PERAN *USER STORIES* DALAM REKAYASA KEBUTUHAN PENGEMBANGAN PERANGKAT LUNAK METODOLOGI *AGILE*

Disusun Oleh:

N a m a : Adi Setiawan

NIM : 18523237

PROGRAM STUDI INFORMATIKA – PROGRAM SARJANA

FAKULTAS TEKNOLOGI INDUSTRI

UNIVERSITAS ISLAM INDONESIA

2023

HALAMAN PENGESAHAN DOSEN PEMBIMBING

PERAN *USER STORIES* DALAM REKAYASA KEBUTUHANPENGEMBANGAN PERANGKAT LUNAK METODOLOGI *AGILE*

TUGAS AKHIR

Disusun Oleh:

N a m a : Adi Setiawan

NIM : 18523237

Yogyakarta, 9 Mei 2023

Pembimbing,

(Hanson Prihantoro Putro, S.T., M.T.)

HALAMAN PENGESAHAN DOSEN PENGUJI

PERAN *USER STORIES* DALAM REKAYASA KEBUTUHANPENGEMBANGAN PERANGKAT LUNAK METODOLOGI *AGILE*

TUGAS AKHIR

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, 9 Mei 2023

Tim Penguji

Hanson Prihantoro Putro, S.T., M.T. _____

Anggota 1

Kholid Haryono, S.T., M.Kom. _____

Anggota 2

Dr. Syarif Hidayat, S.Kom., M.I.T. _____

Mengetahui,

Ketua Program Studi Informatika – Program Sarjana

Fakultas Teknologi Industri

Universitas Islam Indonesia

(Dhomas Hatta Fudholi, ST., M.Eng., Ph.D.)

HALAMAN PERNYATAAN KEASLIAN TUGAS AKHIR

HALAMAN PERNYATAAN KEASLIAN TUGAS AKHIR

Yang bertanda tangan di bawah ini:

Nama	:	Adi Setiawan
NIM	:	18523237

Tugas akhir dengan judul:

PERAN *USER STORIES* DALAM REKAYASA KEBUTUHAN PENGEMBANGAN PERANGKAT LUNAK METODOLOGI AGILE

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, 16 Desember 2022

(Adi Setiawan)

HALAMAN PERSEMBAHAN

Skripsi ini saya persembahkan kepada orang tua saya yang telah memberikan doa, semangat, motivasi, nasihat yang terus diberikan untuk segera menyelesaikan skripsi dan membiayai kuliah saya. Tidak lupa skripsi ini juga saya persembahkan kepada dosen pembimbing saya, Bapak Hanson, yang sudah membimbing saya dalam mengerjakan tugas akhir dan juga kepada teman-teman saya yang sudah memberi dukungan, motivasi bahkan mengingatkan saya untuk tidak lupa mengerjakan skripsi.

HALAMAN MOTO

"Allah tidak membebani seseorang kecuali sesuai dengan kemampuannya"

QS. Al-Baqarah : 286

"Ketika *kepepet* ide-ide akan muncul entah itu benar

atau salah"(Adi Setiawan)

KATA PENGANTAR

Puji syukur ke hadirat Allah SWT yang telah memberikan limpahan rahmat, hidayah, serta kesehatan sehingga penulis dapat menyelesaikan skripsi dengan judul "Penerapan *User Stories* dalam Rekayasa Kebutuhan Perangkat Lunak Metodologi *Agile*", yang mana skripsi ini digunakan untuk salah satu syarat kelulusan dari Program Studi S1 Informatika, Fakultas Teknologi Industri, Universitas Islam Indonesia.

Dukungan dari berbagai pihak yang membuat penulisan laporan tugas akhir ini dapat diselesaikan. Terimakasih kepada:

1. Kedua orang tua yang sudah mendukung dan selalu mendoakan penulis sehingga dapat menyelesaikan laporan tugas akhir ini dengan lancer.
2. Bapak Dr. Raden Teduh Dirgahayu, S.T., M.Sc. selaku Ketua Jurusan Informatika.
3. Bapak Dhomas Hatta Fudholi, S.T., M.Eng., Ph.D. selaku Ketua Program Studi Informatika - Program Sarjana.
4. Bapak Hanson Prihantoro Putro, S.T., M.T. selaku Dosen Pembimbing Tugas Akhir penulis yang telah bersedia membimbing selama pengerjaan tugas akhir

Penulis menyadari bahwa masih terdapat kekurangan dalam laporan tugas akhir ini, sehingga penulis terbuka akan kritik dan saran. Diharapkan pada penelitian ini, mahasiswa lain dapat belajar sehingga dapat meningkatkan ilmu pengetahuan mereka.

Yogyakarta, 9 Mei 2023

(Adi Setiawan)

SARI

Di Program Studi Informatika Universitas Islam Indonesia, *website* penjaluran untuk mahasiswa tahun keempat sudah dikembangkan namun masih sebatas fitur pendaftaran yang tersedia. Masih terdapat beberapa kebutuhan proses bisnis jalur magang yang belum teridentifikasi di *website* tersebut. Pengembangan Sistem Informasi Jalur Magang diperlukan hingga semua proses bisnis jalur magang teridentifikasi secara lengkap. Pengembangan Sistem Informasi Jalur Magang ini dilakukan menggunakan metodologi *agile*. Penelitian ini berfokus pada proses rekayasa kebutuhan menggunakan alat bantu *user stories* untuk mengidentifikasi kebutuhan sistem secara lebih lengkap. Penelitian ini menghasilkan 17 *user stories* yang telah diidentifikasi, dianalisis, dan divalidasi di dalam seluruh tahap rekayasa kebutuhan perangkat lunak pada pengembangan proyek berbasis *agile*. Dalam analisis kebutuhan ini, dihasilkan juga *task*, desain antarmuka sistem, dan *database* yang diperoleh dari pengembangan user *sories*. Selanjutnya dalam tahap validasinya, telah dipastikan bahwa pengembangan *user stories* sesuai dengan proses bisnis dari pelaksanaan jalur magang yang terdapat di Program Studi Informatika – Program Sarjana, Fakultas Teknologi Industri, Universitas Islam Indonesia.

Kata kunci: *user stories*, rekayasa kebutuhan, sistem informasi manajemen, *agile*.

GLOSARIUM

Agile	salah satu metode pengembangan perangkat lunak
Kuisioner Likert	kuisioner untuk mengukur pendapat
Laravel	*framework* PHP yang dapat diakses secara bebas oleh pengguna
PHP	bahasa pemrograman
Prototype	sebuah desain dari perangkat lunak yang belum diimplementasikan ke koding
Stakeholder	semua yang terlibat dalam pengembangan perangkat lunak
System Usability Scale	pengujian perangkat lunak
User Stories	cerita pengguna dari sistem yang dikembangkan

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB I
PENDAHULUAN

1.1 Latar Belakang Masalah

Sistem informasi kini semakin berkembang seiring dengan berkembangnya zaman. Perkembangan sistem informasi kini sangat mempengaruhi segala bidang, khususnya pendidikan. Sistem informasi dibutuhkan manusia sebagai alat untuk menunjang segala aktivitas kerja supaya cepat dan tepat. Sistem informasi adalah proses mengolah, menganalisis, dan menampilkan data sampai data tersebut memiliki makna yang berguna untuk mendukung operasi manajemen dan fungsi pengambilan keputusan pada suatu organisasi (Ridwan Mohamad et al., 2019). Di dunia pendidikan, sistem informasi digunakan untuk mengelola, menyimpan, mengambil suatu informasi dan mengumpulkan data-data seperti data guru/dosen, mahasiswa/siswa pelajar, dan lain-lain serta pengambilan keputusan.

Pada pengembangan perangkat lunak, tahap awal yang dilakukan adalah analisis kebutuhan atau rekayasa kebutuhan. Terdapat masalah terkait rekayasa kebutuhan. Menurut Whitten dan Bentley (2007), fase yang kritis sebuah proyek adalah pada tahap analisis kebutuhan. Menurut Davis (1993) dan Leffingwell (1997), 40% - 60% kesalahan pada pengembangan perangkat lunak yaitu terletak pada tahap analisis kebutuhan. Berdasarkan data statistik dari survei Standish Group yang dikatakan Hull (2011), bahwa biasanya kesalahan pada pengembangan proyek bukan dikarenakan oleh hal teknis, tapi terletak di aspek non-teknis yaitu pada tahapan analisis kebutuhan (Adiguna et al., 2018).

Hasil dari penelitian menunjukkan bahwa praktik rekayasa kebutuhan di industri tidak dilakukan dengan baik. Menurut Kassab (2014), survei yang melibatkan 3000 lebih insinyur praktisi dengan respondennya 250 lebih, sebanyak 37% mengungkapkan bahwa praktik rekayasa kebutuhan yang terdapat di perusahaan tempat mereka bekerja belum memuaskan. Penelitian yang lain menurut Sikora (2012) yang melibatkan tujuh perusahaan rekayasa sistem yang terbenam di Eropa. Hasil dari penelitiannya adalah rekayasa kebutuhan yang ada belum cukup menangani kebutuhan dari suatu sistem terbenam secara kompleks (Laplante, 2016).

Salah satu alasan rekayasa kebutuhan kurang memadai adalah perusahaan yang melakukan praktik rekayasa kebutuhan mengalami kesulitan dalam memperluas rekayasa kebutuhan yang mencakup aktivitas dan struktur dari kebutuhan. Sikora (2012) menemukan alasan yang lain bahwa rekayasa kebutuhan yang dilakukan belum memenuhi kebutuhan tersebut. Alasan lain tersebut adalah, praktisi rekayasa kebutuhan menginginkan sebuah spesifikasi kebutuhan yang terintegrasi dengan keseluruhan arsitektur sistem, akan tetapi tidak

bergantung pada solusi spesifik terkait komponen atau fungsi yang telah ditentukan. Peserta survei juga mencatat bahwa metode yang tersedia tidak cukup untuk mendukung agar mencapai tujuan ini, terlihat rekayasa kebutuhan masih mempunyai tantangan dalam memenuhi kebutuhan para praktisi. Terdapat berbagai pendekatan rekayasa kebutuhan perangkat lunak. Salah satu pendekatan rekayasa kebutuhan perangkat lunak dapat menggunakan *user stories*.

User stories merupakan bagian dari metodologi *agile*. *User stories* adalah cara sederhana untuk mendapatkan permintaan pengguna terhadap projek yang dikembangkan. *User stories* digunakan untuk menjadi sebuah alat bantu untuk pengembangan perangkat lunak. *User stories* yang efektif berdampak besar terhadap pengembangan sistem informasi yang menggunakan metodologi *agile* (Certuche, 2016). *User stori*es digunakan karena mudah dipahami dan menggunakan komunikasi verbal (Cohn, 2009). Untuk penulisan *user stories* yang baik dapat mengikuti panduan. Panduan-panduan tersebut yaitu memulai dengan cerita tujuan, pembagian tugas, cerita yang ditulis harus tertutup, menerapkan batasan pada kartu, penyesuaian kartu cerita terhadap jangkauan, menahan supaya tidak menggunakan antarmuka pengguna, tidak semua hal merupakan suatu cerita, pengguna ikut berperan dalam penulisan cerita, menulis cerita untuk satu pengguna, menulis dalam bentuk kalimat aktif, penulisan cerita dilakukan oleh pelanggan, tidak memberikan nomor pada kartu cerita, dan tidak melupakan tujuan. Adapun kelebihan dari rekayasa kebutuhan yang menggunakan pendekatan *user stories*, yaitu *user stories* menggunakan komunikasi verbal, *user stories* dapat menjadi sebuah tolak ukur untuk perencanaan, dapat digunakan untuk pengembangan berulang, *user stories* secara diam-diam membangun pengetahuan, dapat mendukung desain partisipatif dan desain *oportunistik*, dan juga dapat mendukung penundaan.

Akan tetapi, terdapat masalah di dalam pengembangan perangkat lunak, yang pertama adalah masalah perencanaan perangkat lunak berdasarkan pada aktivitas bukan pada fitur (Mountain Goat Software, 2014). Pada masalah ini, terdapat dua masalah lagi yaitu pelanggan tidak mendapatkan nilai penyelesaian aktivitas, seharusnya perencanaan dilakukan pada tingkat fitur, bukan pada aktivitas. Masalah yang kedua adalah ketika jadwal perencanaan telah dibuat dan direvisi. Peninjauan jadwal yang menunjukkan aktivitas dilakukan untuk mencari aktivitas yang terlupakan dari fitur yang hilang. Perencanaan berdasarkan aktivitas juga mengakibatkan proyek melebihi dari jadwal yang sudah ditentukan. Pada saat terjadi kemunduran jadwal dari yang ditentukan, tidak jarang beberapa tim ingin menghemat waktu dengan cara mengurangi kualitas. Terdapat tim lain yang menerapkan perubahan yang dirancang agar produk dibatasi perubahannya, bahkan untuk perubahan yang penting.

