaluran magang tahun ajaran 2021/2022.

1.2 Ruang Lingkup Magang

Pelaksaan magang di PT Javan Cipta Solusi berlangsung selama kurun waktu enam bulan lebih dengan periode pertengahan Maret 2021 hingga akhir September 2021. PT Javan Cipta Solusi merupakan sebuah perusahaan yang bergerak dalam bidang jasa teknologi informasi yang fokus dalam penyedia solusi untuk optimasi proses bisnis IT. PT Javan Cipta Solusi memiliki dua cabang kantor yang berada di kota Bandung dan Yogyakarta. Beberapa proyek yang pernah terlibat selama proses magang yaitu pengembangan *e-commerce* pada proyek LKPP, *maintenance* proyek Pengaduan WBS-LKPP, dan proyek PLN. Adapun aktivitas yang dilakukan selama magang sebagai berikut:

a. Mempersiapkan *tools* untuk Magang
b. Mengawasi jalannya proyek PLN Devops
c. Mengawasi jalannya proyek PLN Mobile *Backend*
d. Membuat dokumen *deliverable* proyek PLN Devops
e. Membuat dokumen *deliverable* proyek PLN Mobile *Backend*
f. Melakukan analisis *issue* dalam proyek Pengaduan WBS – LKPP
g. Melakukan *testing* dalam proyek Pengaduan WBS – LKPP
h. Membuat *task* proyek E-Katalog (*e-commerce*)
i. Membuat *user guide* proyek E-Katalog (*e-commerce*)
j. Melakukan *testing* dalam proyek E-Katalog (*e-commerce*)

1.3 Tujuan Magang

Tujuan dari Laporan Tugas Akhir ini sebagai berikut:

a. Mengetahui perbandingan *software* manajemen Taiga dengan Jira yang dapat dijadikan sebagai referensi dalam pembuatan *task*.
b. Melakukan improvisasi terhadap *task* pada *software* manajemen Taiga.

1.4 Manfaat Magang

Manfaat dari improvisasi *task* pada *software* manajemen proyek adalah sebagai berikut:

a. Mengetahui perbedaan ketika pembuatan *task* pada *software* manajemen Taiga dan Jira yang dapat dijadikan sebagai referensi dalam pembuatan *task*.
b. Improvisasi *task* dapat mempercepat kinerja dan memperjelas tujuan *task* menjadi lebih lengkap dan rinci dalam proses manajemen proyek.
c. Improvisasi *task* mempermudah proses pemahaman kebutuhan dalam pengerjaan proyek.
d. Improvisasi *task* dilakukan untuk memaksimalkan kebutuhan yang sudah ada menjadi lebih baik dan stabil.
e. Meningkatkan kolaborasi dan *awareness* tim pada saat pengerjaan *task*.

1.5 Sistematika Penulisan

Sistematika penulisan laporan ini disusun untuk memberikan gambaran umum tentang laporan akhir yang dikerjakan. Adapun Sistematika Penulisan dari laporan ini adalah sebagai berikut:

a. BAB I: Pendahuluan

   Bab ini berisikan latar belakang, ruang lingkup magang, tujuan, manfaat dan sistematika penulisan laporan akhir.
b. BAB II: Kajian Pustaka

   Bab ini berisikan ringkasan hasil analisis yang dilakukan dan memiliki kesamaan dengan penelitian terdahulu, serta dasar-dasar teori yang berhubungan dengan laporan ini.
c. BAB III: Pelaksanaan Magang

   Bab ini berisikan tahapan pelaksanaan magang, manajemen proyek, serta improvisasi *task* pada *software* manajemen proyek yang dilakukan menggunakan *task* manajemen Taiga.
d. BAB IV: Refleksi Pelaksanaan Magang

   Bab ini berisikan refleksi dari selama pelaksanaan magang di PT Javan Cipta Solusi mulai dari segi teknis maupun nonteknis.

e. BAB V: Kesimpulan dan Saran

   Bab ini berisikan kesimpulan dari improvisasi *task* pada *software* manajemen proyek, serta saran bagi pihak terkait dengan penelitian dan laporan serupa.

BAB II
KAJIAN PUSTAKA

Bab ini menyajikan dan menjelaskan teori-teori dasar dari proses melakukan improvisasi pada *task*, seperti *Task Management,* Kolaborasi dengan *Task Management Tools*, *Agile Project Management Tools* dan teori lainnya yang masih berkaitan dengan *task management tools,* untuk menunjang dalam perancangan dan pembuatan tugas akhir ini.

2.1 *Task Management*

Manajemen *task* adalah proses melakukan pengorganisasian, penentuan prioritas, dan penjadwalan *task* dengan cara menghasilkan pencapaian dari berbagai tujuan secara efisien, efektif, dan produktif. Tujuan harus didasari pada nilai dan dipandu oleh pernyataan misi. Menurut Bernhardt (2014), terdapat lima *goals* dasar yang disingkat menjadi SMART, yaitu *Specific, Measureable, Attainable, Relevant,* dan *Time-Bound* (Bernhardt, 2014). Tujuan akhir dari pencapaian harus melalui penyelesain sebuah *task. Task* harus dikelola dengan terampil sehingga dapat memenuhi nilai dan tujuan yang telah ditetapkan. Berikut tahap-tahap dalam mengelola *task* (Bernhardt, 2014), yaitu:

a. *Brainstorm tasks required for the goal regardless of order.*

Mengumpulkan semua gagasan dari hasil *brainstorming* dari *task* dan mengabaikan prioritas dalam pembuatan *task,* dapat mempertimbangkan spontanitas dari semua *task* yang diperlukan.

b. *Breakdown larger tasks into smaller bite-sized pieces.*

Setelah *task* dibagikan, *task* yang besar dapat dipecah menjadi beberapa bagian/*sub*. Tujuannya adalah agar lebih mudah dalam pengelolaan untuk mengedepankan penjadwalan yang lebih mudah dalam pengerjaannya, memperkirakan waktu supaya lebih akurat dan lebih cepat dalam penyelesaian *task.*

c. *Divide Task into Major Groups.*

Memisahkan *task* menjadi beberapa kategori untuk mewakili kepentingan dan urgensi, seperti pada Gambar 2.1. Ini berguna untuk memudahkan dalam proses alokasi *task*, memisahkan *task* mana saja yang menjadi prioritas dalam pengerjaan serta estimasi dalam penyelesaian *task* dalam proyek. Berikut adalah kategorisasinya:

1. *Important and Urgent*

   Pada poin ini, *task* harus dikerjakan sesegera mungkin karena *task* tersebut merupakan bagian terpenting dari sebuah sistem dan memiliki konsekuensi *negative* ketika melewati batas perkiraan pengerjaan. *Important and urgent* mempunyai tenggat waktu yang telah ditentukan selesainya.

2. *Important and Not Urgent*

   Pada poin ini, kita dapat memprioritaskan *task* dan menyusun strategi dengan baik, kemudian memiliki waktu untuk bisa memaksimalkan dalam pengerjaan *task.* Di sinilah Javan ketika dalam pembuatan sebuah *task.*

3. *Not Important and Urgent*

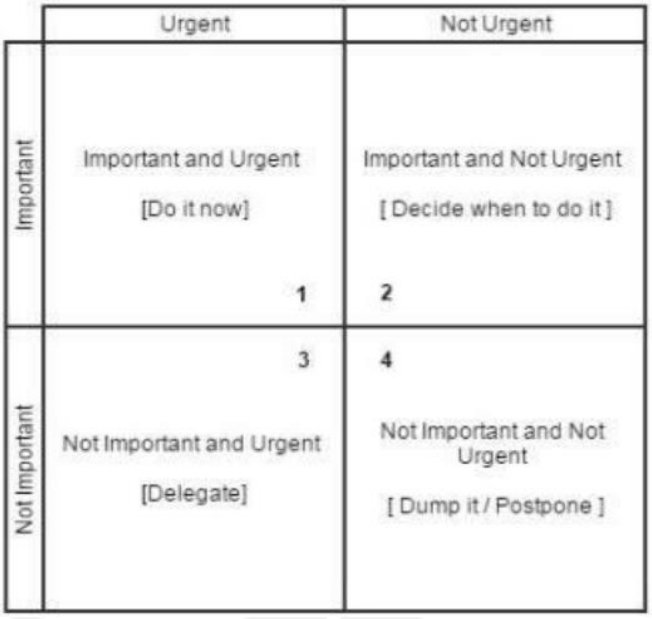
   Pada poin ini, adalah yang paling berhati-hati dikarenakan *task* yang masuk terkadang tidak terdapat pada daftar kebutuhan pada awal pembuatan sistem tetapi harus dikerjakan untuk memenuhi kebutuhan klien, seperti *task bugs/issue*.

4. *Not Important and Not Urgent*

   Pada poin ini, *task* juga datang terlambat atau sebagai tambahan kebutuhan. Dalam pengerjaannya pun bisa ditunda, tetapi sudah masuk ke dalam *list* pengerjaan.

Kategorisasi ini berguna untuk memudahkan dalam proses alokasi *task*, memisahkan *task* mana saja yang menjadi prioritas dalam pengerjaan serta estimasi dalam penyelesaian *task* dalam proyek.

Task harus menentukan prioritas terlebih dulu. Seseorang harus memiliki pemahaman jelas tentang tujuan dan sumber daya yang tersedia. *Task Matrix* berguna untuk membantu memprioritaskan masalah yang kompleks atau tidak jelas, dapat menyediakan metode cepat dan mudah, serta konsisten dalam opsi evaluasi, terakhir dapat disesuaikan untuk banyak kebutuhan dalam pengaturan prioritas seperti: *project, services* dan *personal* (Jyothi & Parkavi, 2016), menjelaskan bahwa sebuah *task* harus ditentukan prioritasnya terlebih dulu. Seseorang harus memiliki pemahaman jelas tentang tujuan dan sumber daya yang tersedia.



Gambar 2.1 *Task Matrix*

Sumber: Jyothi & Parkavi (2016)

d. *Prioritize Tasks within Groups.*

Setelah memisahkan *task* dalam pengelompokan seperti kategori di atas, berikan penomoran atau *sign* sebagai bentuk prioritas dalam pengerjaan *task.*

e. *Schedule Tasks.*

Melakukan penjadwalan pada *task*, sesuaikan dengan tenggat waktu yang sudah diberikan, dan estimasikan sesuai dengan prioritas dan urgensi dari *task.*

f. *Schedule and Work toward Milestones.*

Milestones adalah poin penting dalam pengembangan di sebuah proyek, dengan *milestones* pengerjaan *task* dapat memiliki arah dan tujuan yang pasti.

g. *Check-off Completed Tasks.*

Melakukan cek terhadap *task* yang sudah selesai. Hal ini berfungsi untuk melacak *task* sehingga tidak ada kebingungan pada *task* yang sudah selesai dan *task* yang belum diselesaikan.

2.2 Kolaborasi dengan *Task Management Tools*

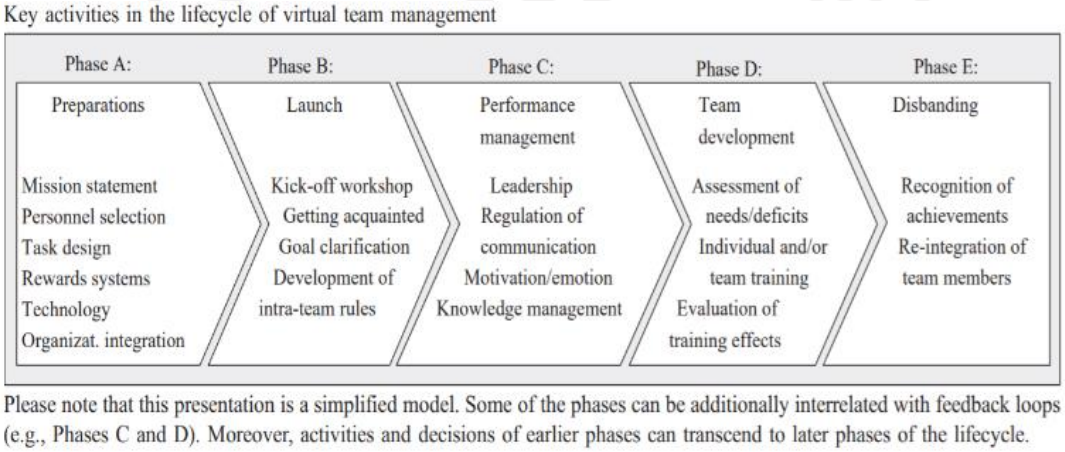
Beberapa dekade lalu, kebutuhan untuk mengatur dan mensistemasikan pekerjaan proyek dari beberapa sumber, mulai memunculkan peran pentingnya dalam pengembangan perangkat lunak. Hal ini memerlukan sebuah tim proyek untuk berkolaborasi dalam pengembangan

mengelola, mengontrol, serta memandu proses sistem perangkat lunak yang berfokus pada efektivitas terhadap pengelolaan kolaborasi tim proyek menggunakan *task management tools* (Chasanidou, Elvesæter, & Berre, 2016).

Menurut model *lifecycle* heuristik tentang manajemen tim virtual dijelaskan bahwa, terdapat lima fase tugas manajemen khusus dan topik yang harus ditangani selama bekerja tim (Chasanidou, Elvesæter, & Berre, 2016), yaitu:

a. Fase pertama, ***Preparation***, mempersiapkan tugas dan keputusan yang relevan untuk implementasi tim virtual.
b. Fase kedua, ***Launch***, menjelaskan kegiatan terkait awal kerja tim.
c. Fase ketiga, ***Performance of management***, mencakup masalah tentang kepemimpinan, pemeliharaan motivasi, dan komunikasi dalam tim.
d. Fase keempat, ***Team development***, menjelaskan kegiatan evaluasi dalam proses tim, pelatihan bersama tim dan anggota baru.
e. Fase kelima, ***Disbanding***, terdiri dari tugas-tugas seperti penghargaan atas pencapaian dan reintegrasi anggota tim.

Berikut merupakan Gambar 2.2, yang menampilkan aktivitas dari manajemen tim virtual, mulai dari fase *preparation* sampai dengan *disbanding*.