Alasan keterlambatan jadwal pada perencanaan berdasarkan aktivitas dikarenakan yang pertama adalah aktivitas tidak diselesaikan lebih awal. Jika terdapat aktivitas yang menunjukkan bahwa aktivitas tersebut membutuhkan lima hari untuk diselesaikan, maka programmer yang diberi suatu tugas akan memastikan bahwa tugas yang dikerjakan tersebut benar-benar membutuhkan lima hari penuh. Yang dilakukan *developer* tersebut mungkin saja adalah melakukan penambahan fitur atau mungkin membagi waktu antara aktivitas dan penelitian teknologi baru yang mungkin berguna. *Developer* jarang menyelesaikan aktivitas yang diberikan tersebut lebih awal.

Kemudian jika terjadi keterlambatan, maka keterlambatan tersebut dilewatkan dalam jadwal. Hal-hal harus berjalan dengan lancar agar dapat memulai sesuatu lebih awal, sedangkan hal-hal yang salah dapat menyebabkan keterlambatan. Masalah bertumpuk apabila ditetapkan aktivitas-aktivitas yang selesainya jarang lebih awal, yang berarti aktivitas akan dimulai secara terlambat dan keterlambatan dalam memulai aktivitas tersebut akan dilewatkan dalam jadwal. Dikarenakan suatu penyelesaian aktivitas yang awal jarang terjadi, maka akan jarang ada suatu aktivitas yang akan dimulai lebih awal.

Terdapat masalah lainnya yaitu aktivitas masih bergantung dengan aktivitas yang lain. Aktivitas dikatakan mandiri jika aktivitas tidak mempengaruhi durasi dari aktivitas yang lain. Banyak aktivitas dari pengembangan perangkat lunak yang tidak saling mandiri, contohnya adalah ketika sedang menulis untuk bagian klien dari aplikasi dan membutuhkan waktu 50% lebih lama pada layar pertama, maka akan besar kemungkinan jika layar kedua dan seterusnya akan membutuhkan waktu yang lebih lama dari yang sudah direncanakan. Ini berarti, aktivitas yang lebih lama dari yang sudah direncanakan, kemungkinan besar aktivitas yang serupa akan memakan waktu yang lebih lama juga dari yang direncanakan.

Terjadi masalah ketika suatu aktivitas dilakukan secara *multitasking*. *Multitasking* ini menjadi masalah ketika pada aktivitas suatu proyek mulai ada yang selesainya terlambat. Hal ini membuat kritis pada aktivitas-aktivitas yang saling ketergantungan. Seorang pengembang menunggu pekerjaan yang sedang dikerjakan oleh pengembang lain hingga pekerjaan tersebut selesai dan diberikan oleh pengembang yang sedang menunggu pekerjaan tersebut. Terdapat dua alasan utama mengapa *multitasking* menjadi suatu masalah dalam pengembangan perangkat lunak secara tradisional. Yang pertama adalah tugas yang diberikan jauh sebelum tugas tersebut dimulai sehingga tugas yang sebelumnya tidak mungkin dialokasikan. Kemudian, pemberian tugas ke individu menjadi masalah daripada tugas tersebut diberikan kepada kelompok. Masalah yang kedua yaitu, fokus dari pendekatan ini adalah tingkat

penggunaan individu yang tinggi pada proyek, daripada meluangkan untuk menghadapi suatu perubahan pada proyek yang seringkali terjadi.

Masalah selanjutnya yaitu, fitur yang dikembangkan tidak berdasarkan pada prioritas. Pekerjaan yang telah dijabarkan pada rencana tidak diberi prioritas berdasarkan nilai bagi pengguna dan pelanggan merupakan salah satu alasan mengapa perancanaan secara tradisional tidak berhasil. Kebanyakan perencanaan tradisional dibuat berdasarkan asumsi bahwa semua aktivitas yang sudah teridentifikasi akan dikerjakan, ini berarti biasanya pekerjaan diprioritaskan dan diurutkan untuk kenyamanan pengembang. Pemikiran tradisional mengatakan, pelanggan tidak memiliki preferensi untuk urutan pekerjaan yang dikerjakan oleh pengembang, ini menyebabkan pengembang mengerjakan fitur-fitur yang terlihat acak urutannya bagi pelanggan. Pada akhir proyek, pengembang mengusahakan pekerjaan selesai sesuai dengan rencana tetapi beberapa fitur dihilangkan. Tak jarang, fitur-fitur yang dihilangkan memiliki nilai yang lebih besar untuk pelanggan.

Selanjutnya, tidak mampu mengakui ketidakpastian. Ketidakpastian tentang produk diabaikan dan menganggap bahwa analisis kebutuhan yang awal dibuat telah menghasilkan spesifikasi produk sempurna dan lengkap. Ini diasumsikan bahwa pengguna tidak akan mengubah pendapat, menyempurnakan opini, atau bahkan mempunyai kebutuhan baru saat perencanaan. Solusi dalam mengatasi ketidakpastian ini adalah dengan mengulang, jika terdapat pekerjaan yang terlewat dapat ditambahkan ke rencana. Kemudian, cara yang lain mengurangi ketidakpastian tentang produk tersebut adalah mengerjakan iterasi pendek dan menunjukkan suatu perangkat lunak yang dapat berfungsi kepada pengguna setiap beberapa minggunya.

Masalah yang terakhir adalah perkiraan menjadi suatu komitmen. Untuk perkiraan yang diberikan terdapat kemungkinan bahwa pekerjaan akan selesai sesuai dengan waktu yang diperkirakan. Menurut Armour (2002), perkiraan merupakan suatu probabilitas dan tidak dapat sebagai komitmen. Komitmen diberikan terhadap tanggal yang ditentukan. Pada umumnya, tanggal yang diminta atau diberitahukan kepada tim untuk dikomitmenkan adalah tanggal yang memiliki kemungkinan kurang dari 100%. Sebelum melakukan komitmen itu, evaluasi terdahadap faktor bisnis dan risiko perlu dilakukan oleh tim pengembang. Penting bagi tim pengembang untuk mengevaluasi faktor bisnis dan risiko supaya pada setiap perkiraan tidak dianggap sebagai komitmen yang melekat. Pada masalah-masalah yang disebutkan, terdapat salah satu jenis pengembangan perangkat lunak yang menggunakan metodologi *agile*.

Metodologi *agile* merupakan salah satu dari metodologi yang terdapat dalam pengembangan sistem informasi. Metodologi *agile* adalah metodologi perangkat lunak yang

dilakukan dengan mengutamakan kolaborasi tim. Metodologi ini banyak dipilih karena kesiapannya untuk menghadapi perubahan dalam pengembangan perangkat lunak. Perubahan dalam pengembangan diperlukan karena mempunyai tujuan agar perangkat lunak lebih menjadi lebih baik (Indra Kharisma Raharjana, 2017). *Agile* adalah metode yang secara strategis selalu diselaraskan dengan kasus bisnis yang diikuti (Goodpasture, 2016). Pengembangan perangkat lunak pada metodologi *agile* menggunakan *sprint*. Fokus dari *sprint* adalah untuk peningkatan yang berkelanjutan pada pengembangan perangkat lunak. *Sprint* memungkinkan tim pengembang untuk mengembangkan perangkat lunak secara singkat dan fokus terhadap tujuan yang ditetapkan. Hasil dari *sprint* adalah iterasi fungsional yang dapat digunakan untuk evaluasi dan penyempurnaan (Lutfiani et al., 2020). Pada setiap *sprint*, tim pengembang berkolaborasi, mengatasi suatu tantangan, dan aktif melakukan suatu perubahan yang sesuai dengan umpan balik yang diterima oleh tim pengembang. *Agile* banyak diterapkan pada pengembangan sistem informasi di berbagai instansi, salah satunya institusi di bidang pendidikan. Metodologi *agile* memiliki beberapa keuntungan, yaitu kepuasan klien meningkat, pengembangan perangkat lunak lebih cepat, kegagalan pada saat implementasi *software* dapat berkurang, dan kerugian yang terjadi saat kegagalan dari segi materi relative lebih kecil.

Pada beberapa universitas, magang menjadi salah satu kegiatan wajib yang dilakukan mahasiswa untuk menjadi syarat kelulusan. Kegiatan magang diadakan dengan tujuan mempersiapkan mahasiswa sebelum terjun ke dunia kerja setelah lulus. Kegiatan magang yang ada tersebut masih memiliki kendala, di antaranya pencarian informasi terkait perusahaan masih dilakukan secara manual. Hal ini menyebabkan mahasiswa kesulitan dalam mencari dan melamar magang di perusahaan.

Pada Program Studi Informatika – Program Sarjana, Fakultas Teknologi Industri, Universitas Islam Indonesia terdapat *website* pemilihan jalur kelulusan yang membantu mahasiswa tahun keempat untuk memilih jalur kelulusannya. Salah satu jalur kelulusan tersebut adalah Jalur Magang. Namun, *website* tersebut masih memiliki fitur yang terbatas. *Website* tersebut hanya memiliki fitur pendaftaran penjaluran beserta daftar nama dosen yang tersedia untuk membimbing jalur magang. Maka dari itu, dibutuhkan pengembangan sistem informasi jalur magang yang dapat mengidentifikasi seluruh proses bisnis penjaluran magang di Program Studi Informatika – Program Sarjana, Fakultas Teknologi Industri, Universitas Islam Indonesia.

Dalam penelitian ini, akan diterapkan rekayasa kebutuhan dengan pendekatan *user stories* dalam metodologi *agile* untuk mengembangkan Sistem Informasi Jalur Magang sehingga pelaksanaan jalur magang di Program Studi Informatika – Program Sarjana, Fakultas

Teknologi Industri, Universitas Islam Indonesia dapat berjalan dengan lancar. Pendekatan *user stories* dalam pengembangan perangkat lunak berbasis *website* ini diharapkan dapat membantu untuk memenuhi kebutuhan yang ada pada tahap pengembangan. *User stories* akan digunakan sebagai alat untuk menggambarkan kebutuhan dari pengguna dan pemangku kepentingan yang berkaitan dengan Sistem Informasi Jalur Magang. Dengan menggunakan metodologi *agile* dan pendekatan rekayasa kebutuhan menggunakan *user stories*, harapannya tim pengembang dapat memahami kebutuhan dari pengguna, tujuan sistem, dan fungsionalitas yang dibutuhkan.

Pendekatan *user stories* dapat memungkinkan pengembang fokus terhadap pengembangan fitur yang penting dan memberikan nilai tambah. *User stories* diuraikan secara spesifik dan jelas, mencakup aspek kegiatan, magang, pengawasan, pelaporan, dan evaluasi. Dengan demikian, rekayasa kebutuhan dengan menggunakan pendekatan *user stories* membantu memastikan jika perangkat lunak Sistem Informasi Jalur Magang yang dikembangkan dapat memenuhi kebutuhan dari pengguna dan memberikan suatu solusi dalam pengelolaan magang pada Program Studi Informatika – Program Sarjana, Fakultas Teknologi Industri, Universitas Islam Indonesia. Diharapkan, penerapan *user stories* ini dapat membantu memenuhi kebutuhan pada tahap pengembangan sistem.

1.2 Rumusan Masalah

Dari latar belakang yang telah dijelaskan, maka rumusan masalah yang diangkat adalah bagaimana *user stories* digunakan dalam rekayasa kebutuhan sebuah pengembangan perangkat lunak menggunakan metodologi *agile*, serta mengapa *user stories* dipilih sebagai rekayasa kebutuhan yang menggunakan metodologi *agile*.

1.3 Batasan Masalah

Penelitian ini memiliki batasan masalah yang dibuat supaya tidak menyimpang dari pokok permasalahan, yaitu:

a. Pengembangan sistem yang dilakukan fokus pada ruang lingkup Program Studi Informatika – Program Sarjana, Fakultas Teknologi Industri, Universitas Islam Indonesia.
b. Sistem yang dikembangkan adalah Sistem Informasi Jalur Magang.
c. Pengembangan sistem ini menggunakan metodologi *agile*.
d. Rekayasa kebutuhan dari sistem ini hanya menggunakan *user stories*.

1.4 Tujuan Penelitian

Tujuan dari penelitian ini adalah melakukan rekayasa kebutuhan dalam pengembangan Sistem Informasi Jalur Magang dalam metodologi *agile* menggunakan pendekatan rekayasa kebutuhan *user stories*. Dengan tercapainya dari tujuan tersebut, penelitian menghasilkan suatu pemahaman yang baik tentang bagaimana melakukan pengembangan Sistem Informasi Jalur Magang dengan menggunakan metodologi *agile* dan rekayasa kebutuhan pada sistem tersebut menggunakan pendekatan *user stories* yang dapat membantu untuk mengidentifikasi, menggambarkan, dan memprioritaskan kebutuhan dari pengguna dengan jelas.