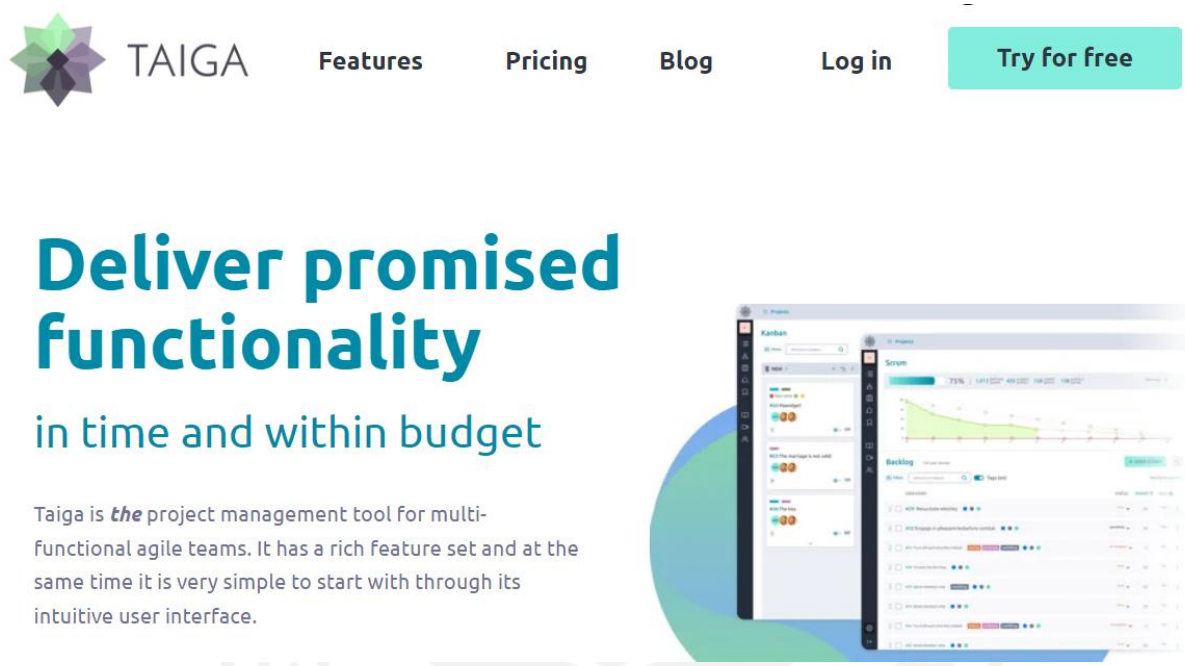
Gambar 2.2 *Life Heuristik*

Sumber: Chasanidou, Elvesæter, & Berre (2016)

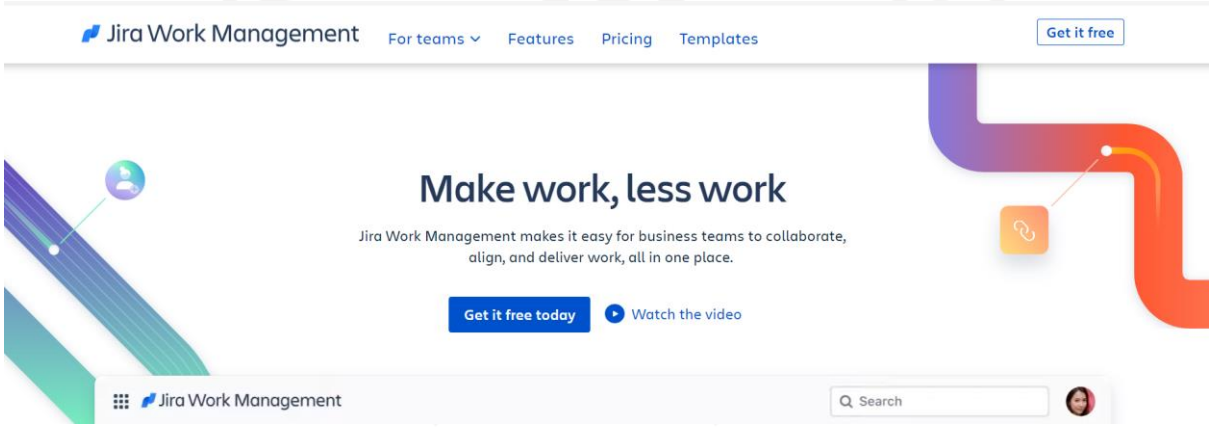
2.3 *Agile Project Management Tools*

Terkait pembahasan mengenai kolaborasi tim dengan *task management tools* pada subbab 2.2, *task management tools* tidak terlepas dari metode *agile.* Salah satu metode *agile* yang paling popular digunakan adalah *scrum*. *Agile scrum* merupakan salah satu pendekatan

terbaik yang mudah disesuaikan dengan kebutuhan yang diperlukan (Chasanidou, Elvesæter, & Berre, 2016). Maka dari itu, diperlukan sebuah *task management tools* yang mengadopsi *Agile scrum*.



Gambar 2.3 Tampilan Website Taiga *Management Tools*



Gambar 2.4 Tampilan Website Jira *Management Tools*

Salah satu *task* manajemen yaitu Taiga (https://www.taiga.io/) memiliki fitur menarik seperti dukungan *scrum, user story,* serta *Kanban*. Penggunaan Taiga mudah digunakan, bagus untuk tim kecil dengan proyek yang tidak terlalu rumit (Manole & Avramescu, 2017). Selain itu, Taiga memberikan kepada pengguna kontrol penuh terhadap *user story*, memberikan

notifikasi tentang status *task*, dan mendukung komunikasi antara anggota tim (Villavicencio, Narváez, Izquierdo, & Pincay, 2017). Tampilan web Taiga dapat dilihat pada Gambar 2.3.

Task manajemen lain yaitu Jira (https://www.atlassian.com/software/jira/work-management) memiliki fitur *reporting* yang canggih, pemetaan alur kerja *issue* dan pengorganisasian proyek, dapat disesuaikan dengan proyek (Makhija & Goyal, 2014). Hasil perhitungan kasar indikator kinerja dari penelitian tersebut, terbukti memperoleh nilai terbaik, hampir maksimal. Ini dibuktikan melalui pangsa pasar yang besar. Tampilan web Jira dapat dilihat pada Gambar 2.4.

Taiga ideal untuk tim *developer*. Selain berbasis *open source*, mereka menyediakan *issue tracking*, *multiplatform importers*, dan kustomisasi mudah. Kemudian, Jira dapat melakukan kustomisasi *scrum boards* dan *Kanban boards*. Fitur lainnya dapat melakukan kustomisasi integrasi dengan *developer tool*, perhitungan alur diagram, *reporting*, *issue* dan *bug tracking* (Özkan & Mishra, 2019).

2.4 Kajian Pustaka

Terdapat beberapa penelitian yang dilakukan tentang *task management* Taiga dan Jira. Pada penelitian yang berjudul *"Task Management Tools for the Structural Engineer"* oleh Bernhardt (2014), dijelaskan bahwa manajer *task* melakukan pengorganisasian, penentuan prioritas, dan penjadwalan *task* dengan cara menghasilkan pencapaian dari berbagai tujuan secara efisien, efektif, dan produktif. Tujuan harus didasarkan pada nilai dan dipandu oleh pernyataan misi.

Pada penelitian yang berjudul "*A Comparative Analysis of Agile Project Management Tools*", dijelaskan bahwa Taiga memiliki fitur menarik seperti dukungan pada *scrum, user story*, serta *Kanban*. Taiga mudah digunakan, kemudian juga bagus untuk tim kecil di dalam proyek yang tidak terlalu rumit. Selain itu, penelitian berjudul "*Learning scrum doing by doing real-life project"*, menjelaskan Taiga memberikan *control* penuh kepada pengguna pada fitur *user story*, memberikan notifikasi tentang status *task*, dan mendukung komunikasi antara anggota tim. Kemudian penelitian lain berjudul "*Agile Project Management Tools: A Brief Comparative View*", menjelaskan Taiga lebih ideal untuk tim *developer*. selain berbasis *open source*, mereka juga menyediakan *issue tracking*, *multiplatform importers*, dan kustomisasi dengan mudah.

Penelitian dengan judul "*Comparative Study of Project Tracking and Management Tools*" mengatakan, Jira sendiri memiliki fitur *reporting* yang canggih, pemetaan alur kerja

terhadap *issue* dan pengorganisasian proyek, yang dapat disesuaikan dengan keadaan proyek. Hasil perhitungan kasar pada indikator kinerja dari penelitian tersebut, terbukti memperoleh nilai terbaik, hampir mendekati maksimal. Ini dibuktikan melalui pangsa pasar yang besar.

Kesimpulan dari kajian pustaka yang dilakukan terdapat penelitian yang menjelaskan kegunaan dari *task* manajemen yang pada tujuan akhirnya untuk menyelesaikan sebuah *task*. Selain itu, *task* harus ditentukan prioritasnya terlebih dahulu sebelum dibuat agar sesuai dengan tujuan akhir proyek. Terakhir, Taiga dan Jira memiliki fitur yang sesuai dengan kebutuhan masing-masing proyek seperti *scrum* dan *Kanban* atau *tracking bug* dan *issue*, serta untuk proses perencanaan yang matang, tergantung dari kebutuhan yang diperlukan oleh setiap tim pengembang.

BAB III
PELAKSANAAN MAGANG

3.1 Tahap Pelaksanaan Magang

Berikut pada Tabel 3.1, merupakan tahap-tahap dalam pelaksanaan magang yang telah dilakukan. Mulai dari melakukan perkenalan dan persiapan sampai dengan penutupan proyek. Ketika masa magang telah selesai, ada beberapa proyek yang masih dalam proses pemantauan dan pengendalian, dan ada juga yang masih dalam tahap implementasi pembuatan *task*.

Tabel 3.1 Tahap Pelaksanaan Magang

No.	Aktivitas	Waktu Mulai	Durasi
1	Perkenalan dan Persiapan	Pekan ketiga Maret 2021	1 Minggu
2	Proyek PLN Devops dan Mobile *Back end*		
	On-Boarding	Akhir Maret 2021	1 Minggu
	Implementasi Pembuatan *Task*	April 2021 – Mei 2021	3 Bulan
	Pemantauan dan Pengendalian	April 2021 – Mei 2021	3 Bulan
	Penutupan Proyek	Juli 2021	1 Minggu
3	Proyek Pengaduan WBS - LKPP		
	On-Boarding	Agustus 2021	1 Minggu
	Implementasi Pembuatan *Task Issue*	Agustus 2021 – September 2021	1 Bulan
	Pemantauan dan Pengendalian	Agustus 2021 – September 2021	1 Bulan
4	Proyek E-Katalog		
	On-Boarding	Pekan kedua Agustus 2021	1 Minggu
	Implementasi Pembuatan *Task dan Task Issue*	Agustus 2021 – September 2021	1 Bulan

3.2 *On-Boarding* Magang

On-boarding dilakukan sebelum terjun ke dalam sebuah proyek, untuk diberikan pembekalan berupa *requirement*/kebutuhan yang diperlukan dalam pembuatan sebuah sistem. Pembekalan yang diberikan dikenal sebagai *sharing knowledge*. Pembekalan diberikan oleh *project manager* maupun *system analyst*.

Pada tahap *on-boarding* ini, diberikan penjelasan berupa pengenalan terhadap proyek, penjelasan alur kerja proyek, dan manajemen pada proyek. Aktivitas tersebut berhubungan dengan tugas utama untuk melakukan manajemen *task* pada sebuah proyek. Dalam pembuatan

task memerlukan beberapa atribut dan indikasi yang sesuai dengan kebutuhan yang diperlukan proyek dan tim proyek.

Proses belajar dan beradaptasi dalam melakukan manajemen *task* pada proyek terbilang singkat tetapi mudah dipahami. Hal tersebut berkaitan dengan ketersediaan kebutuhan/*requirement* proyek oleh *system analyst* utama yang telah melakukan analisis sebelumnya terkait apa saja yang akan dimasukkan ke dalam *task* manajemen. Hasil analisis tersebut berfungsi sebagai bahan utama dalam pembuatan sebuah *task* yang berguna bagi tim dalam pengerjaan sebuah sistem. Pada *task* berisi tujuan utama pengerjaan *user story*, deskripsi *task*, alokasi waktu, hingga alokasi tim.

3.3 Manajemen Proyek

Untuk membangun dan mengembangkan sistem dan server PLN Devops dan Mobile *Back end*, tahapan manajemen proyek yang berjalan selama magang adalah sebagai berikut:

3.3.1 Inisialisasi Proyek

Tahap pertama dari sebuah proyek adalah tahap inisiasi. Pada tahap ini, ide untuk proyek dilakukan eksplorasi dan elaborasi. Tujuannya untuk memeriksa kelayakan proyek (Westland, 2007). Seiring dengan kebutuhan yang mulai bervariasi, proses pengembangan proyek PLN Devops dan Mobile *Back end* terbagi dalam sejumlah peran:

a. *Project manager*, mengelola kemajuan proyek dan menyesuaikan kebutuhan yang telah diberikan oleh klien, memastikan proyek memenuhi tenggat waktu yang telah ditentukan, mengelola komunikasi serta hubungan dengan klien dan pemangku kepentingan serta merancang dan menandatangani kontrak. Untuk pada *scrum* adalah sebagai *product owner*. Pada proyek ini, *project manager* dipimpin oleh satu orang yaitu Bapak Dicky Puja Pratama dan penulis disini berperan sebagai asisten *project manager*.
b. *System analyst*, bertanggung jawab terhadap mengumpulkan semua kebutuhan/*requirement*, membuat *task* untuk tim *programmer* yang berisikan kumpulan kebutuhan yang akan dikerjakan, berkomunikasi dengan *programmer* untuk menghasilkan sistem baru. Untuk peran ini anggotanya ada penulis.
c. *Programmer*, bertanggung jawab dalam melakukan eksekusi dan implementasi perancangan kebutuhan seperti pengkodean dan *debugging*, melakukan kelola *database*, melakukan edit kode sumber/*source code*, dan melakukan analisis algoritma. Adapun

anggota yang terlibat di bidang ini antara lain Muhammad Akmal, Qisthi Ramadhani, Ricky Febriansah, Nas ruddin (sebagai *backup*), dan Dika Priska.

d. *Quality assurance*, bertanggung jawab dalam memastikan bahwa produk yang telah selesai memenuhi standar dari kualitas perusahaan dan berjalan sesuai dengan kesepakatan. Peran ini dilakukan oleh tim dari klien.

3.3.2 Pendefinisian Proyek

Setelah perencanaan proyek yang dikembangkan pada tahap inisiasi telah disetujui, proyek memasuki tahap pendefinisian proyek. Pada tahap ini, persyaratan yang terkait dengan hasil proyek harus ditentukan sejelas mungkin (Westland, 2007). Dalam hal ini melibatkan identifikasi berupa hasil yang diharapkan oleh semua pihak yang terlibat dengan hasil proyek.

PLN Devops dan Mobile *Back end* merupakan sebuah proyek yang dikembangkan pada tahun 2021. Proyek ini dilakukan atas kerja sama antara pihak Javan dan pihak PLN untuk mengembangkan sistem dan server pada aplikasi yang dikembangkan oleh PLN. Sistem kerja pada proyek ini adalah dengan mengirimkan empat *senior programmer* yang pembagiannya menjadi PLN Devops dan PLN Mobile *Back end*, masing-masing dua orang *senior programmer*. Untuk pembuatan *requirement* juga dilakukan oleh tim klien, tim Javan hanya menerima *task* yang ada pada *software project management* yang digunakan oleh klien yaitu Jira.

3.3.3 Pelaksanaan dan Pengembangan Proyek

Selama tahap pengembangan, semua yang dibutuhkan untuk implementasi proyek telah diatur oleh pihak PLN. Mulai dari jadwal, bahan dan alat dipesan, serta instruksi yang diberikan kepada tim dan sebagainya. Tahap pengembangan dinyatakan selesai apabila implementasi telah terealisasi sesuai dengan rencana awal pada tahap perencanaan. Semua hal harus jelas bagi pihak-pihak yang akan melaksanakan implementasi (Westland, 2007). Proyek akan terbentuk selama tahap implementasi, mulai dari *programmer* sampai dengan *quality assurance* yang mendapatkan tugas sesuai dengan perannya.

3.3.4 Pemantauan dan Pengendalian Proyek

Dalam tahap ini, proses manajemen untuk tahap pemantauan dan pengendalian proyek dilakukan guna memantau dan mengontrol penyelesaian *deliverables* sebagai *output* akhir dari pengembangan proyek. Pada fase ini semua yang diperlukan akan diatur sebagai proses

penyelesaian proyek. Contohnya seperti membuat buku panduan atau dokumen teknis bagi proyek yang dikerjakan, menulis kebutuhan apa saja yang diperlukan sebelum memulai proyek ke dalam sebuah dokumen untuk membantu *programmer* dalam memenuhi kebutuhan pembuatan proyek. Proses ini perlu dilakukan guna memantau dan mengontrol penyelesaian *deliverables* sebagai dokumentasi akhir dari pengembangan proyek. Para *development team* menggunakan Taiga sebagai media komunikasi, pengorganisasian proyek, serta koordinasi dalam tim.

1. Telegram

   Telegram merupakan aplikasi yang digunakan untuk bertukar pesan. Telegram digunakan oleh PT Javan Cipta Solusi untuk melakukan komunikasi dengan tim proyek seperti tim *project manager*, tim *system analyst*, tim *programmer*, individu, dan general Javan.

2. *WhatsApp*

   Whatsapp merupakan aplikasi untuk bertukar pesan sama seperti Telegram. *Whatsapp* juga digunakan pada PT Javan Cipta Solusi yaitu untuk melakukan komunikasi dengan klien, sebagai akuntabilitas kepada klien.

3. GitMind

   GitMind merupakan aplikasi untuk perencanaan, pencatatan, serta pembelajaran kolaboratif untuk tim. Javan menggunakan GitMind untuk melakukan pencatatan *progress* terhadap masing-masing tim pada saat *daily scrum* berlangsung.

4. Taiga *project management tools*

   PT Javan Cipta Solusi menggunakan kerangka kerja *scrum* dengan pengerjaan waktu setiap kali melakukan *sprint* yaitu dua minggu sekali. Adapun tahap yang dilakukan selama dua minggu itu, untuk mengawali *sprint*, dilakukannya *sprint planning* sebelum pengerjaan proyek dimulai, tujuannya untuk menganalisis pengerjaan mana saja yang akan masuk ke dalam *sprint* tersebut. Setelah melakukan *sprint planning*, *system analyst* melakukan alokasi *task* kepada anggota tim untuk dikerjakan. Adapun *sprint* yang diakhiri dengan *sprint review*, bertujuan untuk mendapatkan *feedback* dari pekerjaan yang telah diselesaikan selama dua minggu *sprint* berlangsung. *Feedback* diberikan oleh manajer dan CEO (*Chief Executive Officer*) Javan. Untuk laporan *progress* harian, biasa dilakukan pada saat *daily scrum*, tujuannya untuk memastikan tim proyek dapat mengerjakan *task* dengan maksimal, jika ada kendala akan dibicarakan pada saat *daily scrum* berlangsung. Pelaksanaan *daily scrum* dilakukan setiap hari selama 15 menit.

Selama magang, banyak berkontribusi dalam hal pembuatan *task*, membantu *project manager* dalam memaparkan *progress* dari pengembangan sistem kepada *manager* Javan dalam aktivitas *sprint planning*, *daily scrum*, dan *sprint review*. Dalam aktivitas tersebut seluruh tim mengambil andil dalam proses pemaparan hasil lingkup pekerjaannya. Adapun sebagai *system analysis* membantu *project manager* dalam proses pemaparannya ketika diperlukan, seperti membuat *slide* presentasi, menggantikan *project manager* bila berhalangan untuk memaparkan hasil pengerjaan serta hambatan dan tantangan pada saat pengerjaan selama dua minggu berlangsung. Adapun pada Gambar 3.2, merupakan dokumentasi dari aktivitas *sprint planning* yang dilanjutkan dengan Gambar 3.9, menunjukkan kegiatan *daily scrum*, yang merupakan penyampaian *progress* pengerjaan *task* selama waktu yang telah ditentukan, serta dokumentasi dari *sprint review* internal yang dilakukan pada setiap akhir *sprint* seperti Gambar 3.10.

Tabel 3.2 *Role* dan Status *Task*

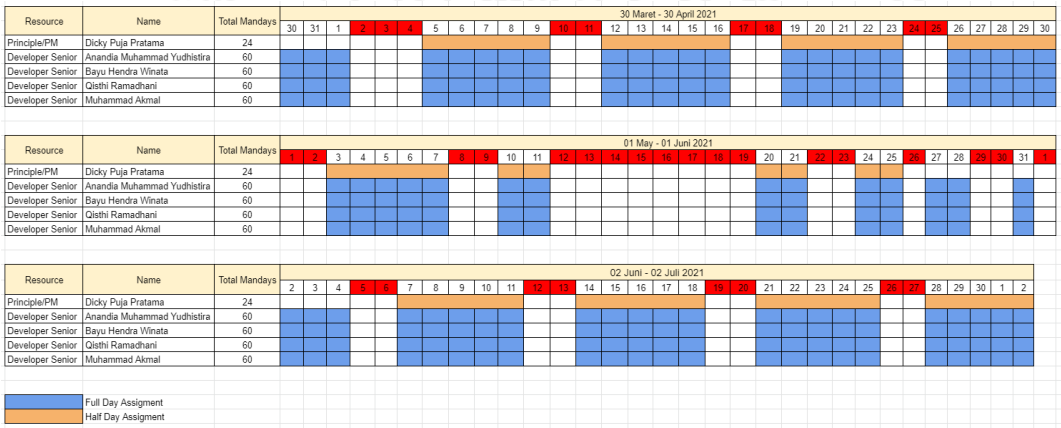
Assigned	***Label Task***	**Keterangan**
Project Manager	*Backlog*	Daftar kebutuhan proyek yang diberikan oleh klien sebelum diberikan kepada *system analyst*.
Analyst	*New*	*Task* yang telah dianalisis sesuai kebutuhan proyek dan siap diberikan kepada *engineer*.
	Needs Info	*Task* yang dikembalikan ke s*ystem analyst* dikarenakan terdapat pertanyaan maupun ketidakjelasan dalam proses analisis sehingga diperlukannya kejelasan lebih lanjut dari *system analyst*.
Programmer	*To Do*	*Task* dari hasil analisis yang telah siap dikerjakan oleh *engineer*.
	In Progress	*Task* yang sedang dikerjakan oleh *engineer*.
	Feedback	*Task* yang dikembalikan oleh *quality assurance* kepada *engineer* dikarenakan terdapat *issue/bugs* pada saat *testing*.
Code Review	*Code Review*	Senior *engineer* melakukan *code review* untuk melakukan *merge request* yang diajukan oleh tim.
	Ready to Test	*Task* telah selesai dikerjakan *engineer* dan siap untuk dilakukan *testing* oleh *quality assurance*.
QA/Tester	*In Progress*	Tester melakukan pengujian *task* yang telah dikerjakan *engineer*.
	Done	*Task* telah selesai diuji dan sudah sesuai dengan *acceptance criteria* dari kebutuhan sistem.

Untuk melakukan manajemen pada *task* memerlukan sebuah *software* yang membantu dalam mendokumentasikan segala kebutuhan sistem. Javan menggunakan Taiga sebagai *task* manajemen dalam proses dokumentasi pengelolaan pekerjaan kebutuhan sistem untuk mempermudah dalam *tracking progress* yang dikerjakan oleh tim. Setiap *progress* harus diberikankan status untuk mengetahui *task* mana saja yang sudah siap dikerjakan tim maupun yang belum siap dikerjakan oleh tim. Adapun daftar *role* dan status *task* pada Taiga dalam proyek dapat dilihat pada Tabel 3.2.

Dalam pengembangan sebuah proyek, diperlukan kerjasama dan kolaborasi antar tim untuk mencapai tujuan bersama. Sebagai s*ystem analyst*, tugasnya melakukan analisis *issue/bugs* bersama *system analyst* utama dan *quality assurance*. Pembagian tugas yang dilakukan yaitu, membantu dalam pembuatan *task* dan dokumen SDD (*Software Design Description*), membantu melakukan *testing* jika diperlukan oleh tim *quality assurance*, serta menjadi penghubung komunikasi antara klien dan pihak internal Javan.

3.3.5 Penutupan Proyek

Pada tahap penutupan proyek, ada beberapa tahapan yang dilakukan. Tahapan tersebut meliputi pembuatan dokumen teknis yang berisikan langkah-langkah instalasi *tools* untuk melakukan *monitoring* server pada PLN Devops, membuat *timesheet* sebagai bukti perhitungan pekerjaan yang sudah dilakukan tim Javan apakah tepat waktu atau melebihi batas dari perjanjian di awal pembentukan proyek, serta dokumen teknis API (*Application Programming Interface)* untuk menunjang kebutuhan dalam pembuatan sistem. Berikut pada Gambar 3.1, merupakan contoh perhitungan kerja antara kesepakatan Javan dengan PLN.



Gambar 3.1 Perhitungan kerja

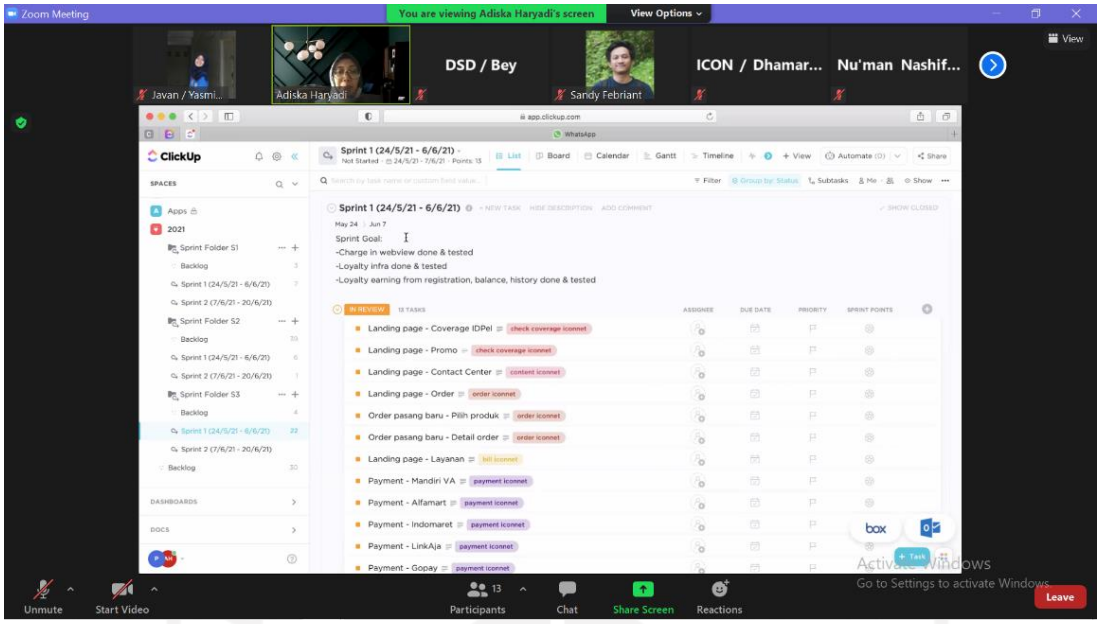
3.4 Pelaksanaan Proyek PLN Devops dan Mobile *Back end*

Dalam pelaksanaannya, proyek ini dikembangkan dengan kerangka kerja *scrum*. Secara garis besar, *scrum* memiliki beberapa tahapan dalam proses pengembangannya. Dalam penerapannya, *scrum* tim terdiri dari tiga bagian di antaranya:

3.4.1 *Sprint Planning*

Sprint planning dapat digambarkan sebagai tahapan dalam melakukan analisis masalah yang akan diselesaikan. *Sprint planning* dibuat untuk memetakan kebutuhan apa saja yang ada dan harus dipenuhi sebelum memulai pembuatan sebuah sistem. Mulai dari menguraikan ruang lingkup, menentukan *product backlog*, menguraikan pekerjaan untuk setiap anggota berdasarkan *role*, dan menentukan tujuan akhir dari *sprint* tersebut, seperti pada Gambar 3.2. *Product backlog* merupakan kumpulan dari semua *user story* yang telah disederhanakan terhadap kebutuhan sebuah sistem. Hal pertama yang dilakukan adalah melakukan identifikasi dan dilakukannya pemeriksaan terhadap kelayakan oleh *project manager*. Kemudian *task* ditambahkan ke *product backlog* oleh *project manager* ataupun *system analyst*. Setelah melalui tahap kelayakan oleh *project manager*, *user story* dipindahkan atau dipetakan ke dalam *sprint* untuk selanjutnya dieksekusi oleh *system analyst* untuk melakukan analisis lebih lanjut pada *task*. Berikut pada Gambar 3.3, merupakan gambaran *product backlog*.