Tujuan yang lain adalah memungkinkan untuk penambahan pengetahuan tentang keuntungan dan tantangan yang terjadi pada pengembangan perangkat lunak yang menggunakan metodologi *agile* dan fase rekayasa kebutuhannya menggunakan pendekatan *user stories*. Penelitian ini dapat memberikan suatu panduan dan pedoman bagi pengembang perangkat lunak yang ingin menggunakan metodologi *agile* dan rekayasa kebutuhan dengan menggunakan *user stories*.

1.5 Manfaat Penelitian

Diharapkan pada penelitian ini dapat memberikan manfaat:

a. Mengetahui bagaimana pengembangan perangkat lunak yang menggunakan metodologi *agile*, khususnya pada tahapan rekayasa kebutuhan perangkat lunak melalui pendekatan *user stories*.

Memahami tentang pengembangan perangkat lunak metodologi *agile* dan fase rekayasa kebutuhan menggunakan *user stories*. Menganalisis pengembangan perangkat lunak yang menggunakan metodologi *agile*, khususnya pada rekayasa kebutuhan perangkat lunak dengan *user stories*. Penelitian ini berguna untuk menemukan praktik yang terbaik dan menemukan tantangan-tantangan yang berkaitan dengan *user stories* dan pengembangan perangkat lunak yang menggunakan metodologi *agile*

b. Harapan dari penelitian ini yaitu menambah wawasan baru tentang *user stories* dan tantangan-tantangan yang terkait dalam praktik pengembangan perangkat lunak yang menggunakan metodologi *agile*.

Diharapkan pada penelitian ini dapat dipahami tentang bagaimana penggunaan *user stories* pada konteks pengembangan perangkat lunak yang menggunakan metodologi *agile*. Penelitian ini diharapkan dapat memberikan pengetahuan tentang tantangan-tantangan yang mungkin muncul pada implementasi *user stories* dalam pengembangan perangkat lunak metodologi *agile*. Selain itu, penelitian ini juga dapat memberikan suatu rekomendasi dan

solusi yang tepat untuk mengatasi tantangan-tantangan tersebut. Pemahaman dapat ditingkatkan tentang manfaat dan kegunaan dari *user stories* pada konteks metodologi *agile*.

c. Penggunaan Sistem Informasi Jalur Magang yang dapat membantu mencari perusahaan yang menyediakan lowongan magang dengan mudah dan cepat.

Pada penggunaan Sistem Informasi Jalur dapat untuk mencari perusahaan yang menyediakan lowongan magang secara signifikan, cepat, dan mudah. Sistem ini dapat digunakan untuk mengakses informasi tentang perusahaan yang menyediakan lowongan magang secara cepat dan efisien. Mahasiswa dapat menggunakan sistem ini untuk mencari perusahaan yang diminati dan diinginkan sesuai dengan kebutuhan mahasiswa.

d. Dengan Sistem Informasi Jalur Magang, mahasiswa yang mengambil jalur kelulusan magang dapat menemukan suatu peluang dengan minat dan kebutuhan.

Dengan menggunakan Sistem Informasi Jalur Magang, mahasiswa dapat mencari perusahaan yang menyediakan lowongan magang dengan suatu kriteria tertentu, seperti lokasi, dan nama perusahaan yang diinginkan, dan lain-lain. Mahasiswa dapat menggunakan suatu fitur pencarian yang dapat digunakan untuk mencari perusahaan yang diminati. Dengan dapat mengakses secara cepat dan mudah untuk mendapatkan suatu informasi tentang perusahaan yang menyediakan lowongan magang, mahasiswa dapat secara optimal untuk memilih perusahaan yang sesuai dengan minat dan kebutuhan mahasiswa tersebut.

Penelitian ini diharapkan dapat memberikan suatu pengetahuan baru tentang penggunaan metodologi *agile* dan rekayasa kebutuhan yang menggunakan *user stories* pada pengembangan perangkat lunak. Selain itu, diharapkan implementasi dari Sistem Informasi Jalur Magang dapat membantu menghubungkan mahasiswa dengan perusahaan magang dengan tepat, sehingga dapat mempermudah pencarian dan penempatan magang.

1.6 Metodologi Penelitian

Proses rekayasa kebutuhan memiliki langkah-langkah sebagai berikut (Bingamawa & Ahmad, 2016):

a. Studi Kelayakan (*Feasibility Study*)

Pada proses pertama, dilakukan studi kelayakan. Studi kelayakan berguna untuk menilai perangkat lunak yang dibuat apakah berguna di dalam bisnis atau tidak. Kemudian hasilnya berupa *feasibility report* (laporan kelayakan). Laporan kelayakan berisi suatu informasi apakah perangkat lunak memiliki nilai tambah dan berpotensi untuk dijadikan bisnis atau tidak. Hasil dari laporan studi kelayakan digunakan untuk penggambaran apakah perangkat lunak yang

dikembangkan memiliki suatu nilai tambah pada bisnis dan dapat mencapai tujuan dari bisnis atau tidak.

b. Elisitasi dan Analisis Kebutuhan (*Requirements Elicitation and Analysis*)

Proses yang kedua ini dilakukan untuk menemukan berbagai kebutuhan dari sistem. Pada proses elisitasi dan analisis kebutuhan yang dilakukan adalah wawancara, observasi dan-lain-lain untuk menemukan suatu kebutuhan dari perangkat lunak yang akan dikembangkan. Keluaran dari proses ini adalah *system models* (model sistem). Model sistem merupakan representasi visual dari kebutuhan perangkat lunak yang telah teridentifikasi dan telah dianalisis.

c. Spesifikasi Kebutuhan (*Requirements Specification*)

Pada proses ketiga, spesifikasi kebutuhan dilakukan perubahan standardisasi kebutuhan sistem. Hasil dalam proses ini adalah *user and system requirements* (kebutuhan pengguna dan sistem).

d. Validasi Kebutuhan (*Requirements Validation*)

Proses yang keempat adalah validasi kebutuhan. Pada validasi kebutuhan dilakukan pemeriksaan terhadap kebutuhan untuk memastikan apakah perangkat lunak tersebut sesuai dengan keinginan pengguna atau tidak. Validasi kebutuhan melibatkan verifikasi fungsi yang telah sesuai dengan kebutuhan dari pengguna dan fitur yang tidak diperlukan dalam perangkat lunak. Tidak ada suatu konflik dan pemastian kebutuhan dapat terbukti pada proses validasi kebutuhan.

1.7 Sistematika Laporan

BAB 1 PENDAHALUAN

Bab ini berisi tentang latar belakang masalah dari penelitian ini dilakukan, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, metodologi penelitian, dan struktur laporan.

BAB 2 DASAR TEORI

Bab ini berisi tentang teori-teori dari metodologi *agile*, rekayasa kebutuhan, *user stories*, sistem informasi magang, dan kajian pustaka.

BAB 3 ANALISIS KEBUTUHAN

Bab ini berisi tentang studi kelayakan, elisitasi kebutuhan, dan spesifikasi kebutuhan.

BAB 4 VALIDASI KEBUTUHAN

Bab ini berisi tentang pengecekan konsistensi, pengecekan kebenaran dan kelengkapan, verifikasi *user stories*, dan validasi *user stories* melalui *prototype*.

BAB 5 KESIMPULAN DAN SARAN

Bab ini berisi tentang kesimpulan yang dari penelitian ini dan saran untuk penelitian ke depan.

BAB II
DASAR TEORI

2.1 Metodologi *Agile*

Metodologi *agile* merupakan metodologi untuk mengembangkan perangkat lunak yang dapat beradaptasi dengan perubahan. Perubahan yang dilakukan dimaksudkan untuk lebih meningkatkan kualitas dari perangkat lunak agar menjadi lebih baik (Indra Kharisma Raharjana, 2017). Pengembangan di dalam metodologi *agile* dilakukan dengan singkat atau bisa disebut dengan "*sprint*". *Sprint* ini berfokus pada peningkatan yang berkelanjutan di dalam pengembangan perangkat lunak (Lutfiani et al., 2020). *Agile* adalah cara untuk menyelaraskan secara strategis pada kasus bisnis yang diikuti (Goodpasture, 2016).

Metodologi *agile* menjadi bagian iteratif yang fokusnya pada perubahan, kolaborasi, dan pengiriman produk yang lebih awal dengan kualitas yang tetap dijaga. Terdapat praktik dan prinsip yang harus dianut oleh para pengembang yang menggunakan metodolgi *agile*. Sifat dari metodologi ini adalah adaptif dibandingkan prediktif. Metodologi *agile* berbeda dari pendekatan yang lebih menekankan perancangan perangkat lunak secara rinci dalam jangka waktu lama, dan perubahan spesfikasi yang mengakibatkan masalah pada perangkat lunak. Berubahnya kebutuhan yang mengakibatkan terhambatnya pendekatan desain yang berfokus pada dokumen awal dapat direspons dengan baik dengan menggunakan metodologi *agile*.

Metodologi *agile* berguna pada rekayasa kebutuhan. Terdapat elemen metologi *agile* yang diterapkan pada rekayasa kebutuhan terkhusus pada pertimbangan manusia. Biasanya, metodologi *agile* digambarkan ramping apabila diterapkan pada sistem nonperangkat lunak. Pengembangan yang menggunakan metodologi *agile* bergantung pada beberapa rangkaian *prototype* cepat yang tidak dibuang dan menggunakan pendekatan yang tidak praktis pada sistem yang berbasis perangkat keras. Namun, insinyur nonperangkat keras masih bisa menggunakan metodologi *agile* dan mendapat keuntungan karena metode ini semakin banyak digunakan.

Agile berorientasi pada orang daripada proses, yang mengakibatkan pengembangan perangkat lunak yang menggunakan metodologi *agile* menjadi menyenangkan. Namun, terdapat tantangan pada pengembangan perangkat lunak yang menggunakan metodologi *agile* (Laplante, 2016). Tantangan tersebut, yaitu:

a. Komunikasi dengan semua stakeholder menjadi perantara agar tidak terjadi kesenjangan.
b. Partisipasi pelanggan ditingkatkan.
c. Ukuran dan kerumitan pada dokumentasi dikurangi.

d. Mengendalikan *scope creep* atau pertumbuhan tidak terkendali yang terjadi pada ruang lingkup proyek.
e. Tercapainya validasi kebutuhan.

Agile dikembangkan karena pada metodologi tradisional masih terdapat banyak permasalahan yang membuat proses pengembangan perangkat lunak tidak berhasil dengan baik sesuai dengan tuntutan dari *user*. Keunggulan yang dimiliki *agile* dibandingkan dengan metodologi yang lain adalah kepuasan dari pengguna meningkat, *review* dari perangkat lunak yang dikembangkan bisa dilakukan lebih awal, kemudian dapat mengurangi resiko kegagalan dalam pengembangan perangkat lunak dari segi teknis maupun non-teknis, dan jika terjadi kegagalan, kerugian material dan immaterial menjadi tidak terlalu besar (Ependi, 2017).

Salah satu jenis di dalam metodologi *agile,* yaitu *Simple Agile Methodology* atau bisa disebut dengan *SAM*. *SAM* merupakan jenis metodologi *agile* yang terinspirasi dari jenis metodologi *Scrum* dan *Extreme Programming*. Metodologi *Scrum* memungkinkan tim diatur oleh diri sendiri dengan menggunakan komunikasi verbal di antara anggota tim dan para *stakeholder*. Prinsip dari *Scrum* adalah metodologi pengembangan perangkat lunak yang bergerak sesuai dengan rencana seperti *waterfall* dan pengembangan berulang yang fokus pada suatu proses. Pada metodologi *Scrum,* lebih ditekankan pentingnya perangkat yang berfungsi sejak awal. Fokus dari *Scrum* adalah kemampuan tim yang dimaksimalkan agar tantangan yang muncul dapat diselesaikan dengan cepat. Sedangkan *Extreme Programming* atau XP yaitu salah satu jenis dari metodologi *agile* yang paling sering digunakan. Secara sederhana, XP adalah jenis pengembangan yang ditargetkan untuk tim pengembang kecil dan artefak yang dibutuhkan sedikit. Siklus dari pengembangan yang menggunakan XP adalah pendekatan berulang.