Pada proyek PLN Devops dan Mobile *Back end*, semua kebutuhan sudah dipetakan ke dalam sebuah *task*. Ketika *sprint planning* berjalan, tim dan *product owner/project manager* melakukan validasi terhadap semua kebutuhan yang telah disusun apakah sudah sesuai dengan tujuan awal pembuatan atau memiliki kekurangan dalam penyusunan kebutuhan. Kemudian memastikan estimasi yang ditentukan sesuai dengan kesepakatan awal pada *kick off meeting*. *Kick off meeting* dilakukan untuk memastikan semua kebutuhan sudah sesuai dengan persetujuan kontrak yang telah dibuat, *kick off meeting* dihadiri oleh tim proyek, klien, serta pemangku kebutuhan. *Sprint planning* dilakukan pada awal minggu yang dihadiri oleh semua tim proyek, untuk memastikan kebutuhan yang telah dipetakan siap untuk dieksekusi.



Gambar 3.2 *Sprint Planning*

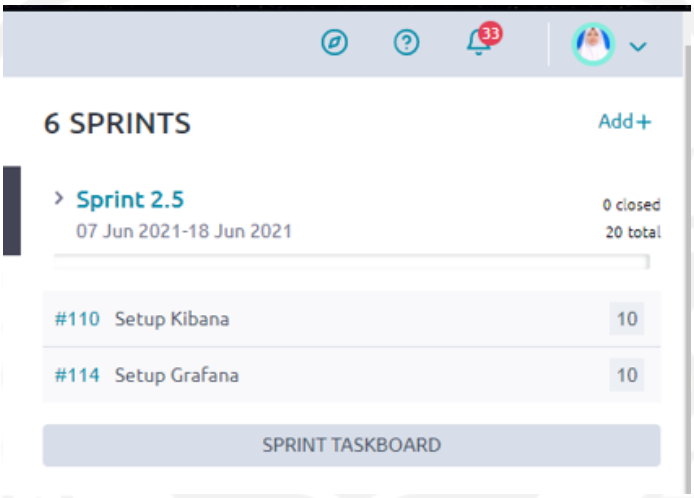
Gambar 3.3 *Product Backlog*

3.4.2 *Sprint Backlog*

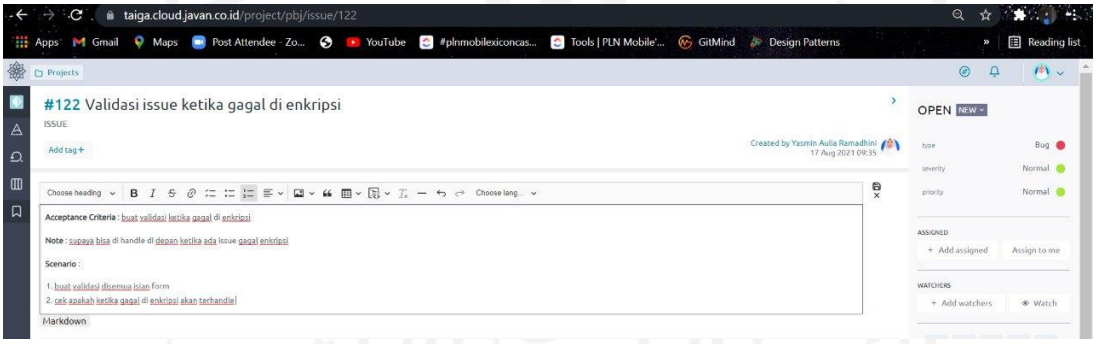
Sprint backlog berisi daftar pekerjaan atau *task* pada hasil kesepakatan *sprint planning* yang akan dikerjakan tim selama waktu satu *sprint* berlangsung. Untuk Javan sendiri, *sprint* dilaksanakan dua minggu. Pada proyek PLN Devops dan Mobile *Back end* menggunakan alat manajemen proyek Taiga. *Task* yang telah ada pada daftar *product backlog* akan dipindahkan ke *sprint backlog*. Kemudian *system analyst* memberikan tambahan informasi/ulasan lebih rinci pada setiap *task* yang ada dalam daftar *sprint backlog* seperti deskripsi *task*, *objective task*, *scenario*, alokasi *task*, dan alokasi waktu seperti pada Gambar 3.5.

Pada saat identifikasi, semua *user story* didekomposisi menjadi beberapa bagian *task* lagi, untuk memudahkan pengerjaan bagian-bagian dari sistem. *Task* yang telah selesai dibuat,

akan ditambahkan ketika akan memulai *sprint* yang sering disebut *sprint planning*. Semua *task* akan diidentifikasi pada saat *sprint planning* berlangsung, mempertimbangkan bagian mana saja yang menjadi prioritas dalam hal pengerjaan, agar semua tim dapat memahami *task* yang telah dianalisis dan disusun sebelumnya. Berikut pada Gambar 3.4 merupakan *sprint backlog* yang telah di-*mapping* untuk memulai *sprint*.



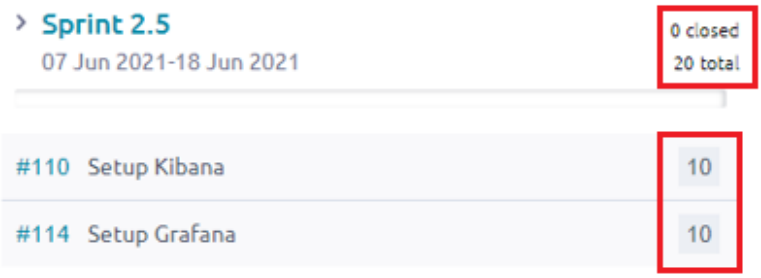
Gambar 3.4 *Sprint Backlog*



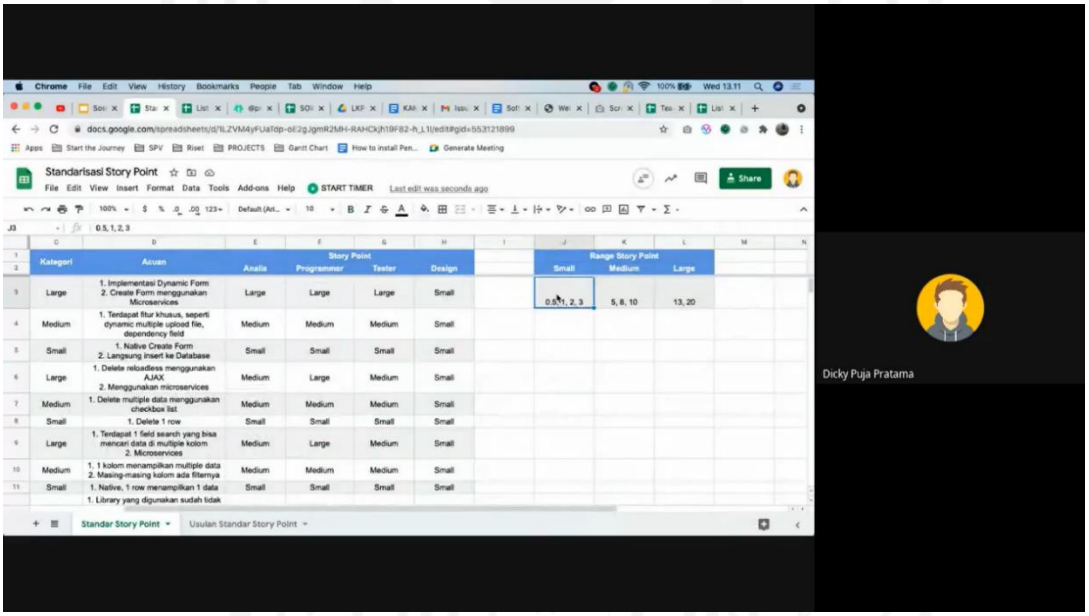
Gambar 3.5 *System Analyst* Membuat *Task*

Adapun *story point* pada *task*. Saat pembuatan *user story/task* berguna untuk menentukan tingkat *priority*/ukuran sebuah *task*, apakah termasuk ke dalam kategori mudah, sedang, dan sulit untuk dikerjakan. Penentuan *story point* biasa menggunakan bilangan *fibonacci* dari yang terkecil/termudah memiliki nilai 1 poin dan yang terbesar/tersulit memiliki nilai 10 poin, seperti Gambar 3.6, menambahkan nilai 10 poin pada *user story/task* yang sulit. Nilai tersebut digunakan untuk mengukur seberapa mudah atau sulit sebuah *task* yang akan dikerjakan oleh

tim *engineer*. Tim *junior engineer* akan diberikan *task* yang mudah, sedangkan *senior engineer* akan diberikan *task* yang sulit. Penetapan *point* untuk *story point* atas dasar kesepakatan bersama oleh PMO (*Project Manager Officer*) di Javan. Berikut pada Gambar 3.7, merupakan kegiatan sosialisasi kepada tim tentang *story point* menggunakan *Google Sheet*, untuk memudahkan seluruh tim memahami penggunaan *story point*.

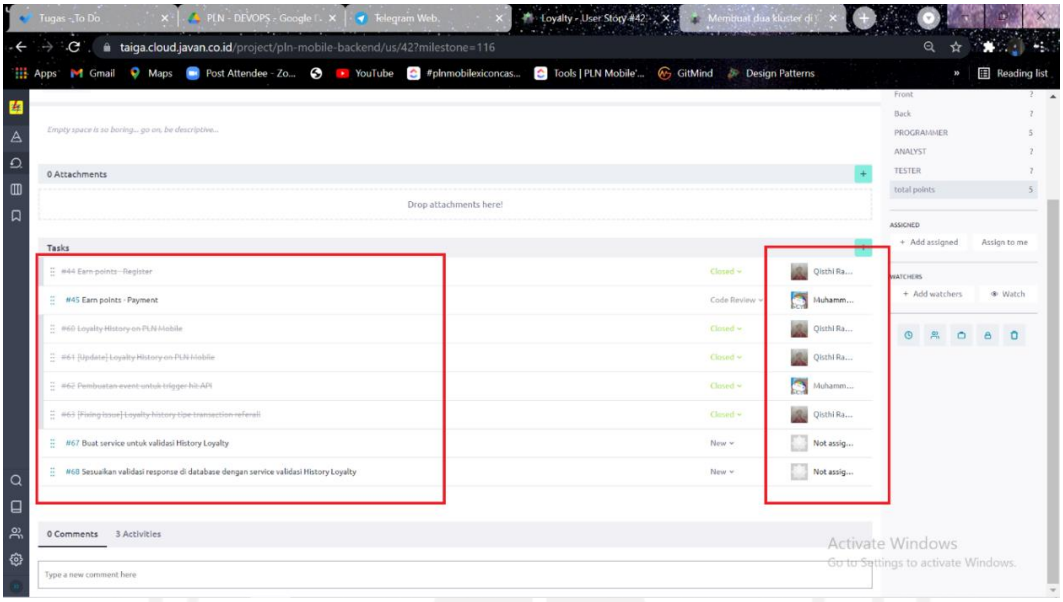


Gambar 3.6 *Story Point*



Gambar 3.7 Sosialisasi *Story Point*

Kemudian dalam melakukan alokasi *task*, harus memperhatikan *priority*/ukuran dari sebuah *task*. Alokasi *task* juga disesuaikan dengan kemampuan masing-masing tim dalam memahami *task* yang diberikan. Kemudian *system analyst* melakukan pendistribusian *task* kepada tim proyek sesuai dengan *role* yang telah dialokasikan pada awal proyek ketika dikembangkan. Berikut pada Gambar 3.8, merupakan gambaran untuk melakukan alokasi pada *task*.

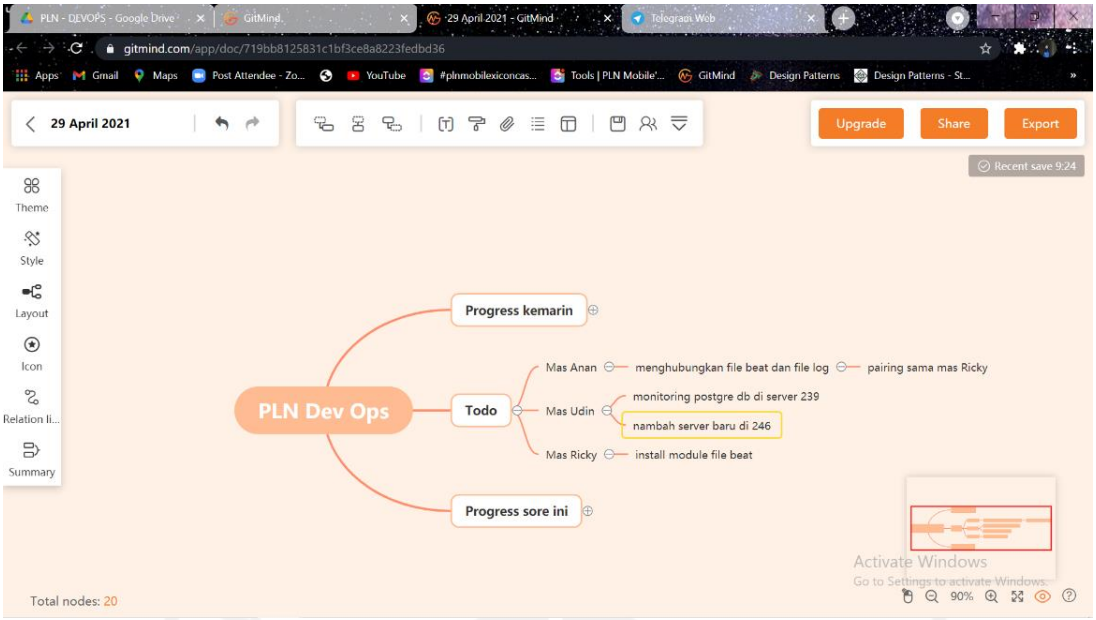


Gambar 3.8 Alokasi *Task*

3.4.3 *Daily Scrum*

Untuk memastikan apakah *sprint* berjalan sesuai dengan target yang telah ditentukan, *project manager* akan mengadakan *daily scrum* yang umumnya berdurasi 15 menit setiap pertemuannya. Pada realitanya, durasi tergantung pada jumlah anggotanya dan pembahasan. Pembahasan yang disampaikan biasanya berupa *"Apa saja yang telah dicapai kemarin", "Apa saja yang tidak tercapai kemarin? Sebutkan alasannya!", "Apa saja yang dikerjakan hari ini?",* "*Bagaimana progress saat ini?",* "*Kendala apa yang ditemukan?".*

Setelah *system analyst* menguraikan *task* pada daftar *sprint backlog*. *project manager* atau asisten *project manager* melakukan proses monitor dan kontrol melalui Taiga *project management tools* dan setiap jam 16.00 WIB, akan menanyakan *progress* dari pekerjaan yang dilakukan masing-masing tim di *Telegram*. Untuk memudahkan tim jika memiliki kendala lain dan dapat berdiskusi untuk menyelesaikan kendala yang ditemukan. Berikut pada Gambar 3.9, merupakan kegiatan *daily scrum* menggunakan Zoom dan GitMind.

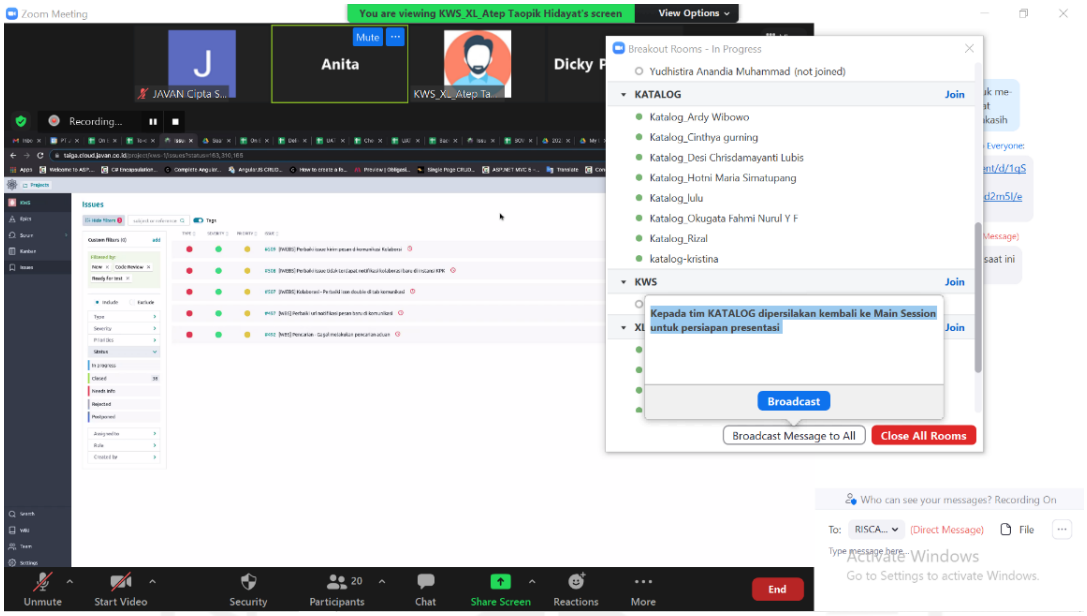


Gambar 3.9 *Daily Scrum* menggunakan GitMind

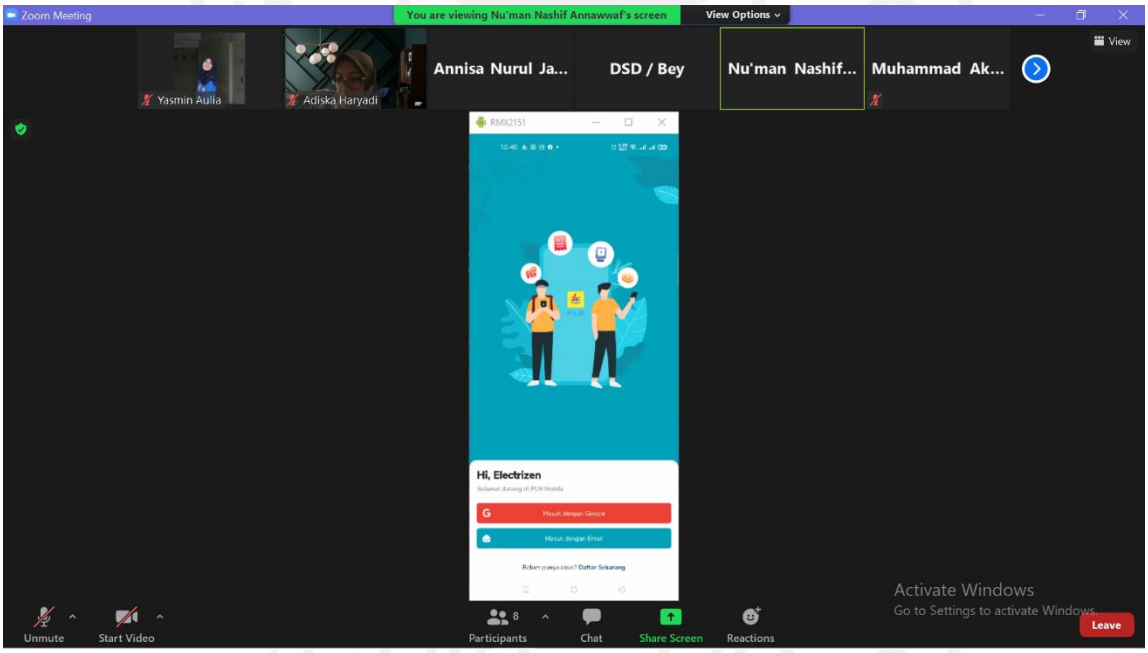
3.4.4 *Sprint Review*

Sprint review dilaksanakan pada akhir masa *sprint* berlangsung. Peserta yang mengikuti kegiatan ini adalah tim proyek dan pemangku kebutuhan. Pada Javan, *sprint review* dilakukan setiap Jumat dari pukul 08.00 sampai dengan 12.00 WIB, seperti Gambar 3.10. *project manager* memaparkan *product backlog* yang telah dan belum terselesaikan, jika terdapat kendala akan didiskusikan secara bersama-sama untuk menemukan solusi yang tepat. Karena proyek PLN Devops dan Mobile *Back* merupakan proyek kerjasama yang mengirimkan tim kepada klien, jadi tim Javan tidak memiliki akses untuk melakukan demo terhadap aplikasi. Aplikasi bisa didemokan ketika melakukan *sprint review* bersama dengan tim dari PLN, seperti Gambar 3.11, dikarenakan yang bisa melakukan akses terhadap sistem hanyalah tim PLN. Tim PLN mengadakan *sprint review* setiap hari Jumat pukul 10.00 WIB. Kegiatan yang dilakukan hampir sama dengan Javan. Setelah melakukan *sprint review*, tim Javan dan PLN melakukan *sprint retrospective* untuk melakukan evaluasi terhadap kinerja dan kendala yang ditemukan selama *sprint* berjalan.

Dalam tahap ini, semua masukan dan tanggapan akan ditampung dan dicatat. Apabila terdapat perubahan pada fitur kan dimasukkan ke dalam *sprint* selanjutnya dan akan dikomunikasikan dengan tim yang bersangkutan.



Gambar 3.10 *Sprint Review* pada Javan

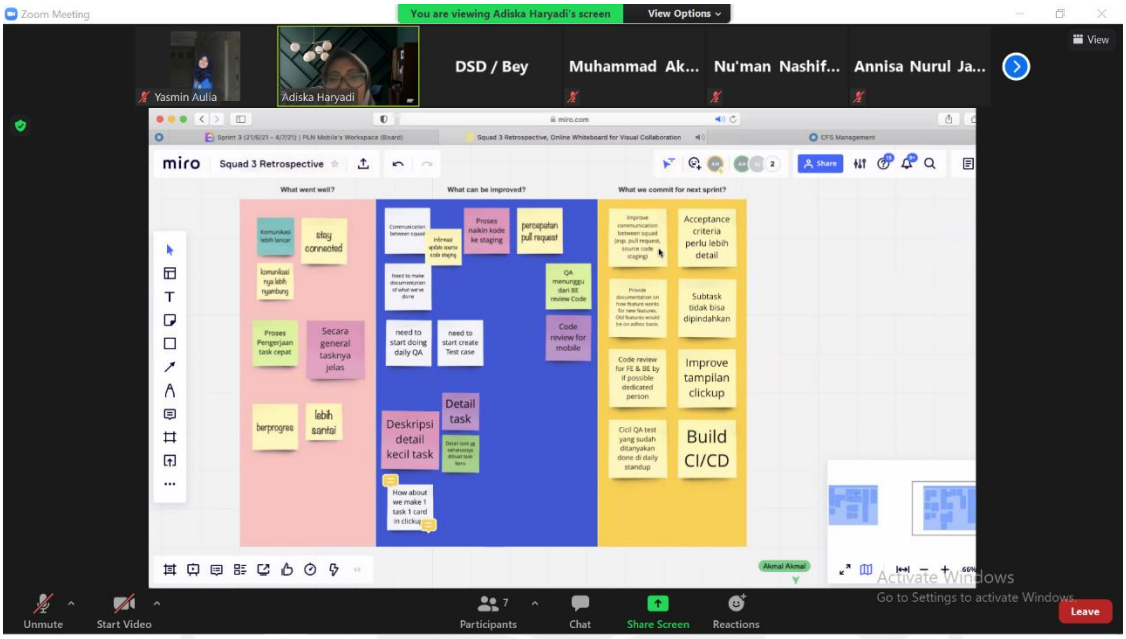


Gambar 3.11 *Sprint Review* bersama tim PLN

3.4.5 *Sprint Retrospective*

Sprint retrospective diadakan setelah melaksanakan *sprint review*. Pada tahap ini semua tim yang terlibat (Javan dan PLN) akan melakukan evaluasi terhadap kinerja selama dua minggu ke belakang. Apakah terdapat komunikasi antar tim berjalan lancar, proses pengerjaan *task* menjadi cepat, dan apakah *task* yang dibuat sudah jelas dan detail. Kemudian dari hasil

sprint retrospective bisa diterapkan pada *sprint* selanjutnya. Berikut pada Gambar 3.12, merupakan kegiatan *sprint retrospective* bersama dengan tim PLN.



Gambar 3.12 *Sprint Retrospective* bersama tim PLN

3.4.6 *Weekly Report*

Weekly report merupakan laporan mingguan dari kegiatan proyek. Laporan kegiatan proyek akan dikirimkan kepada manajemen perusahaan setiap minggu. Tujuannya agar manajemen perusahaan dapat mengetahui *progress* dari pengembangan yang telah dikerjakan. Setiap hari Kamis, *project manager/system analyst* akan mengirimkan *weekly report* kepada manajemen perusahaan melalui email, seperti pada Gambar 3.13. Pada *weekly report* memuat kendala yang ditemukan selama satu minggu, solusi apa yang dapat dilakukan untuk mengatasi kendala tersebut, dan pekerjaan/*task* apa saja yang telah selesai dikerjakan selama satu minggu. Selain itu *weekly report* juga mencantumkan *burndown chart* yang digunakan untuk melakukan pemantauan dan melihat *progress* capaian yang didapatkan selama satu minggu, seperti pada Gambar 3.14.

Gambar 3.13 *Weekly Report*

Gambar 3.14 *Burndown Chart* pada PLN Mobile *Back end*

3.5 Penerapan SOP (*Standard Operating Procedure*) pada *Task*

Sebuah *task*, idealnya berisi informasi penting yang relevan termasuk informasi yang sesuai dengan kebutuhan proyek yang telah disusun di awal pembuatan proyek, termasuk informasi tekstual dan data kategori. Data tekstual yang dimaksud berupa judul, deskripsi, *comments*, *date*, *created*, dan lainnya. Di sisi lain data kategori termasuk ke dalam prioritas, meliputi *story point*, *severity*/tingkat krusial, komponen, *reported by*, alokasi *task*, status, dan lainnya.

Sebelum membuat sebuah *task*, Javan membuat sebuah standar dalam hal pembuatan *task* gunanya memberikan standar bagi pengguna Taiga. Selain itu, standar tersebut diatur agar dapat dijadikan konsistensi data yang nantinya bisa di-*generate* menjadi *dashboard* KPI. Menurut Banerjee dan Buoti (2012), KPI atau *Key Performance Indicator* adalah skala dan indikator kuantitatif yang digunakan untuk mengevaluasi kinerja pada organisasi untuk mencapai tujuan organisasi. Penggunaan KPI bertujuan memperbaiki kesalahan dengan lebih spesifik, guna mengurangi kesalahan yang akan terjadi pada masa mendatang (Putri et al., 2019).

Pembuatan SOP berguna untuk meningkatkan pemahaman tim proyek pada saat mengerjakan *task*. Pada Javan, pembuatan SOP memerlukan diskusi yang serius untuk mendapatkan *value* dari kesepakatan bersama. Dari kesepakatan tersebut berguna bagi seluruh proyek dan menambah kecepatan dalam proses pembangunan sebuah sistem. SOP juga bisa disebut format baku yang telah dibuat oleh Javan untuk memudahkan dalam dokumentasi tujuan dari pembuatan *user story*/*task*. Namun, dalam sebuah standar pasti terdapat perubahan-

perubahan ketika standar tersebut diterapkan. Sebagai acuan khusus ketika membuat SOP adalah adanya *tools* yang berfungsi untuk membantu analis dalam menentukan deskripsi *task*. Kemudian *task* yang dibuat dapat menjawab semua pertanyaan *engineer*. Berikut merupakan SOP *Tasking* yang berlaku di Javan. Terdapat tiga komponen yang menjadi acuan pada SOP (Javan Cipta Solusi, 2021), yaitu:

a. Standar *Task*

Standar *task* dibuat bagi tim proyek untuk menyamakan tujuan/*objective* suatu *task* agar mudah dipahami. Standar muncul karena kurangnya pemahaman dari *engineer* terhadap tugas yang dikerjakan, sehingga mengharuskan suatu *task* harus memuat informasi yang lengkap sesuai dengan analisis yang telah dilakukan sebelumnya oleh *system analyst*. Dikarenakan Javan pun memiliki jam terbang kerja yang padat dan memungkinkan satu orang *multiple projects*, jadi dibuat sebuah standar *task*. Seperti Tabel 3.3 di bawah ini, menjelaskan tentang SOP *Tasking* Taiga di Javan.