SAM menggunakan pendekatan empiris yang menerima bukti bahwa masalah itu sepenuhnya tidak dapat didefinisikan pada awal. Sebagai gantinya, *SAM* berfokus pada cara memaksimalkan kemampuan dari tim agar dapat tersampainya produk kepada *user* dengan cepat, perubahan persyaratan yang muncul, dan adaptasi perkembangan teknologi yang terjadi pada kondisi pasar (uBugtrack, 2023). *SAM* memungkinkan untuk melakukan pemadatan terhadap proses pengembangan ke dalam waktu yang singkat, tidak hanya mengurangi pemborosan proses sebesar 50% lebih, akan tetapi pemasaran dapat dilakukan dengan waktu yang lebih cepat. Pada umumnya, perusahaan perangkat lunak membutuhkan 3-4 minggu untuk menggambarkan dan pencarian solusi, akan tetapi perusahaan pengembangan perangkat lunak yang menggunakan pendekatan *SAM* ini dapat melakukan hanya dalam tiga hari saja. Proses yang dilakukan adalah adalah memahami kebutuhan, dan dari kebutuhan itu perusahaan

tersebut menggunakan *SAM* untuk mengembangkan *MVP* (*Minimum Viable Product*) yang kemudian diluncurkan sebelum 12 minggu. *SAM* dapat memenuhi sebagian besar kebutuhan dengan mudah, meskipun kebutuhan dari perangkat lunak tersebut kompleks. Pendekatan *Simple Agile Methodology* ini sangat kolaboratif dikarenakan klien ikut berpartisipasi dalam proses pengembangan perangkat lunak.

Di dalam metode *agile*, interaksi dan personel lebih penting dibandingkan proses dan alat. Perangkat lunak yang berfungsi juga akan lebih penting dibandingkan dokumentasi yang lengkap. Antara kolaborasi dengan klien dan negosiasi kontrak, kolaborasi terhadap klien menjadi hal yang lebih penting. Kecepatan mengikuti perubahan juga lebih penting daripada harus selalu sesuai dengan rencana. Oleh karena itu interaksi antara pengembang dan klien harus sering dilakukan untuk mengatasi kecepatan perubahan sehingga menghasilkan perangkat lunak yang berfungsi dengan baik (OKI, 2022).

Salah satu ciri dari metodologi *agile* adalah komunikasi menjadi hal yang penting antara klien dan pengembang. Ciri yang lain yaitu klien masuk dalam tim pengembang *software*. Prioritas utamanya adalah kepuasan klien dengan hasil produk yang lebih awal dan berkelanjutan. Selain itu, terdapat 12 prinsip yang dapat mendukung ciri-ciri dari metodologi *agile* yaitu (Messer Hugo, 2021):

a. Kepuasan klien adalah prioritas utama.
b. Segala kebutuhan yang terjadi saat pengembangan perangkat lunak diterima.
c. Hasil produk dikembangkan hanya dalam beberapa minggu atau beberapa bulan.
d. Kerja sama antara klien dan pengembang sangat diperlukan dalam pengembangan.
e. Orang-orang dengan motivasi tinggi diperlukan dalam pengembangan.
f. Komunikasi efektif dan efisien diperlukan dalam pengembangan dengan cara tatap muka langsung antara klien dan pengembang.
g. Ukuran utama untuk kemajuan proyek adalah perangkat yang dapat berfungsi dengan baik.
h. Perlunya dukungan dari sponsor, pengembang, dan pengguna.
i. Teknis dan desain yang baik dapat meningkatkan sifat dari kelincahan.
j. Kesederhanaan merupakan hal yang penting.
k. Segala bentuk arsitektur, kebutuhan, dan desain yang bagus bergantung pada individu yang ada di dalam tim pengembang.
l. Evaluasi secara individu adalah cara efektif yang dilakukan di dalam pengembangan metodologi *agile*.

2.2 Rekayasa Kebutuhan

Rekayasa kebutuhan yaitu cabang dari teknik yang bersangkutan dengan tujuan dunia nyata, fungsi, dan kendala pada suatu sistem. Ini berhubungan dengan spesifikasi yang tepat dari suatu aktifitas dan perubahan sistem dari waktu ke waktu (Laplante, 2016).

Rekayasa kebutuhan perangkat lunak adalah upaya untuk menentukan komponen suatu sistem (Lewenusa, 2017). Dari proses tersebut dihasilkan desain yang dijadikan dasar rekayasa kebutuhan perangkat lunak. Yang menjadi tantangan rekayasa kebutuhan yaitu pemahaman apa sebenarnya yang menjadi kebutuhan dari perangkat lunak. Kebutuhan dapat bertumpu pada kebutuhan yang abstrak bertingkat tinggi, sketsa, hingga spesifikasi yang formal. Contoh kebutuhan dihasilkan berbagai macam bentuk, hal ini dikarenakan kebutuhan dan kemampuan *stakeholder* berbeda- beda. Langkah-langkah dalam rekayasa kebutuhan adalah sebagai berikut (Bingamawa & Ahmad, 2016):

a. Studi kelayakan (*feasibility study*)

Studi kelayakan merupakan proses evaluasi penentuan apakah suatu investasi pada proyek perlu dilakukan. Ketika studi kelayakan dilakukan lebih fokus pada penilaian proyek, disebut studi kelayakan proyek. Jika fokus studi kelayakan pada penilaian sebuah usaha disebut studi kelayakan bisnis. Studi kelayakan proyek ataupun studi kelayakan bisnis bertujuan untuk mengevaluasi beberapa aspek seperti keuangan, teknis, operasional, dan strages suatu inisiatif. Dari studi kelayakan dapat diketahui apakah proyek dapat memiliki suatu potensi tercapainya keberhasilan dan menghasilkan keuntungan yang diharapkan (Harahap, 2018).

Pertama kali yang dilakukan pada rekayasa kebutuhan perangkat lunak adalah studi kelayakan. Studi kelayakan memiliki tujuan untuk menilai apakah dari perangkat lunak yang dikembangkan akan memiliki suatu manfaat dalam bisnis atau tidak. Hasil dari studi kelayakan yang dilakukan menghasilkan sebuah *feasibility report*. Laporan ini berisi informasi yang berkaitan dengan kecukupan dan kelayakan dari suatu perangkat lunak yang dikembangkan. Dari *feasibility report* yang dihasilkan, perangkat lunak yang dikembangkan apakah memiliki suatu potensi nilai tambah dan dapat menghasilkan dalam lingkup bisnis.

b. Elisitasi dan analisis kebutuhan (*Requirements Elicitation and Analysis*)

Elisitasi dan analisis kebutuhan adalah proses yang penting dalam pengembangan perangkat lunak yang memiliki tujuan untuk mengumpulkan, membagi, dan memprioritaskan suatu kebutuhan yang menjadi dasar dari perangkat lunak yang dikembangkan (Adikara et al., 2018). Dari proses yang dilakukan, berbagai kebutuhan dari sistem dan pengguna ditemukan. Yang dilakukan pada elisitasi dapat berupa wawancara, observasi, dan lainnya. Sementara

analisis kebutuhan melibatkan suatu pemahaman tentang kebutuhan yang telah teridentifikasi. Hasil dari proses elisitasi dan analisis kebutuhan ini yaitu *system models*. *System models* merupakan penggambaran visual dari suatu kebutuhan perangkat lunak yang telah diidentifikasi dan dianalisis.

c. Spesifikasi kebutuhan (*Requirements Specification*)

Proses ini melakukan perubahan untuk standarisasi kebutuhan. Hasil yang dilakukan pada spesifikasi kebutuhan adalah *user and system requirements*. Terdapat beberapa macam jenis dari kebutuhan adalah kebutuhan fungsional, kebutuhan nonfungsional, dan kebutuhan domain.

Kebutuhan fungsional adalah penggambaran layanan yang harus tersedia pada sistem dan reaksi dari sistem yang telah menerima *input*. Secara eksplisit, kebutuhan fungsional juga menyatakan tentang aktivitas-aktivitas yang tidak dilakukan oleh sistem. Kebutuhan dapat bersifat umum dan tingkat tinggi, atau bersifat terperinci, mencakup masukan, keluaran, pengecualian, dan lain-lain.

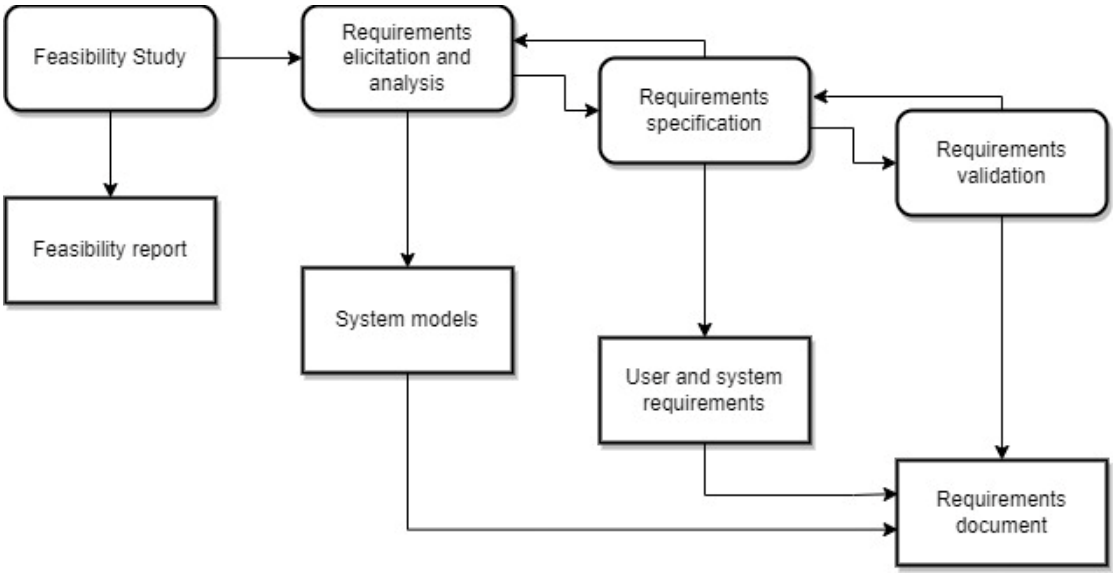
Kebutuhan nonfungsional lebih penting untuk membedakan antara produk-produk yang bersaing dibandingkan kebutuhan fungsional. Namun nyatanya, kebutuhan nonfungsional sering diabaikan daripada kebutuhan fungsional. Kebutuhan nonfungsional terabaikan dikarenakan sifat unik dari kebutuhan nonfungsional itu sendiri yang mengakibatkan adanya tantangan dalam memutuskan penanganannya pada tahap awal pengembangan. Sifat dari kebutuhan nonfungsional adalah subyektif, relatif, dan seringkali menyebar di antara beberapa modul pada saat dipetakan dari domain kebutuhan ke ruang solusi. Dan sifat dari kebutuhan nonfungsional juga saling berinteraksi.

Asal dari kebutuhan domain adalah dari domain aplikasi yang digunakan. Jenis kebutuhan domain dapat mencakup kebutuhan fungsional baru atau batasan yang pada kebutuhan fungsional yang sudah ada, atau dapat juga menentukan cara komputasi tertentu dilakukan.

d. Validasi kebutuhan (*requirements validation*)

Yang keempat adalah proses validasi kebutuhan sistem. Dalam proses ini, dilakukan pemeriksaan terhadap kebutuhan untuk memastikan perangkat lunak sesuai dengan keinginan pengguna. Validasi kebutuhan melibatkan verifikasi apakah sistem memenuhi semua fungsi yang paling baik untuk mendukung kebutuhan dari pelangan, dan bahwa sistem tidak menyedian berbagai fungsi yang tidak diperlukan oleh pelanggan. Tak hanya itu, validasi harus memastikan bahwa tidak ada konflik antar kebutuhan dan pemenuhan kebutuhan dapat dibuktikan.

Dokumen yang dihasilkan dari langkah elisitasi dan analisis kebutuhan yang berupa system models serta langkah spesifikasi kebutuhan yang menghasilkan user and system requirements kemudian digabungkan. Gabungan dari dokumen system models dan user and system requirements berupa requirements document. Langkah dari proses rekayasa kebutuhan dapat dilihat pada Gambar 2.1.



Gambar 2.1 Proses Rekayasa Kebutuhan Perangkat Lunak (Bingamawa & Ahmad, 2016)

Terdapat tantangan-tantangan yang dihadapi pada metodologi *agile* terkhususnya pada fase rekayasa kebutuhan (Laplante, 2016). Tantangan-tantangan tersebut adalah *agile* seringkali tidak terdapat penanganan yang memadai terhadap kebutuhan nonfungsional, dikarenakan kebutuhan nonfungsional tidak selalu jelas saat hanya fokus pada fungsionalitas kebutuhan melalui *user stories*. Cara untuk mengatasi masalah ini adalah untuk melengkapi *user stories* dengan teknik penemukan kebutuhan nonfungsional yang tepat.