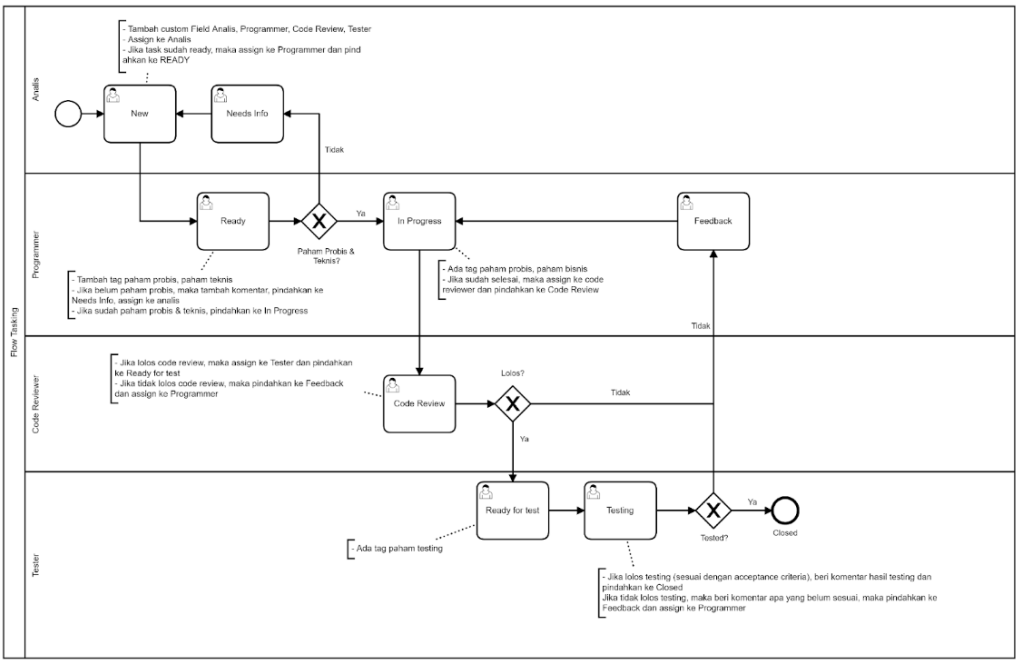
Tabel 3.3 SOP *Tasking* pada Taiga

SOP *Tasking* Taiga	
Standard Task	Penjelasan
Format Judul *Task*	1. *Standard Task* digunakan pada *task* yang general dengan format: a. <kata kerja> + **<aktivitas>** → Lakukan ***Backup*** (menjelaskan *breakdown* dari suatu *user story*). b. <kata kerja> + **<*object*>** → Buat **Halaman Beranda**; Buat **Fitur *Login***; Buat **API Registrasi**. 2. *Improvement Task* digunakan apabila *task* yang dibuat memiliki *objective* untuk meningkatkan fitur yang sudah ada dengan format: a. <*object*> + **juga** + <Tambahan Fungsi> → Fitur *Login* **juga** dapat menggunakan SSO. b. Buat + **<fitur>** menjadi lebih cepat → Buat **fitur *download* laporan *via* jasper** menjadi lebih cepat. c. Tingkatkan *performance* dari + **<fitur>** → Tingkatkan *performance* dari **KTR *Backup* Data**. d. *Update* + **<fitur>** + dengan/menjadi + <*update*> → *Update* **Fitur Registrasi Karyawa**n menjadi Registrasi Karyawan dan Magang. e. *Rename* + **<fitur/*text*>** + menjadi + <nama baru> → *Rename* ***wording* Registras**i menjadi Pendaftaran.

	3. *Bugs Task* digunakan apabila membutuhkan *task* untuk menyelesaikan *bugs* dengan format: a. <*role*> + **tidak dapat** + <*action*> → Admin **tidak dapat** menambahkan *user* baru. b. **Ketika** + <*role*> + <aksi yang dikerjakan> + , + <fitur> + **tidak bekerja** → **Ketika** admin sedang *login* dengan *username* yang salah, validasi **tidak bekerja**. c. <Fitur> + **tidak berjalan** → Fitur notifikasi **tidak berjalan**. d. <Fitur> + **seharusnya** + <ekspektasi> + **tetapi tidak** → Fitur *chatting* **seharusnya** *realtime* **tetapi tidak**. e. <FItur> + **tidak** + <ekspektasi> → Fitur *login* **tidak** menampilkan *captcha*. f. <*Roles*> + <hasil yang didapatkan> + **tetapi seharusnya** + <ekspektasi> → Verifikator tidak dapat membatalkan verifikasi, seharusnya dapat membatalkan verifikasi ketika masih dalam tenggat waktu verifikasi.
Deskripsi	Apabila ada informasi/deskripsi tambahan yang diperlukan, dapat dituliskan juga di dalam *task*.
Skenario	1. Menjelaskan langkah-langkah untuk mencapai suatu *objective*. 2. Skenario dibuat menggunakan *numbering*, dan dibuat secara sistematis/terurut dan jelas. 3. Sertakan gambar bila dibutuhkan. 4. **Wajib** mencantumkan URI FE/BE.
Existing Condition	1. Menjelaskan kondisi saat ini yang ingin diubah. 2. Sertakan gambar bila dibutuhkan.
Acceptance Criteria	1. Boleh dibuat menggunakan *numbering/bullets*. 2. Menjelaskan hal-hal apa saja yang harus dikerjakan. dan terpenuhi pada *task* tersebut. 3. Sertakan gambar bila dibutuhkan.
Due Date	1. *Task* yang dibuat wajib memiliki *due date*. *Task* sudah harus *ready* maksimal H-1 dan *due date* jam 8 pagi. Contoh: *Task ready* tanggal 9, jam 8 pagi. Maka *due date*-nya adalah tanggal 10 atau boleh lebih lama lagi. 2. *Due date task* yang dibuat tidak boleh melebihi *due date user story* dan *due date sprint*.
Assignee	Tambahkan *assignee* sesuai dengan status *task* dan *mapping* di *custom fields*.

b. *Flow Task*

Flow task digunakan untuk menunjukkan alur pembuatan *task* agar semua analis dalam proyek memiliki pemahaman yang sama dalam pembuatan *task*. Berikut merupakan Gambar alur *task* yang dimiliki oleh Javan. Berikut pada Gambar 3.15, merupakan *flow* pada *task*.



Gambar 3.15 *Flow Task*

Sumber: Javan Cipta Solusi (2021)

c. Standar Status *Task*

Dalam SOP *tasking*, terdapat standar yang mengatur status dari sebuah *task*. Gunanya sebagai peringatan bahwa *task* sedang dalam pembuatan, *task* sedang dikerjakan, dan *task* telah selesai dikerjakan. Berikut merupakan tahapan-tahapan dalam alur dari status *task*. Kemudian pada Tabel 3.4, merupakan *Standard* Status pada *Task*.

Tabel 3.4 Standar Status pada *Task*

Role	***User Task***	**Deskripsi**
System Analyst	*New*	Setiap *task* dibuat harus berelasi dengan *user story*. Pada saat pembuatan *task*, analis menambahkan: 1. *Custom fields* analis, *programmer, code reviewer* dan *tester*. 2. Melengkapi komponen pada *task* sesuai dengan kategori *task* berikut ini,

Kategori *Task*	**Komponen**
Feature	Siapa aktor yang akan menggunakan fitur ini?
	Bagaimana tampilan yang diharapkan?
	Bagaimana *user story*/deskripsi dari fitur ini?
	Database/table terkait
	Apa URL API yang digunakan?
	Skenario *testing/acceptance criteria*
Bug	Siapa yang menemukan/ melaporkan *bug* ini?
	Screenshot bug
	Bagaimana skenario hingga terjadi *bug*?
	Apakah *bug* ini *blocking*?
	Apa pesan *error*
	URL sentry/Postman
	Acceptance criteria/definition of done
Support DevOps	Apa kebutuhannya?
	Credential akses ke server
	Acceptance criteria/definition of done
Enhancement	Siapa yang menyarankan *enhancement* ini?
	Apa *goal* yang ingin dicapai dari *enhancement* ini?
	Bagaimana tampilan yang diharapkan?
	Bagaimana *user story* dari fitur ini?
	Database/API terkait
	Skenario *testing*/*acceptance criteria*
	Bagaimana kondisi saat ini?

Apabila *task* sudah **siap dikerjakan oleh programmer,** maka:

		1. *System analyst* menambahkan ***due date***. 2. *System analyst* memindahkan *task* dari *New* ke ***Ready***. 3. *System analyst* ***assign*** ke ***Programmer***.
Engineer	*Ready*	Saat *task ready*, 1. *Programmer* melakukan validasi pemahaman *task* dengan menambahkan *tag* **paham probis (proses bisnis)** dan **paham teknis**. 2. Apabila *programmer* **belum paham** proses bisnis pada *task,* maka kembali letakkan *task* pada posisi ***Needs Info***, *assign task* kepada *system analyst* dan memberi komentar penjelasan apa yang belum dipahami.
Engineer	*In Progress*	Apabila *task* yang di-*assign* sudah siap (paham probis dan paham teknis) untuk dikerjakan, maka pindahkan *task* dari ***Ready*** ke ***In Progress.*** Jika *task* sudah selesai dikerjakan, pindahkan *task* ke ***Code Review***. Lalu Programmer melakukan *assign* ke *Code Reviewer*. Pada *case* tidak ada *code review* maka *task* tidak masuk *code review*, sehingga dari status *In Progress* dipindahkan menjadi *Ready for Test*.
Code Reviewer	*Code Review*	Apabila **Lolos** *Code Review, Code Reviewer* akan di-*assign task* ke *Tester* dan *Task* dipindahkan ke *Ready for Test*. Jika tidak **Lolos** *Code Review, Code Reviewer* mengembalikan *task* tersebut ke ***Feedback***. Lalu melakukan *assign* ke *Programmer* yang mengerjakan sebelumnya.
QA/Tester	*Ready for Test*	Semua *task* yang siap diuji terkumpul pada *task Ready for Test*. Pada tahap ini *tester* memastikan, 1. Pemahaman *task* dengan pemberian *tag* 'Paham *Testing* '. 2. Memastikan *assignee* dan *custom fields tester* sudah sesuai *assignment*.
QA/Tester	*Testing*	Apabila *Tester* sudah memahami *task* dan akan melalui proses uji coba, pindahkan *task "Ready for Test"* menjadi "*Testing*". *Task* **Lolos** *Testing* ketika hasil yang diharapkan sudah **memenuhi** *acceptance criteria*. Apabila **Lolos** ***Test***, maka 1. Berikan komentar pada *task*, berupa a. Ceklis yang dites sesuai *acceptance criteria*. b. Gambar yang mempresentasikan hasil *testing*.

		2. Pindahkan status *task* ke *Closed*. Jika **Tidak Lolos *Test***, maka 1. Pindahkan *task* ke ***Feedback***. 2. Beri komentar pada *task*, yang meliputi: a. *Existing condition* (*issue* yang muncul). b. Akun, URL, *Role* yang digunakan. c. Skenario yang digunakan. d. Gambar yang mempresentasikan *issue*. e. Masukkan kembali a*cceptance criteria*. 3. *Reassign* ke *programmer* yang telah mengerjakan sebelumnya.
Engineer	*Feedback*	*Task* yang masih memiliki *issue* akan diletakkan oleh *tester* pada kolom *Feedback*. *Task* yang masuk ke *feedback* menjadi prioritas dalam pengerjaan oleh *programmer*. *Programmer* wajib memindahkan *task* dari *Feedback* ke *In Progress* apabila *task* sedang dieksekusi.
System Analyst	*Needs Info*	Pada *task Needs Info* akan ada diskusi/pembahasan sampai *task* tersebut menjadi jelas dan dapat dikerjakan. Jika *system analyst* belum dapat menjawab dan atau mendetailkan *task. Task* akan dipindahkan dari ***Needs Info*** ke ***New*** untuk dianalisis lebih lanjut.

Peningkatan dan perubahan akan terjadi pada SOP *task* tergantung dari proses pemahaman dan kejelasan informasi yang diinputkan ke dalam *task* manajemen yang dilakukan oleh analis. Meskipun standar telah dibuat, pemakaian kata yang bertele-tele dan tidak beraturan juga dapat menghambat pemahaman *engineer* terhadap *task* yang telah dibuat. Analis tinggal menyesuaikan kategori atau tujuan apa yang diingin dibuat pada *task*, seperti *Feature, Bug, Support DevOps,* dan *Enhancement/Improvement*.

3.5.1 Perbandingan Kriteria *Agile Project Management Tools* Taiga dan Jira

Perbandingan dilakukan terhadap kriteria dari *agile project management* Taiga dan Jira, untuk menemukan apakah ada perbedaan yang dapat dilakukannya improvisasi *task* pada indikator dan atribut yang akan memengaruhi informasi apa saja yang bisa dimuat atau sebagai tambahan baru dalam membuat *task*. Seperti yang diketahui sebelumnya, Javan sendiri berkiblat pada pendekatan dengan *best practice scrum* dan *agile*. *Software management tools* yang paling mendekati adalah Taiga, karena Taiga memiliki *sprint, user story, story point* dan *burndown chart* sebagai komponennya. Namun ada satu lagi *task management* yang hampir

mirip dengan Taiga, yaitu Jira. Jira sendiri juga memiliki *sprint, user story/task, story point,* serta *burndown chart*. Untuk dapat menghasilkan perbandingan yang baik, harus mempertimbangkan beberapa kriteria dan aspek kepuasan dari *tools* (Taheri & Sadjadi, 2015). Berikut Tabel 3.5 yang merupakan perbandingan kriteria *agile project management tools*. Perbandingan dilakukan dengan menitikberatkan indikator yang digunakan oleh Javan.

Tabel 3.5 Perbandingan Kriteria *Agile Management Tools*

Kriteria	***Agile Project Management Tools***	
	Taiga	**Jira**
Mengadopsi *scrum&agile*	Memiliki *sprint, user story, task/subtask, story point, burndown chart* pada level *sprint* dan proyek.	Memiliki *sprint, task/user story/bug, story point, burndown chart.*
Kemudahan penggunaan	Ketika membuat *user story* akan otomatis dapat membuat *task* baru yang menjadi satu kesatuan dalam pembuatannya.	Ketika membuat sebuah *task/issue* dapat terintegrasi dengan *task* lainnya, sehingga mudah melakukan *tracking task/issue*.
Harga	Taiga memiliki harga kisaran $7/bulan untuk per *feature*-nya atau sekitar seratus satu ribu rupiah.	Jira memiliki harga kisaran $7/bulan untuk per *user*-nya atau sekitar seratus satu ribu rupiah.
Ketersediaan laporan	Taiga menyediakan *soft reporting* berupa *timeline, Kanban zoom level, epic* dengan *multi project support* atau dengan melihat *activity task*. Dapat juga berupa CSV *report*.	Jira menyediakan empat bagian pada *burndown chart level sprint, Kanban project* sepeti *control chart* dan perhitungan *flow diagram*, serta *general report* untuk analisis *issue*, seperti *pie chart.*
Kustomisasi sesuai kebutuhan	Taiga menyediakan *scrum* dan *Kanban* yang dapat dikustomisasi sesuai kebutuhan. Mudah dalam *drag and drop.*	Jira mempunyai *scrum boards* yang dapat dikustomisasi dan *Kanban boards* yang fleksibel. Mudah dalam *drag and drop.*
Terintegrasi dengan sistem lain	Terintegrasi dengan *repositories* seperti: Github, Gitlab, Bitbucket. Selain itu, terintegrasi dengan Metabase dan Data Studio sebagai visualisasi data. Taiga juga menggunakan Rest API.	Terintegrasi dengan Rest API, kemudian Slack, Zoom, Github hingga aplikasi yang siap digunakan seperti adobeXD for Jira, Figma, dan lainnya.