Tantangan selanjutnya adalah meskipun interaksi yang sangat sering dilakukan dengan pelanggan di dalam pengembangan yang menggunakan metodologi *agile*, interaksi yang dilakukan tersebut sebagian besar dilakukan melaui *prototyping*. Cara lain untuk mendapat pemahaman yang mendalam tentang kebutuhan yang lebih halus dan memahami kebutuhan para *stakeholders*, seperti menggunakan berbagai teknik wawancara.

Validasi yang dilakukan di dalam metodologi *agile* sangatlah kuat. Validasi ini dilakukan melaui pengujian dan *prototyping*, namun verifikasi tidaklah sekuat itu. Menurut Williams (2004), cara untuk memperkuat rekayasa kebutuhan di dalam pengembangan perangkat lunak yang menggunakan metodologi *agile* adalah dengan metode formal.

Dokumen-dokumen yang sering diabaikan dalam pengembangan perangkat lunak yang menggunakan metodologi *agile*. Pengetahuan domain, arsitektur, dan desain yang muncul tidak tertangkap baik oleh praktik kebutuhan *agile* tradisonal. Dokumen yang diperlukan di dalam kebutuhan *agile* perlu diberi catatan, disertai pengetahuan domain yang relevan. Pemberian catatan yang dilengkapi dengan arsitektur dan desain sistem yang sesuai pada dokumen arsitektur dan desain *agile*. Menurut Williams (2004), penambahan praktik manajemen konfigurasi standar supaya manajemen kebutuhan kuat.

Terdapat beberapa tantangan yang lain terkait pengembangan perangkat lunak yang metodologi *agile* terkhususnya pada fase rekayasa kebutuhan. Menurut Inayat (2015), tantangan-tantangan tersebut adalah:

a. Kebutuhan nonfungsional diabaikan.
b. *Requirements traceability* atau jejak kebutuhan kurang.
c. Kesalahan pada prioritas kebutuhan.
d. Minimnya dokumentasi kebutuhan.
e. Permasalahan kontrak.
f. Ketersediaan pelanggan.
g. Persetujuan dari pelanggan.

Meskipun penggunaan metodologi *agile* dalam pengembangan perangkat lunak khususnya pada fase rekayasa kebutuhan masih terdapat tantangan, terdapat juga keuntungan. Oleh karena itu, metodologi *agile* layak dipertimbangkan untuk digunakan dalam pengembangan perangkat lunak. Bahkan ketika metodologi tidak cocok untuk proyek pengembangan perangkat lunak tertentu atau proyek yang bukan pengembangan perangkat lunak, prakti-praktik yang dibahas pada metodologi *agile* masih bisa diterapkan. Sebagai contoh praktik, pelanggan yang selalu ada di tempat, lebih fokus produk daripada dokumentasi, dan pengembangan awal kasus uji.

2.3 *User Stories*

User stories merupakan sebuah cara yang sederhana untuk mendapatkan permintaan pengguna dari projek yang akan dikembangkan. *User stories* dapat berisi masalah atau dapat mengubah desain produk (Lau, 2010). *User stories* adalah unit yang paling dasar dalam pada sebagian besar pengembangan perangkat lunak yang menggunakan metodologi *agile*. Setiap dari cerita mencerminkan fitur sistem yang diinginkan oleh pelanggan. Tim rekayasa perangkat lunak memperoleh kebutuhan formal, *use case*, dan artefak lainnya dari *user stories* sesuai dengan kebutuhan. Pada awal, biasanya *user stories* dikumpulkan pada pertemuan di luar lokasi. Pendekatan berorientasi pada tujuan atau pendekatan interaktif yang dilakukan dapat

menghasilkan suatu cerita. Pengembangan *user stories* merupakan proses yang iteratif dan interaktif. Pengembang mengelola ukuran dari cerita agar konsisten, misalnya jika ukuran ceritanya terlalu besar, pengembang dapat mengecilkan ukuran cerita tersebut dan jika ukuran cerita terlalu kecil, maka pengembang akan menggabungkan cerita tersebut.

Pelanggan harus memahami *user stories* dan setiap dari *user stories* harus memberikan nilai tambah. *User stories* ditulis oleh pelanggan, bukan dari pengembang yang melakukan, akan tetapi, cerita yang ditulis harus cukup kecil agar beberapa cerita dapat diselesaikan dalam setiap iterasi. Sebisa mungkin, cerita yang ditulis harus independen, yang berarti satu cerita tidak boleh merujuk kepada cerita yang lainnya. Yang terakhir, cerita harus dapat diuji seperti kebutuhan yang lainnya. Jika cerita tersebut tidak dapat diuji, maka cerita tersebut bukanlah suatu kebutuhan.

Terdapat perbedaan yang signifikan antara *user stories* dan *use case*. *User stories* asalnya dari sudut pandang pelanggan yang sederhana dan menghindari rincian implementasi. Sedangkan *use case* lebih kompleks dan memungkinkan untuk mencakup rincian implementasi. Pada biasanya, pelanggan tidak menulis *use case*, apabila pelanggan menulis *use case*, perlu diwaspadai, karena itu artinya pelanggan terlibat dalam rekayasa perangkat lunak. Beberapa *use case* dapat setara dengan satu *user stories*, sedangkan satu *user* stories dapat setara dengan satu atau lebih *use case* (Laplante, 2016).

2.3.1 Panduan Menulis *User Stories*

Terdapat beberapa panduan untuk menulis cerita agar cerita menjadi lebih baik. Beberapa panduannya yaitu, memulai dengan cerita tujuan, kemudian membagi tugas, yang selanjutnya yaitu cerita yang ditulis tertutup, batasan kartu diterapkan, menyesuaikan ukuran dari cerita dengan jangkauan, menahan untuk menggunakan antarmuka pengguna sejauh mungkin, tidak semua hal merupakan cerita, menyertakan peran dari pengguna di dalam cerita, menulis cerita untuk satu pengguna, menulis *user stories* dalam bentuk kalimat aktif, pelanggan yang menulis cerita, tidak memberikan nomor pada kartu cerita, dan tidak melupakan tujuannya. Berikut adalah penjelasan dari panduan-panduan untuk menulis *user stories* menjadi lebih baik:

a. Memulai dengan cerita tujuan

Pada proyek besar, terutama yang melibatkan banyak peran dari pengguna, sering menemukan kesulitan dalam menentukan permulaan dalam mengidentifikasi cerita. Cara yang efektif untuk mengatasi masalah tersebut adalah dengan melakukan pertimbangan setiap peran dari pengguna dan mengidentifikasi tujuan yang dimiliki oleh pengguna saat berinteraksi dengan perangkat lunak yang dikembangkan. Tujuan yang dimiliki oleh cerita ini adalah cerita

tingkat tinggi yang bisa digunakan sebagai dasar dari hasil cerita tambahan yang sesuai dengan kebutuhan.

b. Pembagian tugas

Pada cerita yang besar, biasanya terdapat banyak cara untuk membagi cerita yang besar tersebut menjadi bagian yang kecil. Kebiasaan umum yang dilakukan oleh sebagaian besar pengembang yang adalah memecah cerita berdasarkan aspek teknis. Sementara terdapat pendekatan yang jauh lebih baik yaitu menulis cerita pengganti hingga sedemikian rupa sehingga masing-masing dari cerita dapat memberikan tingkat fungsionalitas yang lengkap. Setiap dari cerita herus memiliki paling tidak sedikit dari setiap lapisan. Cerita yang mewakili potongan tugas yang lengkap lebih disukai daripada yang tidak memiliki potongan cerita, alasannya adalah, yang pertama, pengujian setiap dari lapisan arsitektur, penemuan risiko masalah pada salah satu lapisan pada menit terakhir dapat dikurangi. Yang kedua adalah, pada dasarnya, aplikasi yang dirilis berdasarkan dengan fungsionalitas parsial selama fungsionalitas disertakan dalam perilisan meliputi seluruh sistem, meskipun tidak ideal.

c. Cerita yang ditulis tertutup

Konsep penutupan tugas dalam kumpulan teknik yang disusun merupakan sebuah konsep yang dikenalkan oleh Soren Lauesen (2002). Konsep ini dapat diterapkan pada *user stories*. Cerita yang tertutup merupakan cerita yang diakhiri dengan pencapaian dari tujuan yang signifikan dan dapat memungkinkan pengguna untuk merasa telah mencapai sesuatu. Setiap dari cerita yang tertutup merupakan bagian cerita asli yang belum selesai. Setelah salah satu tertutup ini selesai, maka kemungkinan besar pengguna akan merasa telah mencapai sesuatu. Akan tetapi, keinginan untuk menulis cerita yang tertutup harus seimbang dengan kebutuhan yang bersaing. Perlu diingat juga, bahwa cerita harus cukup kecil agar dapat diestimasi dan dijadwalkan dengan nyaman pada satu iterasi. Namun, cerita yang ditulis juga harus cukup besar agar tidak menggali lebih rinci terkait cerita tersebut lebih awal.

d. Menerapkan batasan pada kartu

Panduan selanjutnya dalam menulis *user stories* yang bagus adalah menerapkan batasan pada kartu cerita. Batasan pada kartu cerita ini dikenalkan oleh Newkirk dan Martin (2001). Pada praktik ini dilakukan pemberian tanda kartu cerita dengan “batasan” untuk cerita yang harus dipatuhi daripada cerita yang diimplementasikan secara langsung. Meskipun kartu batasan tidak diperkirakan dan tidak dijadwalkan ke dalam iterasi seperti kartu-kartu biasa, tetap saja kartu batasan memiliki manfaat. Minimal, kartu batasan ditempelkan di dinding untuk pengingat. Agar lebih baik, suatu tes penerimaan ditulis untuk memastikan batasan supaya tidak dilanggar.

e. Penyesuaian kartu ukuran cerita dengan jangkauan

Kemudian, panduan selanjutnya adalah menyesuaikan ukuran dari cerita dengan jangkauan. Dalam menulis cerita, perhatian lebih banyak pada hal-hal yang terjadi pada waktu dekat dibandingkan dengan hal-hal yang terjadi pada masa mendatang dapat dilakukan dengan menulis cerita dengan berbagai tingkat berdasarka dari jangkauan implentasi cerita tersebut. Cerita untuk iterasi mendatang akan disusun dengan ukuran yang sesuai untuk direncanakan dalam iterasi tersebut, sedangkan cerita yang jauh ke masa mendatang akan memliki skala yang berukuran lebih besar dan kurang presisi. Cerita akan diperluas seiring dengan pembahasan mengenai hal rinci yang berkaitan dengan cerita tersebut. Dalam menulis cerita, dapat memanfaatkan flesibilitas cerita agar dapat berguna di berbagai tingkat.

f. Menahan agar tidak menggunakan antarmuka pengguna

Pencampuran kebutuhan dengan spesifikasi solusi adalah salah satu permasalahan yang sering terjadi pada pendekatan kebutuhan perangkat lunak. Pada saat menyatakan suatu kebutuhan, secara eksplisit dan implisit solusi juga diikut sertakan. Seringkali hal ini terjadi terkait aspek antarmuka pengguna. Oleh karena itu, sebaiknya menghindari antarmuka pengguna dalam cerita-cerita. Akhirnya, hal yang tidak dapat dihindari bahwa detail dari antarmuka pengguna masuk dalam cerita. Ini terjadi ketika perangkat lunak sudah semakin lengkap dan cerita berubah dari fungsionalitas yang baru menjadi modifikasi atau bisa juga perluasan dari fungsionalitas yang telah ada.

g. Tidak semua hal merupakan cerita

Meskipun *user stories* merupakan format yang fleksibel dan efektif dalam menggambarkan fungsionalitas sebagian besar dari banyak sistem, format tersebut tidak selalu sesuai untuk semua hal. Pendokumentasian dan menyetujui antarmuka antara sistem penting, terutama jika salah satu dikembangkan oleh pihak eksternal. Gunakan format yang berbeda jika menemukan bahwa terdapat beberapa aspek dari system yang lebih baik jika disampaikan dengan format yang berbeda.

h. Menyertakan peran dari pengguna di dalam cerita

Apabila tim proyek telah mengindetifikasi peran dari pengguna, maka sebaiknya tim pengembang memanfaatkan dalam menulis berbagai cerita. Fokus utama dalam pikiran pengembang adalah pengguna. Sebagai pengganti memikirkan pengguna abstrak, tak berwajah, dan dapat digantikan, pengembang mulai memikirkan pengguna nyata, dengan kebutuhan yang perlu dipenuhi oleh perangkat lunak.

i. Menulis cerita untuk satu pengguna

Cerita akan lebih mudah dibaca saat ditulis untuk satu pengguna. Mungkin, beberapa cerita tidak terpengaruh oleh apakah cerita tersebut ditulis untuk satu pengguna atau banyak. Namun, pada beberapa cerita, terdapat perbedaan yang signifikan. Masalah ini akan jelas pada saat mempertimbangkan cerita dengan satu pengguna dalam pikiran.

j. Menulis *user stories* dalam bentuk kalimat aktif

Penulisan *user stories* menggunakan kalimat aktif. Hal ini dikarenakan *user stories* yang ditulis menggunakan kalimat aktif mudah dimengerti dan mudah dibaca. Dengan menggunakan kalimat aktif, *user stories* dapat dengan mudah dimengerti dan mudah dibaca oleh tim pengembang, pemangku kepentingan, dan semua pihak yang terlibat dalam pengembangan perangkat lunak.

k. Penulisan cerita dilakukan oleh pelanggan

Idealnya, yang menulis cerita-cerita pada *user stories* adalah pelanggan. Pada beberapa proyek, pengembang ikut membantu dengan cara menulis langsung pada *workshop* penulisan cerita awal pengembang memberikan saran cerita yang baru pada pelanggan. Akan tetapi, pelanggan mempunyai tanggung jawab yang utama dalam penulisan cerita dan tidak dapat dialihkan kepada para pengembang. Tanggung jawab utama lainnya, pelanggan melakukan prioritisasi cerita-cerita yang akan dimasukkan ke dalam setiap iterasi, oleh karena itu pengembang harus memahami setiap dari cerita yang ditulis. Salah satu cara yang terbaik untuk pelanggan memahami cerita tersebut adalah dengan melakukan penulisan cerita sendiri.

l. Tidak memberikan nomor pada kartu cerita

Ketika pertama kali menggunakan kartu cerita, kebanyakan ingin memberikan nomor pada setiap dari cerita. Alasan umumnya yaitu memberikan tingkat keterlacakan pada cerita atau agar mempermudah dalam melacak kartu individu. Pemberian nomor kartu cerita hanya menambah beban yang tidak diperlukan pada proses dan akan membawa ke sebuah diskusi abstrak tentang fitur yang diperlukan. Jika dirasa perlu untuk memberikan nomor pada kartu cerita, tambahkan judul singkat pada setiap kartu cerita dan judul tersebut digunakan untuk menjelaskan isi dari cerita yang lain.

m. Tidak melupakan tujuan

Mengingat tujuan utama dari kartu cerita ialah untuk pengingat pembahasan dari fitur tersebut. Pengingat yang dibuat seharusnya ringkas. Penambahan detail yang diperlukan sebagai pengingat di mana melanjutkan sebuah percakapan, namun tidak menggantikan sebuah percakapan tersebut dengan menambahkan detail yang lebih lanjut pada kartu cerita.

2.3.2 Komponen *User Stories*

Tujuan *user stories* menurut Cohen (2004) adalah untuk mengetahui apa yang dibutuhkan dari sisi pengguna. Tiga aspek utama *user stories* yaitu: 1) Penulisan penjelasan cerita yang berfungsi sebagai perencanaan dan pengingat; 2) Penjelasan yang mendeskripsikan detail-detail dari cerita; 3) Pengujian dokumentasi secara detail untuk memutuskan apakah cerita sudah lengkap atau belum (Pratidana, 2017). Menurut Leffingwell (2011) terdapat tiga bagian dalam menulis *user stories*, bagian-bagian tersebut meliputi:

a. *Role*, menunjukkan orang yang memperoleh nilai dari sistem yang melakukan aktivitas tersebut.
b. *Activity*, menunjukkan aktivitas yang dilakukan oleh sistem.
c. *Business value*, menunjukkan nilai yang berada di dalam bisnis dan alasannya.

Sebagai contoh "Sebagai mahasiswa <*role*>, saya ingin mencari perusahaan yang menawarkan lowongan magang <*activity*>, sehingga saya dapat mengirim lamaran magang di tempat perusahaan tersebut <*business value*>."

2.3.3 Kelebihan *User Stories*

User stories memiliki kelebihan. Beberapa kelebihan *user stories* yaitu *user stories* mudah dipahami, *user stories* mengutamakan bentuk komunikasi verbal, *user stories* menjadi tolak ukur untuk perencanaan, dapat berguna untuk pengembangan berulang, membangun pengetahuan secara diam-diam, mendukung desain partisipatif dan *oportunistik*, dan mendukung penundaan secara detail (Cohn, 2009). Berikut adalah penjelasan masing-masing dari kelebihan *user stories*:

a. *User stories* mudah dipahami oleh pengguna dan pengembang.

Pada dokumen skenario yang bergaya IEEE 830 bahwa spesifikasi kebutuhan dapat dipahami dengan mudah oleh pengembang dan juga pengguna. Disebutkan bahwa pada dokumen yang bergaya IEEE 830, banyak jargon teknis yang dapat dibaca oleh pengguna dan banyak terdapat jargon khusus untuk dibaca oleh pengembang. Skenario secara detail yang ditempatkan pada kehidupan nyata membuat pandangan terhadap skenario menjadi blur atau kabur yang mengakibatkan skenario sulit untuk dipahami. Pelaku bisnis dan pengembang dapat memahami dengan mudah cerita singkat yang tetuju pada nilai pengguna. Penelitian pada akhir tahun 1970 menemukan bahwa orang dapat dengan mudah mengingat sesuatu yang disusun menjadi sebuah cerita. Cerita dapat ditulis singkat daripada spesifikasi kebutuhan yang sederhana atau bahkan use case, karena ditulis sebagai cerita, daya ingat lebih besar.

b. *User stories* menggunakan komunikasi verbal.

Dalam komunikasi verbal yang dilakukan antara pengembang, pelanggan, dan pengguna, terdapat peluang untuk umpan balik yang menjadi arah pembelajaran bersama. Tujuan dari cerita pengguna untuk menampung kalimat singkat yang akan mengingatkan pengembang dan pelanggan agar diadakan percakapan pada masa yang akan datang. Cerita yang cukup dimengerti menjadi tujuan yang lebih penting dibandingkan dengan persyaratan yang sempurna.

c. *User stories* menjadi tolak ukur dari perencanaan.

Sebagian pengembang meminta pengguna untuk memprioritaskan kebutuhan dengan gaya IEEE 830. Hasilnya adalah 90% kebutuhan wajib, 5% sangat diinginkan tapi ditunda dengan waktu yang lebih singkat, dan 5% sisanya ditunda dengan waktu yang lebih lama, dikarenakan sulit untuk melakukan prioritisasi dan bekerja dengan banyak kalimat bahkan ribuan kalimat yang dimulai dengan "Sistem harus…". Ketika ribuan bahkan puluhan ribu pernyataan yang terdapat di dalam spesifikasi kebutuhan dupertimbangkan, akan kesulitan pada saat melakukan prioritisasi. Pada masalah use case dan skenario desain interaksi berbeda. Untuk memprioritaskan use case dan skenario desain interaksi itu mudah, akan tetapi terkadang hasilnya adalah tidak semua yang berada di prioritas utama lebih penting dibandingkan yeng berada di prioritas kedua. Di lain sisi, cerita dapat digunakan dengan mudah untuk rencana perilisan, pemrograman, dan pengujian dikarenakan cerita mempunyai ukuran yang dapat dikelola.

d. Sebelum memulai menulis kode program tidak perlu untuk menuliskan semua cerita.

Dari beberapa cerita yang ditulis dapat dijadikan kode untuk kemudian diuji cerita tersebut, lalu diulang seperlunya. *User stories* dapat dijadikan pengembangan berulang dikarenakan kemudahannya dari mengulang cerita. Pada dokumen IEEE 830 memiliki arti tersirat bahwa jika tidak ada pernyataan yang tertulis "Sistem harus…", maka pernyataan tersebut tidak ada di sistem. Hal ini dapat membuat ketidakpastian dikarenakan suatu pernyataan bisa saja hilang atau bahkan persyaratan tersebut belum ditulis. Skenario memiliki kekuatan pada detail. Kegunaan skenario akan hilang jika skenario dimulai tanpa adanya detail yang kemudian secara progresif detail ditambahkan. *Use case* dapat berbentuk teks bebas, namun sebagian besar dari orgranisasi menggunakan template standar. Organisasi memberi pesan untuk menggunakan template standar jika ingin membuat *use case*. Yang menjadi masalahnya adalah, semua orang harus mengisi formulir sesuai dengan tenplate standar yang sudah dibuat. Pada praktiknya, organisasi yang menggunakan template standar *use case* hanya

sedikit. *User stories* sebagai pelengkap bekerja dengan baik karena belum ada yang mengusulkan untuk dibuatkan template dari setiap cerita.

e. *User stories* membangun pengetahuan secara diam diam.

Hal ini terjadi dikarenakan komunikasi tatap muka lebih ditekankan. Jika pengembang dan pengguna sering melakukan komunikasi secara langsung, maka akan banyak pengetahuan yang nantinya akan diperoleh tim.

f. *User stories* mendorong desain partisipatif.

Banyak proyek pengembangan perangkat lunak yang gagal dikarenakan kurangnya partisipasi dari penggunanya untuk mendesain perangkat lunak melalui cerita. Menurut Kuhn dan Muller (1993) serta Shuler dan Namioka (1993), desain partisipatif adalah pengguna dari sistem juga menjadi tim yang mendesain aktivitas perangkat lunak tersebut. Pengguna masuk ke dalam tim dikarenakan kebutuhan dan teknik desain yang digunakan terikat dengan pengguna. Desain partisipatif melibatkan pengguna untuk membantu pengembangan *prototype* dari awal. Sedangkan desain empiris adalah pengembang perangkat lunak baru yang mempelajari calon pengguna dan situasi dalam perangkat lunak untuk membuat sebuah keputusan. Desain empiris bergantung pada wawancara dan observasi, akan tetapi pengguna tidak menjadi bagian dari pengembangan perangkat lunak sejak awal, terutama pada desain perangkat lunak. *User stories* dan skenario mendorong pengguna untuk pembuatan desain perangkat lunak. Pengguna mempelajari cara karakterisasi pada kebutuhan yang dibutuhkan pada perangkat lunak dalam cerita yang dapat berguna bagi pengembang. Pengembang dapat lebih sering melibatkan pengguna. Siklus baik ini menguntungkan baik pengembang maupun pengguna perangkat lunak.

g. *User stories* juga mendorong desain *oportunistik*.

Desain *oportunistik* yaitu pendekatan yang digunakan untuk mengembangkan perangkat lunak baru dengan menggunakan kembali dan menggabungkan dari komponen yang tidak dirancang untuk digunakan (Mäkitalo et al., 2020). Pengembang menggunakan pendeketaan *oportunistik* yang dapat memungkinkan pengembang untuk bergerak lebih bebas dalam mempertimbangkan suatu kebutuhan dalam menemukan dan mendiskusikan penggunaan skenario atau dapat juga merancang berbagai abstraksi. Perangkat lunak yang dikembangkan harus secara *oportunistik*, hal ini dikarenakan tidak ada proses berjalan yang benar-benar linier dari kebutuhan tingkat tinggi ke kode. *User stories* dapat dengan mudah untuk memungkinkan tim beralih antara tingkat pemikiran tinggi dan rendah dan dapat juga beralih kebutuhan.

2.3.4 Kekurangan *User Stories*

User stories tidak hanya memiliki kelebihan, namun *user stories* memiliki kekurangan. Kekurangan dari *user stories* adalah (Cohn, 2009):

a. Proyek besar menghasilkan cerita banyak

Dengan proyek yang besar, banyak cerita yang dihasilkan, sulit untuk memahami hubungan antar cerita. Namun, dengan penambahan peran dan penjagaan tingkat cerita dari sedang ke tinggi, masalah ini dapat dipecahkan. Penetapan peran yang jelas dapat membantu memastikan bahwa tingkat dari kompleksitas cerita disesuaikan dengan kemampuan dari tim pengembang, sehingga tim pengembang siap mengembangkan cerita yang rumit. Pendekatan ini, dapat mengatasi masalah pemahaman hubungan antar cerita, dan manajamen proyek dapat lebih efisien.

b. Penambahan dokumen jika terjadi ketertelusuran.