Setelah melakukan perbandingan, didapatkan indikator dan atribut dari Taiga dan Jira. Pada Tabel 3.6, beberapa memiliki kesamaan tapi ada beberapa juga yang berbeda. Seperti

halnya dalam penamaan di Jira memiliki *Story/Task/Bug*, dan Taiga memiliki hanya memiliki penamaan berupa *user story* dan *task* memiliki bagian yang terpisah lagi di dalamnya.

Tabel 3.6 Indikator dan Atribut

Indikator dan Atribut *Task*	
Taiga	**Jira**
a. Backlog *b. User Story* i. Judul *User Story* *ii. Tagging* *iii. Story Point* iv. *Task* 1. Judul *Task* *2. Description* *3. Tagging Task* *4. Objective Task* *5. Custom Fields* *6. Assignee* *7. Due Date* *8. Comment Section/Activities* *9. Watchers* *10. Attachments*	*a. Backlog* *b. Story/Task/Bug* i. Judul *ii. Description* *iii. Objective Task* *iv. Status* *v. Assignee* *vi. Labels* *vii. Story Point estimate* *viii. Comment Section/History* *ix. Attachments*

BAB IV
REFLEKSI PELAKSANAAN MAGANG

4.1 Relevansi Akademik

Menurut Jyothi & Parkavi (2016), menjelaskan bahwa sebuah *task* harus ditentukan prioritasnya terlebih dulu. Seseorang harus memiliki pemahaman jelas tentang tujuan dan sumber daya yang tersedia. Pada kenyataan yang terjadi ketika magang berlangsung terdapat perbedaan yang terjadi. Dalam proses menentukan prioritas sebuah *task*, masih mempertimbangkan beberapa hal seperti: task yang tidak terlalu *urgent* harus tetap dikerjakan, sedangkan pada waktu melakukan *sprint review* kepada klien, terdapat beberapa perubahan fitur yang memaksa tim proyek untuk melakukan *enhancement*. Perubahan tersebut membuat tim merasa bingung *task* manakah yang harus dikerjakan terlebih dahulu.

Javan mencoba menerapkan metode *scrum* dan *agile* dalam proses *task management*-nya. *Product backlog* yang sudah ditetapkan untuk *sprint* merupakan keputusan dari tim *development* sepenuhnya (Schwaber & Sutherland, 2020). Dalam implementasinya, *product backlog* yang akan dimasukkan pada *sprint* masih merupakan keputusan dari *project manager* dan *system analyst*. Menurut *Srum Guides* (2020), ketika ada pekerjaan baru yang diperlukan, maka tim *development* yang akan menambahkan ke dalam *sprint backlog*. Dalam penerapannya, ketika ada pekerjaan baru yang diperlukan, maka *system analyst* yang akan menambahkan ke dalam *sprint backlog*. *Daily scrum* adalah acara yang dilakukan 15 menit untuk pada *developers* dalam *scrum team* (Schwaber & Sutherland, 2020). Pada kenyataannya masih melebihi 15 menit tergantung pada pembahasan yang disampaikan oleh tim proyek.

Pengetahuan akan metode *scrum* dan *agile* menjadi salah satu pengetahuan yang berharga bagi penulis, sangat mudah digunakan dan dapat mempercepat proses penyelesaian sebuah proyek. *Scrum* melibatkan kelompok yang memiliki keterampilan dan keahlian untuk melakukan pekerjaan atau memperoleh keterampilan yang dibutuhkan tim proyek (Schwaber & Sutherland, 2020). Ada baiknya materi tentang pengembangan perangkat lunak lebih diulas pada perkuliahan, agar ketika mahasiswa terjun dalam dunia kerja akan lebih familiar terhadap proses pengembangan perangkat lunak. Pengembangan perangkat lunak bukan hanya *agile* saja tetapi banyak metode lainnya seperti *waterfall* dan metode lainnya.

4.2 Pembelajaran Magang

Terkait pembelajaran pada magang, disajikan lebih mengarah kepada aspek keilmuan terkait manfaat dalam bentuk teknis dan non teknis, kendala, hambatan, dan tantangan yang diperoleh selama magang berlangsung.

4.2.1 Manfaat Magang dalam Bentuk Teknis

Terdapat dua manfaat yang didapatkan selama magang di PT Javan Cipta Solusi. Pertama, mahasiswa mendapatkan pengalaman tentang bagaimana membuat sebuah *task* dalam sebuah proyek dengan menggunakan bantuan *software* manajemen proyek Taiga. Kedua, Perusahaan dapat melakukan improvisasi dalam pembuatan *task* sesuai kebutuhan dan kepentingan dalam pembangunan sebuah proyek. Dari pembuatan *task* yang dilakukan, membuahkan sebuah indikasi dan atribut dari perbandingan yang dilakukan antara Taiga dan Jira serta improvisasi *task* yang didapatkan setelah melakukan perbandingan tersebut. Dari perbandingan yang telah dilakukan pada pembahasan sebelumnya, dapat dilihat pada Tabel 4.1, didapatkan atribut dan indikator yang terdapat pada Taiga dan Jira.

Tabel 4.1 *Checklist* Indikator dan Atribut *Task*

Indikator dan Atribut	***Agile Project Management Tools***	
	Taiga	**Jira**
Backlog	√	√
User Story/Task	√	√
Story Point	√	√
Judul	√	√
Tagging/Labels/Status	√	√
Custom Fields	√	**X**
Assignee	√	√
Due Date	√	√
Comment Section/History	√	√
Watchers	√	**X**
Attachments	√	√

Dapat ditemukan dari indikator dan atribut di atas pada Jira tidak ditemukannya "*Watchers*" yang digunakan untuk melakukan pemantauan terhadap *task* yang dikerjakan yang biasa disebut dengan melakukan *tracking task*. Kemudian pada Jira tidak ditemukan *Custom Fields*, fitur tersebut berguna untuk mencantumkan nama dari tim yang mengerjakan *task*, berfungsi sebagai pertanggungjawaban ketika ada *issue* maupun memerlukan informasi yang lebih lengkap lagi. Kemudian setelah mendapatkan perbedaan dari perbandingan atribut dan indikator pada kedua

software manajemen proyek tersebut, dapat dilakukannya improvisasi pada *task*. Improvisasi pada *task* dilakukan untuk membantu memaksimalkan penggunaan *tools*. Improvisasi dapat memuat informasi detail untuk menunjang pembuatan *task* yang sebelumnya tidak tersampaikan secara maksimal, sehingga informasi tersebut dapat bermanfaat bagi tim proyek dan mempercepat *closing* dalam sebuah proyek, dilakukan improvisasi pada *task* sebagai pada Tabel 4.2, berikut:

Tabel 4.2 Improvisasi *Task*

Kriteria	**Keterangan**
Judul *task*	Diterapkan pada saat membuat judul *task*. 1. *General Task*, digunakan pada *task* yang umum, bisa juga digunakan untuk memulai sebuah *task* yang baru. b. *Improvement Task*, digunakan ketika *task* yang dibuat memiliki perubahan terhadap *objective*, sehingga diperlukannya peningkatan. c. *Bugs/Issue Task*, digunakan ketika menemukan sebuah *bugs* atau *issue*.
Deskripsi *task*	Informasi yang dapat dimasukkan ke dalam deskripsi *task*, seperti *definition of done* dan deskripsi dari pembuatan *task* itu sendiri.
Objective task	Berisi informasi berupa tujuan dari keseluruhan dan garis besar pembuatan *task*, *1. Acceptance Criteria* *2. Scenario* *3. Existing Condition*
Assignee/Alokasi *task*	Harus ditentukan pada saat melakukan *sprint planning* berlangsung, untuk memudahkan pengerjaan ketika dimulainya *sprint*.
Due date	*Due date* juga di-*set* ketika *sprint planning* berlangsung, guna membuat tim *aware* terhadap tanggung jawab yang sudah diberikan.
Tagging/Labels	*Tagging/Labels* berguna ketika melakukan *filtering* sesuai dengan format masing-masing proyek (status, *urgency*, kategori *task* dan *role*).
Custom Fields	Berguna untuk memberitahukan, siapa saja yang bertanggung jawab terhadap *task* yang dikerjakan, memuat nama tim dan *role* dari tim. Untuk memudahkan *tracking* jika *task* mengalami hambatan.

Dari analisis yang telah dilakukan, ditemukan beberapa hasil setelah melakukan improvisasi pada *task*, sebagai berikut:

a. Setiap proyek dapat menentukan kriteria dari masing-masing pembuatan *task*, contohnya seperti SOP (*Standard Operating Procedure* pada *task*).

b. Dengan kriteria, dapat membantu percepatan sebuah *task*, memperjelas tujuan *task* menjadi lebih lengkap dan rinci.
c. Improvisasi *task* membantu tim dalam memberikan informasi sesuai tujuan dari proyek yang dikembangkan.
d. Improvisasi *task* dilakukan untuk memaksimalkan kebutuhan yang ada menjadi lebih baik dan stabil.
e. Dapat membantu menyelesaikan *sprint* dengan tepat waktu, bahkan lebih cepat dari perkiraan.
f. Dapat meningkatkan kolaborasi dan *awareness* tim pada saat pengerjaan *task.*

4.2.2 Manfaat Magang dalam Bentuk Nonteknis

Keahlian/Keterampilan

Dalam hal ini, keahlian atau keterampilan adalah hal utama yang dipelajari ketika magang berlangsung. Keahlian tersebut dibutuhkan untuk menyelesaikan tugas yang diberikan selama magang sebagai seorang asisten *project manager*, *system analyst*, dan *quality assurance*. Selama magang berlangsung, tugas yang diberikan adalah melakukan komunikasi secara langsung dengan klien sebagai bukti akuntabilitas, kemudian melakukan konfirmasi terkait *requirement* maupun kebutuhan terhadap proyek yang dikerjakan, serta melakukan manajemen terhadap pembagian *task* kepada tim teknis. Kemampuan tersebut dapat menambah *softskill* dalam berinteraksi dengan orang baru, memperluas relasi, serta mendapatkan hal dan ilmu baru yang belum pernah ditemukan sebelumnya. Selain itu saya juga melakukan *testing* terhadap proyek yang dalam tahap *maintenance*, melakukan analisis *issue/bug* yang ditemukan, dan melakukan *testing* terhadap *issue/bug* yang telah di-*reproduce* pada server Javan, terakhir memastikan tidak ada *step* yang tertinggal maupun terlewati serta memastikan produk yang dikembangkan dapat berjalan sesuai kebutuhan klien. Manfaat lainnya adalah melatih ketelitian ketika melakukan pengujian produk, menjalin komunikasi sesama tim maupun klien supaya tidak terjadi kesalahpahaman dalam proses pembangunan proyek.

Manajemen Diri dan Tanggung Jawab

Selama magang, banyak terlibat dalam beberapa proyek yang ada di Javan. Ketika diberikan sebuah *task*, penulis harus mampu menyelesaikan tanggung jawab dan dapat melakukan manajemen diri dengan baik, supaya hasil yang didapatkan terlaksana dengan baik. Sebelum resmi magang di Javan, setiap mahasiswa diminta untuk membuat sebuah rencana

dimulai dari awal mengikuti magang, hingga menyelesaikan magang. Apa saja tujuan yang akan dicapai, apakah setelah magang tujuan yang telah dibuat sebelumnya bisa tereksekusi dengan sesuai, apakah kita dapat menyelesaikan *task* yang diberikan dengan sesuai harapan. Selain itu, tidak lupa juga untuk melakukan manajemen diri terhadap tugas-tugas yang diberikan oleh kampus, dan berusaha memaksimalkan diri untuk bisa mengerjakan kewajiban yang telah diberikan.

Komunikasi dan Relasi

Dalam proyek, juga banyak melakukan komunikasi secara langsung *via* Google Meet/Zoom dan secara tidak langsung *via* Telegram/WhatsApp. Belajar bagaimana caranya untuk dapat menyampaikan pendapat, memberikan informasi yang disampaikan oleh klien dengan sesuai, menghargai pendapat anggota tim maupun lingkungan perusahaan, menyampaikan kendala yang ditemukan pada saat melakukan *testing*, serta membantu tim jika ada yang kesulitan. Terlepas dari semua itu, komunikasi juga dapat menambah relasi dengan orang lain seperti klien, jika kita memiliki sikap dan komunikasi yang sesuai pada tempatnya, relasi akan mengikuti dengan sendirinya.

Kepercayaan Diri

Dalam perjalanan magang, banyak perkembangan yang didapatkan, mulai dari berani mengemukakan pendapat di depan rekan-rekan magang, tim, maupun karyawan lainnya. Pada ‘Hari Berkualitas’ yang dilakukan pada setiap hari Jumat di Javan, tentu banyak mendapatkan hal baru, selain berani berbicara di depan banyak orang, juga mendapatkan pandangan yang berbeda dari hasil diskusi yang dilakukan oleh rekan-rekan yang memang ahli dalam bidangnya, membuktikan bahwa keyakinan dan kepercayaan diri itu membuahkan sebuah hasil untuk dapat bertukar pikiran dengan elemen yang ada di Javan.

4.2.3 Hambatan dan Tantangan

Selama magang berlangsung, hambatan pasti selalu ada dalam bekerja. Hambatan yang dirasakan adalah ketika WFH (*Work From Home*). Sangat merasa kesulitan dalam berkomunikasi pada awal dimulainya magang, dikarenakan komunikasi yang dilakukan tidak secara langsung/tidak bertatap muka. Ada satu budaya yang diterapkan oleh Javan, yaitu ketika ada masalah atau hal yang ingin ditanyakan tidak boleh *chat* personal dan harus di grup supaya teman-teman yang lain tahu dan bisa membantu. Karena masih baru dan pemula, ada rasa

sungkan dan malu untuk bertanya di grup, ketika ingin bertanya di grup merasa harus mempersiapkan kalimat yang baik, itu memakan waktu yang dimiliki hingga banyak terbuang.