Pada proses pengembangan perangkat lunak, penting untuk menjamin jika ketertelusuran persyaratan terjadi. Masalah ini dapat diatasi dengan penambahan dokumentasi. Salah satu cara yang dilakukan adalah dengan menyertakan dokumen yang berisi catatan pada setiap iterasi pengembangan. Dokumen ini mencakup suatu informasi tentang persyaratan yang diusulkan, perubahan pada saat pengembangan, dan pemahaman yang jelas tentang hubungan antar persyaratan. Pengembang dapat melacak dan memahami perubahan yang terjadi pada prooses pengembangan perangkat lunak karena ada dokumen ini. Dokumentasi tambahan juga memiliki peran dalam membantu dalam komunikasi dan kolaborasi antara pemangku kepentingan dan tim pengembang. Penambahan dokumen perlu dilakukan pada setiap iterasi agar jika terjadi suatu ketertelusuran dokumen persyaratan dapat diatasi pada setiap pengembangan perangkat lunak.

c. Untuk tim pengembang yang besar, *user stories* tidak menambah pengetahuan

Meskipun *user stories* dapat meningkatkan pengetahuan secara diam-diam, namun hal ini tidak berlaku untuk bagi tim pengembangan yang besar. Pada tim pengembangan yang besar, komunikasi menjadi hal yang lebih penting, terutama karena informasi sering kali hanya ditulis dan orang yang memiliki informasi tersebut jumlahnya terbatas. Hal ini menyebabkan kesulitan untuk memahami konteks dan hubungan antar cerita. Cara mengatasi ini adalah memperkuat komunikasi di dalam tim dengan mendorong diskusi secara langsung, bertukar informasi langsung, dan kolaborasi aktif. Selain itu, dokumentasi yang terstruktur dan jelas dapat membantu juga dalam menjaga suatu kejelasan dan pemahaman pada cerita-cerita. Dengan demikian, penting bagi tim pengembang untuk menjaga komunikasi kepada para

pemangku kepentingan dan memastikan jika informasi yang diperlukan tersedia dan dapat diakses seluruh anggota tim.

Kunci dari pengembangan perangkat lunak yang menggunakan metodologi *Agile* mengutamakan orang dan *user stories* menempatkan pengguna terakhir untuk menjadi pusat dari percakapan. Pengguna menjadi fokus utama pada setiap percakapan yang dilakukan dan pengambilan keputusan. *User stories* dirancang agar dapat memberikan suatu pemahaman yang jelas tentang kebutuhan dari pengguna. Pembuatan *user stories* menggunakan bahasa non-teknis sehingga para pengembang mendapatkan konteks dari kebutuhan pengguna. Setelah membaca *user stories*, mengetahui mengapa membangun perangkat lunak tersebut, mengetahui apa yang dibangun, dan apa nilai yang dibuat dari perangkat lunak tersebut. Adanya *user stories*, pengembang dapat memiliki suatu panduan jelas untuk mengembangkan fitur yang bermanfaat dan relevan bagi pengguna. Tim pengembang dapat lebih fokus terhadap pengalaman dan kebutuhan dari pengguna, serta memastikan nilai yang dihasilkan dari perangkat lunak sesuai dengan keinginan dari pengguna.

Pada metodologi *agile*, pengguna memiliki peran untuk aktif dalam pengembangan tidak hanya menunggu perangkat lunak selesai dikembangkan. Dengan menempatkan pengguna sebagai pusat dari percakapan, maka tim pengembang memastikan bahwa perangkat lunak yang dikembangkan dapat memberikan suatu nilai dan kepuasan yang maksimal untuk pengguna.

2.4 Sistem Informasi Magang

Sistem adalah dasar yang penting dari semua kegiatan, dan berbagai bidang membutuhkan sistem. Kegiatan atau pekerjaan akan tidak terkendali jika tidak ada sistem. sistem akan berfungsi dengan baik jika semua karakteristik yang berada di dalam sistem bekerja bergabung guna mencapai suatu tujuan yang telah ditetapkan pada awalnya.

Pada era revolusi industry 4.0 menuju masyarakat 5.0 pada konteks informasi, suatu sistem menjadi hal yang penting dan dapat diakses secara optimal. Informasi adalah suatu sumber pengetahuan yang dapat memungkinkan manusia melihat dunia melalui sudut pandang yang lebih luas dan dapat memberikan manfaat bagi kehidupan. Berbagai bidang pekerajaan memanfaatkan kemajuan teknologi dalam otomatisasi. Sistem informasi dapat mudah diakses melalui teknologi, meskipun secara fisik terlihat rumit dan jauh. Adanya sistem informasi ini sangat membantu mempermudah untuk kehidupan manusia (Ridwan Mohamad et al., 2019).

Sistem informasi adalah sistem yang berisi subsistem yang saling berkaitan satu dengan lainnya dan membentuk suatu informasi yang berguna bagi pengguna (RIZALDI, 2021).

Adanya sistem informasi dapat meningkatkan produktivitas, kinerja, efektivitas, serta efisiensi (Baharuddin, 2021). Sistem informasi memiliki peran penting pada organisasi zaman sekarang. Sistem informasi membantu perusahaan ataupun organisasi untuk mendapatkan suatu keunggulan kompetitif (Islahuddin et al., 2020).

Magang adalah salah satu pelatihan kemampuan, keterampilan dan pemahaman bagi mahasiswa sebelum bekerja di lapangan kerja. Kegiatan magang dapat berupa hal kecil hingga hal kompleks yang mana kegiatan ini memperbanyak pengalaman mahasiswa sebelum benar-benar terjun ke dalam dunia kerja. Adapun manfaat dari magang yang lain yaitu, wawasan mahasiswa/siswa menjadi bertambah dan menjalin hubungan antara perusahaan penyedia magang dan instansi pendidikan (Islahuddin et al., 2020).

Sistem informasi magang dimanfaatkan untuk mengelola data permagangan dari bentuk data fisik menjadi bentuk data digital. Dalam sistem informasi magang ini terdapat beberapa pihak yang menjadi pendukung kegiatan magang. Pihak-pihak tersebut antara lain seperti penyelenggara magang, peserta magang, dan perusahaan atau tempat yang menyediakan lowongan magang. Proses bisnis dari sistem informasi magang, yaitu pendataan peserta yang melakukan magang, pendataan tempat penyedia lowongan magang, dan termasuk kegiatan peserta sehari-hari yang dilakukan pada saat magang dapat dilakukan disatu sistem (Arif Setia Sandi A. et al., 2020).

Dari penjelasan di atas, dapat diambil sebuah definisi sistem informasi magang yaitu suatu kegiatan belajar yang terjadi pada dunia nyata dan didukung penggunaan teknologi dari komputer (Amelia et al., 2021). Sistem informasi magang mengintegrasikan berbagai komponen dan membentuk suatu sistem yang dapat mengumpulkan, menyimpan, dan menampilkan suatu informasi yang berkaitan dengan kegiatan magang. Melalui sistem informasi ini, peserta magang dengan mudah mengakses suatu informasi tentang jadwal, tugas, evaluasi magang, dan catatan tentang magang. Sistem informasi magang dapat menjadi fasilitas untuk peserta magang, pembimbing magang, dan pihak lain yang terkait untuk melakukan interaksi. Adanya sistem informasi magang ini, kegiatan magang menjadi lebih efisien, terstruktur, dan transparan dalam menyediakan suatu informasi yang diperlukan peserta magang atau semua pihak yang terkait kegiatan magang.

Pada program studi Informatika Universitas Islam Indonesia, magang menjadi salah satu penjaluran tugas akhir. Mahasiswa yang mengambil jurusan magang harus mengikuti kegiatan di perusahaan yang mereka ajukan. Mahasiswa menjalan kegiatan magang selama dua semester atau paling tidak minimal selama enam bulan. Akhir dari program magang adalah mahasiswa menulis laporan kegiatan magang dan skripsi. Program studi Informatika Universitas Islam

Indonesia terdapat website Sekawan. Sekawan digunakan para mahasiswa untuk mendaftar program penjaluran. Pada website Sekawan belum terpenuhi kebutuhan untuk para mahasiswa mencari informasi tentang kegiatan magang. Diharapkan dengan Sistem Informasi Jalur Magang ini, mahasiswa Program Studi Informatika – Program Sarjana, Fakultas Teknologi Industri, Universitas Islam Indonesia dapat mencari lowongan magang dengan mudah.

2.5 Pengujian *System Usability Scale* (SUS)

John Broke memperkenalkan sebuah konsep pengujian yang bernama *System Usability Scale*. *System Usability Scale* adalah skala *usability* yang teruji dengan biaya murah yang digunakan untuk menguji usability dari sistem secara universal. SUS yang didasarkan pada kuisioner Likert menggunakan pertanyaan yang telah distandarisasi. Pertanyaan yang distandarisasi tersebut dapat memberikan nilai rata-rata dari usability sistem dan juga dapat memberikan kepuasan pengguna. Skala dari kepuasan pengguna yaitu dari 0-100 (Ramadhan, 2019). SUS merupakan metode pengujian yang biayanya murah tetapi efektif untuk menguji produk seperti website, handphone, dan lain-lain. Hasil dari SUS berskala 0-100 yang membuat mudah dipahami. *System usability scale* merupakan metode pengujian yang menghasilkan hasil cukup berdasarkan dari pertimbangan jumlah sampel, biaya, dan waktu yang sedikit. Aplikasi layak atau tidak, dipertimbangkan dari hasil SUS yang dikonversi sebuah nilai.

2.6 Kajian Pustaka

Terdapat beberapa makalah yang membahas tentang sistem informasi magang, magang, metodologi *agile*, dan *user stories*. Makalah tersebut disajikan dalam Tabel 2.1.

Tabel 2.1 Kajian Pustaka

No.	Makalah	Masalah	Metode	Hasil
1	(Islahuddin et al., 2020)	Proses pengajuan magang sampai penerbitan surat masih manual, dan tidak ada rekapitulasi kegiatan siswa magang.	*Waterfall*	Mengembangkan sistem informasi magang.

2	(Yulianto, Hery Dwi, 2018)	Pelaporan absensi dan kegiatan harian mahasiswa magang masih konvensional.	Metode deskriptif	Merancang sistem untuk mengubah pelaporan absensi dan kegiatan harian mahasiswa magang yang masih dilakukan secara konvensional.
3	(Anggraini, 2021)	Catatan keuangan masjid baik pemasukan maupun pengeluaran masih dilakukan secara manual.	Metode pengumpulan data *User stories*	Mengembangkan sistem informasi administrasi untuk masjid.
4	(RIZALDI, 2021)	Penyampaian informasi akademik membutuhkan waktu lama dan kertas yang digunakan terlalu banyak sehingga memerlukan banyak tempat.	*Waterfall*	Memudahkan mahasiswa mengakses informasi akademuk kapan dan di mana sajasecara *mobile*, mempersingkat waktu mahasiswa dalam melihat informasi akademik, dan memudahkan pihak Universitas menyebarluaskan informasi akademik kepada mahasiswa.
5	(Baharuddin, 2021)	Memerlukan sarana pendukung untuk meningkatkan kinerja, efisiensi, efektivitas, dan produktivitas pengelolaan magang.	Pengumpulan data	Pengembangan sistem informasi manajemen pelaksanaan magang yang dapat memudahkan menampilkan informasi dan administrasi pada kegiatan magang.

6	(Ismail et al., 2018)	Perlunya visi dan misi menjadi seorang guru.	Pengumpulan data	Meningkatkan keterampilan, keahlian, dan kompetensi pedagogic pada mahasiswa calon guru.
7	(Amelia et al., 2021)	Belum memanfaatkan sistem informasi digital untuk melakukan pengolahan data dan informasi yang berkaigtan dengan kegiatan magang.	*Waterfall*	Mengembangkan *web* sistem informasi magang yang menggunakan metode *waterfall.*
8	(Priskila, 2018)	Masih menggunakan *Microsoft Excel* untuk mencatat persediaan barang dan laporan.	*Agile software development User stories*	Merancang sistem informasi persediaan barang menggunakan analisis kebutuhan berdasarkan penggunaan kebutuhan sistem.
9	(Lestari & Novita, 2019)	Pendaftaran magang lama, yang mana mengharuskan mahasiswa datang ke perusahaan untuk menanyakan tentang ketersediaan lowongan magang.	*Waterfall*	Mengembangkan sistem informasi magang
10	(Komalasari et al., 2022)	Pendaftaran magang masih dilakukan secara manual dengan menggunakan Microsoft Word.	*Waterfall*	Mengembangkan sistem sistem informasi pendaftaran magang

Pada peneltian (Islahuddin et al., 2020) membahas tentang sistem informasi magang. Terdapat masalah yang menyebabkan waktu yang terbuang menjadi lebih banyak. Permasalahannya yaitu proses pengajuan kegiatan magang hingga penerbitan surat balasan masih dilakukan secara manual. Calon peserta magang harus mendatangi kantor BKN yang mengakibatkan waktu menjadi lama. Kemudian, permasalahan yang lain yaitu pemberitahuan status keterima atau tertolaknya masih berupa surat yang harus diambil di kantor BKN. Kemudian, terdapat kendala dari pihak BKN yaitu tidak ada rekapitulasi dari kegiatan sehari-hari peserta magang yang mengakibatkan pihak dari BKN tidak mengetahui apa yang dilakukan peserta magang dalam kesehariannya. Tidak hanya itu, masalah yang lain yang dimiliki oleh pihak BKN adalah tidak ada rekapitulasi nilai dikarenakan pihak BKN tidak memiliki format dari penilaian peserta magang. Jika pihak BKN memiliki sebuah rekapitulasi dari penilaian peserta magang, rekapitulasi peneliaian tersebut dapat menjadi acuan untuk penerimaan peserta magang. Selanjutnya, terdapat masalah lainnya yaitu, terpisahnya proses rekapitulasi dari masing-masing unit yang mengakibatkan data-data yang tercatat sulit untuk diakses. Permasalahan-permasalahan yang ada, didapatkan dari wawancara yang narasumbernya adalah seorang Kepala Sub Bagian Tata Usaha Biro Kepegawaian. Dari permasalahan-permasalahan tersebut, kemudian dikembangkan sistem informasi magang agar memudahkan proses pendaftaran magang, dari peserta mendaftar magang sampai peserta tersebut selesai magang dan memudahkan pihak BKN dalam mengontrol kegiatan sehari-hari peserta magang. Pengembangan sistem informasi magang ini menggunakan metode *waterfall*. Setelah pengembangan sistem informasi magang selesai, maka selanjutnya sistem tersebut diuji. Pengujian yang dilakukan menggunakan pendekatan *User Acceptance Test*.