Selain itu di proyek PLN Devops dan PLN Mobile *Back end* ini, diberi amanah sebagai asisten *project manager*, yang sebelumnya belum memiliki pengalaman terhadap bidang itu, dan sedikit kesulitan. Sebab ketika menjadi asisten *project manager*, kita harus mengetahui semua hal yang berhubungan dengan proyek yang kita pegang, harus bisa mengatur tim supaya mendapatkan *task* yang seimbang dan melakukan komunikasi dua arah dengan klien. Ketika itu, masih merasa ragu dalam hal memberikan masukan maupun pendapat. Tapi dari sanalah mencoba dan berusaha memberanikan diri untuk mempelajari hal yang belum pernah ditemui sebelumnya.

Pada proyek lain seperti PBJ dan E-Katalog, juga mengalami hambatan dan tantangan tersendiri, mulai dari pemahaman tentang *issue/bug* yang dilaporkan oleh klien, melakukan *testing* jika *issue/bug* telah *solved*, dan membuat dokumen *deliverable* untuk bukti fisik pengerjaan proyek tersebut. Terkadang agak lambat untuk memahami *task* yang diberikan, tetapi dari situlah banyak melakukan *explore* dan bertanya jika memang tidak memahami *task* yang diberikan.

4.2.4 Kontribusi Selama Magang

Selama magang terlaksana, terdapat kontribusi yang diberikan kepada proyek yang telah diikuti dalam enam bulan terakhir, dimulai dari pertengahan bulan Maret 2021 hingga akhir bulan September 2021, proyek tersebut adalah PLN Devops & PLN Mobile *Back end*, PBJ-LKPP, dan E-Katalog. Dari proyek tersebut, banyak mengikuti berbagai macam kegiatan yang dilakukan sesuai *timeline* kegiatan. Pekerjaan tersebut dilakukan secara maksimal untuk mencapai hasil yang terbaik untuk perusahaan. Adapun beberapa kontribusi yang telah dilakukan selama magang berlangsung, dapat dilihat pada Tabel 4.3.

Tabel 4.3 Kontribusi Magang

No	Proyek	Aktivitas	Kontribusi
1	PLN Devops & Mobile *Back end*	Asisten PM dan Analis	Menerima limpahan tugas dari *project manager*, membantu *project manager* dalam hal komunikasi dengan klien, membuat *task* untuk tim *engineer* yang kemudian dieksekusi, dan melakukan validasi apakah *task* yang dikerjakan telah selesai atau belum. Selain itu melakukan *meeting* dengan klien, seperti *sprint planning, daily scrum, sprint review* dan *sprint*

			retrospective untuk menjaga hubungan antar tim dan komunikasi saat membangun sebuah sistem pada proyek tersebut.
2	PBJ - LKPP	Analis dan Tester	Ketika *user* mendapatkan sebuah *issue/bug*, akan dilakukan analisis *issue/bug* yang ditemukan dengan cara melakukan *reproduce* pada *server cloud* Javan, jika sudah ditemukan analis akan melaporkan kepada *engineer* untuk melakukan *fixing issue/bug*, jika sudah dilaksanakan oleh *engineer*, tester akan melakukan *testing* terhadap *issue/bug* sampai benar-benar *solved*, terakhir melakukan konfirmasi kepada klien bahwa *issue/bug* tersebut telah selesai ditangani. Jika *issue/bug* telah selesai dan tidak ada laporan keluhan, *project manager* akan meminta klien untuk diadakannya *meeting* dalam hal konfirmasi akan ditutupnya masa *maintenance issue/bug*.
3	E-Katalog	*Technical Writer*	Banyak membantu analis utama dalam pembuatan dokumen *deliverable*, membuat *task* untuk tim teknis untuk memudahkan eksekusi dalam proyek. Jika diperlukan, akan siap membantu melakukan *testing* terhadap *web* yang sedang dikembangkan dalam proyek E-Katalog dan melakukan *meeting sprint review* dengan klien sebagai bukti dan *progress* dalam pengerjaan proyek.

4.2.5 Evaluasi oleh *Supervisor*

Evaluasi sangat penting untuk menunjang keberhasilan bagi seorang *project manager*. Evaluasi/*feedback* yang diberikan dapat membantu meningkatkan mutu dan keberhasilan dalam memimpin suatu proyek. Dalam hal ini, penulis sebagai asisten *project manager* diberikan beberapa *feedback* agar ke depannya dapat menjadi lebih baik lagi dalam menjalankan tugas yang diberikan oleh *project manager* sebagai berikut:

a. Banyak bertanya dengan anggota tim, jika tidak memahami kebutuhan yang ada. Karena penulis baru dalam peran asisten *project manager*. Banyak bertanya membantu penulis untuk dapat memahami kondisi dan situasi yang terjadi pada saat pembuatan proyek.
b. Tidak boleh bertanya melalui *private chat*, harus melalui grup *chat*, agar tim lain bisa membantu menjawab dan jawaban tersebut dapat membantu anggota tim lain yang memiliki permasalahan yang sama.
c. Dalam hal komunikasi juga diperlukan. Jangan sampai klien yang mengakhiri percakapan, harus penulis yang melakukan respon paling akhir. Meskipun hanya membalas "*Baik*,

terima kasih pak/bu" atau *"Baik, akan saya diskusikan dengan tim Javan terlebih dahulu".* Respon tersebut adalah bentuk dari akuntabilitas Javan terhadap klien.

d. Selalu mempertimbangkan ketika ada tambahan kebutuhan/fitur baru dari klien, karena semua yang ada dalam *sprint* sudah disusun sesuai dengan tenggat waktu berakhirnya sebuah proyek.
e. Selalu melakukan konfirmasi dengan melampirkan capaian-capaian yang sudah tim Javan kerjakan kepada klien. Konfirmasi kepada klien juga merupakan bentuk pertanggungjawaban tim Javan kepada klien.
f. Lebih teliti dalam pembuatan dokumen *timesheet* tim. *Timesheet* berguna untuk mencatat semua kegiatan yang dilakukan oleh tim Javan, sebagai hasil yang harus diserahkan ketika proyek telah selesai dikerjakan.

BAB V
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Task manajemen dapat melakukan pengorganisasian, penentuan prioritas, dan penjadwalan *task* dengan cara menghasilkan pencapaian dari berbagai tujuan secara efisien, efektif, dan produktif. Hingga mencapai tujuan yang diinginkan sesuai dengan kesepakatan pada awal pembuatan proyek. Improvisasi pada *task* bertujuan untuk membantu memaksimalkan penggunaan *tools* pada proyek. Improvisasi *task* dapat memuat informasi detail untuk menunjang pembuatan sebuah *task* yang sebelumnya memiliki kekurangan, sehingga informasi yang dilengkapi tersebut dapat bermanfaat bagi tim proyek dan mempercepat *closing* proyek yang dilakukan oleh *project manager* di sebuah proyek. Adapun kesimpulan yang didapatkan sebagai berikut:

a. Pertimbangkan kebutuhan apa saja yang diperlukan pada saat membangun sebuah sistem, sesuaikan dengan *task management tools* dengan kebutuhan tersebut, agar *tools* dapat bekerja dengan maksimal dan mempercepat *goals* yang akan dituju.
b. Taiga lebih cocok untuk proyek untuk tim *developer*, karena mudah digunakan dan mendukung *scrum* dan *Kanban*.
c. Jira lebih cocok digunakan untuk proyek yang berfokus untuk melakukan *tracking task/bug*, Jira juga dapat melakukan pengorganisasian ketika proyek berlangsung.
d. Berdasarkan analisis, improvisasi *task* dapat memberikan informasi yang berguna bagi tim pengembang, serta membantu dalam mempercepat pengerjaan *task* pada proyek yang sedang berjalan.
e. Improvisasi *task* sangat berguna bagi semua proyek, terutama di PT Javan Cipta Solusi sendiri, gunanya untuk memaksimalkan informasi pada *task* menjadi lengkap dan rinci.
f. Pada kenyataannya tidak selalu berjalan sesuai harapan, masih banyak kriteria yang dilanggar oleh tim *developer*, solusinya dengan membuat sebuah SOP (*Standard Operating Procedure*) *task* yang mengatur kriteria dalam pembuatan *task.*

Selain kesimpulan dalam improvisasi *task*, ada beberapa manfaat yang didapatkan ketika terlibat dalam proyek di Javan, sebagai berikut:

a. Dapat meningkatkan keahlian dan keterampilan, seperti berpikir kritis, mampu menghargai setiap pendapat yang disampaikan, lebih percaya diri dalam menyampaikan

pendapat, menambah ilmu dan wawasan di dalam proses magang berlangsung, serta manfaat lain mendapat relasi dari berbagai elemen di Javan maupun di luar Javan.

b. Pengalaman kerja bersama dengan tim, bertukar pikiran dan adanya saling tolong menolong jika belum paham dan mengerti. Ada kalimat yang melekat hingga saat ini, yaitu "*bertanyalah sebelum ditanya*" dan "*jika kalian tidak bertanya, kami anggap kalian sudah bisa*". Secara tidak langsung kalimat tersebut mendorong penulis untuk terus bertanya akan hal yang belum dipahami, dari situlah kepercayaan diri meningkat dikarenakan rasa keingintahuan yang besar.
c. Sering mengadakan evaluasi terhadap kinerja yang dilakukan, gunanya untuk meningkatkan performa kerja dari hari ke hari, minggu ke minggu sampai dengan penyelesaian proyek. Karena salah satu faktor keberhasilan sebuah proyek adalah terjalinya hubungan komunikasi yang baik antar tim.

5.2 Saran

Dalam proses pembuatan *task* perlu ketelitian dan pemahaman yang dalam akan kebutuhan dari suatu proyek. Dari pemahaman tersebut akan menghasilkan banyak perubahan dan penambahan dalam pembuatan *task*. Pembuatan sebuah standar dibutuhkan informasi yang lengkap dan rinci pada sebuah *task*. Dalam improvisasi *task* pada *software* manajemen proyek yang telah dilakukan dapat memberikan saran demi kesempurnaan dalam pembuatan *task* ke depannya.

Bagi mahasiswa yang sedang mengambil magang dalam penjaluran di semester terakhir, untuk ke depannya sebelum terjun dalam pelaksanaan magang, mahasiswa harus siap dalam memperbanyak bekal dan pengetahuan terkait bidang yang akan digeluti baik sebagai seorang *project manager*, *system analyst*, atau *quality assurance*. Mampu beradaptasi dengan situasi dan kondisi yang terjadi di dalam proyek, serta berperan aktif dalam pembangunan sebuah proyek. Selain itu harus mampu melakukan manajemen waktu terhadap diri sendiri supaya fokus tidak terpecah dalam hal melakukan pekerjaan magang dan tugas dari kampus. Terakhir tidak kalah penting, pekerjaan yang dilakukan harus didasari dengan ikhlas, tanggung jawab, dan giat untuk mencapai hasil yang maksimal.

DAFTAR PUSTAKA

Banerjee, J., & Buoti, C. (2012). General specifications of KPIs. *International Telecommunication Union.*

Bernhardt, B. R. (2014). Task Management Tools for the Structural Engineer. *Structures Congress, 2014*, (pp. 22-23).

Buana, S. Y. (2021). *Implementasi Scrum pada Pengembangan Modul Leadership Quality Feedback (LIQUID) (Studi Kasus: Pengembangan Aplikasi Komando).*

Chasanidou, D., Elvesæter, B., & Berre, A.-J. (2016). Enabling team collaboration with task management tools. *Proceedings of the 12th International Symposium on Open Collaboration*, (pp. 1-9).

Dimyati, H., & Nurjaman, K. (2014). *Manajemen Proyek.* Bandung: Pustaka Setia.

Hertel, G., Geister, S., & Konradt, U. (2005). Managing virtual teams: A review of current empirical research. (15), 69-95.

Javan Cipta Solusi. (2021). *SOP Taiga Javan.* Javan Cipta Solusi. Retrieved April 14, 2022

Jyothi, N. S., & Parkavi, A. (2016). A Study on Task Management System . *International Conference on Research Advances in Integrated Navigation Systems (RAINS - 2016)*, (pp. 1-6).

Kinasih, D. B. (2021). *Pengembangan Sistem Informasi Pengelolaan Kinerja Karyawan dengan Metode Scrum.*

Lam, H. E., & Maheshwari, P. (2001). Task and Team Management in the Distributed Software Project Management Tool. *In 25th Annual International Computer Software and Applications Conference. COMPSAC 2001* (pp. 401-408). IEEE.

Makhija, Y., & Goyal, A. (2014). Comparative Study of Project Tracking and Management Tools. *International Journal of Computer Science and Information Technologies. 5(4).* 5075-5080.

Manole, M., & Avramescu, M.-Ș. (2017). A Comparative Analysis of Agile Project Management Tools. *Academy of Economic Studies. Economy Informatics, 17(1),* pp. 25-31.

Özkan, D., & Mishra, A. (2019). Agile Project Management Tools: A Brief Comprative View. *CYBERNETICS AND INFORMATION TECHNOLOGIES, 19(4).*

Schwaber, K., & Sutherland, J. (2020). *Scrum Guides.*

Setyawati, E., Santamoko, R., Handoko, A. L., & Setiawan, P. (2021). *Manajemen Proyek Sistem Informasi.* Insan Cendekia Mandiri.

Shafiq, S., Mashkoor, A., Mayr-Dorn, C., & Egyed, A. (2021). TaskAllocator: A Recommendation Approach for Role-based Tasks Allocation in Agile Software Development. *2021 IEEE/ACM Joint 15th International Conference on Software and System Processes (ICSSP) and 16th ACM/IEEE International Conference on Global Software Engineering (ICGSE)* (pp. 39-49). IEEE.

Sitanggang, N., Simarmata, J., & Luthan, P. L. (2019). *Pengantar Konsep Manajemen Proyek untuk Teknik.* Yayasan Kita Menulis.

Suharno, H. R., Gunantara, N., & Sudarma, M. (2020). Analisis Penerapan Metode Scrum Pada Sistem Informasi Manajemen Proyek Dalam Industri & Organisasi Digital. *Majalah Ilmiah Teknologi Elektro, 19(2).*

Taheri, M., & Sadjadi, S. M. (2015). A Feature-Based Tool-Selection Classification for Agile Software Development. *SEKE*, (pp. 700-704).

Villavicencio, M., Narváez, E., Izquierdo, E., & Pincay, J. (2017). Learning Scrum by doing real-life projects. *2017 IEEE Global Engineering Education Conference (EDUCON)* (pp. 1450-1456). IEEE.

Westland, J. (2007). *The Project Management Life Cycle: A Complete Step-by-step Methodology*.

LAMPIRAN