Penelitian (Yulianto, Hery Dwi, 2018) membahas tentang permasalahan magang, yakni absen dan laporan harian mahasiswa yang melakukan magang masih menggunakan cara konvensional dan belum terdokumentasi secara baik. Hal ini menyebabkan, pengawas magang kesulitan dalam mengawasi kehadiran peserta magang dan kegiatan sehari-hari peserta magang. Dari permasalahan yang didapatkan, solusinya adalah membangun sistem agar absen dan laporan harian mahasiswa magang tidak lagi dilakukan secara konvensional dan dapat diawasi oleh pengawas. Metode yang digunakan yaitu metode deskriptif. Hasil dari penelitian ini adalah sistem informasi monitoring magang yang dapat digunakan untuk membantu proses dari kegiatan magang agar kualitas kegiatan magang yang dilakukan oleh peserta magang meningkat.

Selanjutnya, terdapat penelitian (Anggraini, 2021) yang membahas tentang *user stories*. Terdapat masalah pada catatan keuangan masjid. Catatan keuangan masjid yang berupa

pengeluaran maupun pemasukan masih dicatat manual. Pencatatan dilakukan menggunakan buku besar sehingga sulit untuk mengetahui pengeluaran dan pemasukan. Terdapat masalah yang lain, yaitu para Jemaah yang tidak mengetahui informasi rinci terkait uang kas masjid. Pemberitahuan uang kas masjid hanya dilakukan seminggu sekali pada hari Jumat. Dari permasalahan tersebut kemudian dikembangkan sistem administrasi masjid dengan metode yang digunakan adalah metode pengumpulan data dan *user stories*. Hasil penelitian ini adalah sistem informasi pengelolaan masjid yang dapat membantu pengelola masjid untuk menyusun laporan keuangan masjid pada setiap bulan hingga pengelola masjid mengambil keputusan dalam pengelolaan uang kas secara efisien dan dapat memantau penerimaan dan pengeluaran dari uang kas.

Kemudian, penelitian yang dilakukan oleh (RIZALDI, 2021), terdapat permasalahan tentang penyampaian informasi akademik yang berada di Fakultas Kedokteran Universitas Islam Indonesia yang masih dilakukan dengan konvensional. Pada cara konvensional ini membutuhkan banyak kertas, membutuhkan banyak tempat, dan membutuhkan waktu yang lebih lama. Mahasiswa dipaksa mengikuti perkembangan jadwal yang terkini dan jadwal tersebut hanya bisa didapatkan di Biro Administrasi Akademik dan Kemahasiswaan (BAAK). Dari permasalahan tersebut, terdapat solusi yakni untuk membuat sistem informasi agar mempermudah penyampaian informasi akademik. Aplikasi yang akan dikembangkan adalah aplikasi *android* yang akan digunakan dosen dan mahasiswa dan aplikasi berbasis *web* yang akan digunakan oleh pihak dari BAAK. Metode yang dipakai untuk mengembangkan aplikasi sistem informasi untuk penyampaian akademik ini menggunakan metode *waterfall*.

Terdapat penelitian dari (Baharuddin, 2021) yang membahas mengenai sistem informasi magang. Sistem informasi manajemen magang yang dibuat diharapkan dapat berguna untuk peningkatan kinerja, efektivitas, efisiensi, dan produktivitas di dalam pengelolaan magang. Metode yang digunakan adalah metode pengumpulan data dari observasi, wawancara, dan juga studi pustaka. Sistem informasi manajemen magang yang dibuat telah diuji menggunakan pengujian *Blackbox* yang hasilnya semua form dan tampilan berfungsi dengan baik.

Lalu terdapat penelitian dari (Ismail et al., 2018) yang membahas tentang magang. Mahasiswa calon guru memerlukan visi dan misi untuk menjadi seorang guru. Sudah selayaknya mahasiswa calon guru memiliki visi dan misi agar menjadi guru yang berkarakter dan profesional selagi mahasiswa calon guru tersebut belajar pada perguruan tinggi. Penlitian yang bertujuan untuk bagaimana magang dapat meningkatkan kompetensi dari calon guru. Metode yang digunakan dalam penelitian yaitu pengumpulan data melalui wawancara, observasi, dan juga dokumentasi. Hasil dari penelitian yang dilakukan yaitu kompetensi

mahasiswa yang melakukan magang mencapai standar profesi dan menjadi acuan untuk calon guru.

Penlitian yang dilakukan oleh (Amelia et al., 2021) membahas tentang sistem informasi magang. Pada UPT Teknologi dan Komunikasi (TIK) UPN Veteran Jakarta terdapat suatu masalah yaitu untuk memperoleh suatu informasi alur PKL belum tersedia sistem informasi yang memfasilitasi pendaftar dan pemagang agar memperoleh informasi tersebut. Dari permasalahan tersebut menyebabkan proses PKL menjadi lebih lama. Diperlukan adanya suatu informasi magang berbasis *web* agar dapat mengelola seluruh data terkait PKL. Sistem yang dibuat menggunakan metode *waterfall* yang bahasa pemrogramannya menggunakan PHP, *framework* Codeigniter, dan database-nya menggunakan MySql. Penelitian ini menghasilkan suatu sistem informasi magang berbasis *web* dengan fitur utama yang dimiliki, yaitu pendaftaran magang, data log harian dan kehadiran yang bisa dikelola, serta penliaian pemagang.

Selanjutnya, terdapat penelitian dari (Priskila, 2018) yang membahas mengenai metodologi *agile* dan *user stories*. Terdapat permasalahan yang membuat Perusahaan Karya Cipta Buana Sentosa dirasa kurang efektif dan efisien. Permasalahannya yaitu proses pencatatan pemasukan barang, pengeluaran barang, dan barang yang tersedia pada gudang masih menggunakan *Microsoft Excel*. Ketika informasi ketersediaan barang dibutuhkan, harus melihat *file* atau tabel satu persatu. Agar pengelolaan barang yang berada pada perusahaan tersebut lebih efektif dan efisien, maka dikembangkan suatu sistem informasi untuk persediaan barang. Metode yang digunakan dalam pengembangan perangkat lunak sistem informasi persediaan barang menggunakan pendekatan *Agile Software Development* yang berjenis *Extreme Programming (XP).* Metode *XP* dipilih karena pengembangan perangkat lunak lebih cepat dan dapat berdaptasi dengan perubahan. Analisis kebutuhan yang digunakan untuk mengembangkan sistem informasi persediaan barang ini adalah analisis kebutuhan *user stories*.

Penelitian dari (Lestari & Novita, 2019) membahas tentang metodologi dalam pengembangan perangkat lunak yang menggunakan *waterfall*. Permasalahan yang ditemui yaitu, proses pendaftaran magang yang memakan waktu lama dikarenakan proposal yang dikirim, diproses dan mahasiswa harus menunggu beberapa hari atau bahkan menunggu selama lebih dari seminggu sampai proposal yang dikirimkan disetujui oleh atasan. Mahasiswa harus memantau perkembangan dari proposal yang dikirimkan oleh mahasiswa yang akan melakukan magang. Kemudian, jika mahasiswa yang bersangkutan tidak diterima magang pada suatu tempat, harus mencari magang di tempat lain yang mana mahasiswa tersebut harus menunggu lagi agar proposal yang dikirimkan disetujui oleh atasan. Dari permasalahan tersebut, kemudian

dikembangkan sistem informasi magang berbasis *website* agar masalah tersebut dapat teratasi. Pengembangan sistem informasi magang menggunakan metode *waterfall*, bahasa pemrograman PHP, dan menggunakan *text editor* PhpStorm. Hasil dari penelitian tersebut yaitu sistem informasi magang berbasis website yang dapat digunakan untuk kegiatan magang terutama pada pendaftaran magang.

Selanjutnya, yang terakhir penelitian (Komalasari et al., 2022) yang membahas tentang sistem informasi magang dan metodologi pengembangan perangkat lunak yang menggunakan waterfall. Permasalahannya adalah pada PT. Semen Baturaja Palembang pendaftaran magang masih dilakukan manual dengan menggunakan Microsoft Word. Data yang dikelola menggunakan Micrososft Word kemungkinan masih banyak kesalahan dalam pengelolaan data. Oleh karena itu dibutuhkan suatu sistem informasi magang berbasis web yang dapat memudahkan PT. Semen Baturaja untuk pendaftaran mahasiswa magang.

Penelitian tentang sistem informasi magang yang sudah ada telah melakukan penelitian dengan metodologi *waterfall*. Belum ada penelitian yang menggunakan metodologi *agile* yang analisis desainnya menggunakan *user stories*. Oleh karena itu, pada penelitian ini akan dilakukan pengembangan *prototype* sistem informasi magang menggunakan metodologi *agile* dan analisis desainnya menggunakan *user stories*.

BAB III
ANALISIS DAN KEBUTUHAN

3.1 Studi Kelayakan (*Feasibility Study*)

Dalam pengembangan perangkat lunak, studi kelayakan dilakukan untuk menentukan apakah proyek pengembangan perangkat untuk layak untuk dilanjutkan atau tidak. Studi kelayakan yang dilakukan untuk memastikan proyek yang dikembangkan dapat dilaksanakan secara ekonomis dan teknis serta dapat memenuhi kebutuhan dari pengguna. Pada studi kelayakan dilakukan wawancara menggunakan *google form*. Pemberian kuisioner yang berupa *google form* kepada empat mahasiswa aktif dan satu dosen Program Studi Informatika – Program Sarjana, Fakulta Teknologi Industri, Universitas Islam Indonesia.

3.1.1 Proses Bisnis Magang

Secara umum, proses bisnis pada Program Studi Informatika – Program Sarjana, Fakultas Teknologi Industri, Universitas Islam Indonesia yaitu, pertama yang dilakukan oleh mahasiswa adalah mencari informasi tentang perusahaan yang menyediakan lowongan magang. Jika perusahaan penyedia lowongan magang tersebut bukan mitra, maka mahasiswa mendiskusikannya dengan tim magang program studi, akan tetapi jika tim magang program studi tidak setuju terhadap perusahaan yang didaftarkan oleh mahasiswa, maka mahasiswa mencari kembali informasi tentang perusahaan penyedia lowongan magang. Jika perusahaan penyedia lowongan magang tersebut adalah mitra jurusan, maka mahasiswa mengambil surat permohonan magang yang berasal dari unversitas kemudian ke perusahaan. Setelah pengambilan surat permohonan, mahasiswa melakukan tes yang diselenggarakan oleh perusahaan, kemudian mendapatkan pengumuman tentang diterima atau ditolaknya pengajuan magang. Jika tidak ada tes yang diselenggarakan oleh perusahaan, maka langsung mendapat pengumuman tentang diterima atau ditolaknya permohonan magang mahasiswa.

Mahasiswa yang tidak diterima magang dan tidak mendaftar magang lagi, maka mahasiswa tersebut pindah jalur dan alur dari proses magang selesai. Jika mahasiswa yang tidak diterima magang akan mendaftar magang lagi, maka mahasiswa kembali ke pencarian perusahaan yang menyediakan lowongan magang. Mahasiswa yang diterima magang pada suatu perusahaan akan menerima surat balasan dari perusahaan. Setelah mahasiswa menerima surat balasan dari perusahaan, maka mahasiswa melakukan *key-in* yang dilanjutkan dengan mahasiswa melakukan magang pada perusahaan yang sudah menerima.

3.1.2 Survey Kondisi Magang

Pada studi kelayakan dilakukan wawancara menggunakan *google form*. Pemberian kuisioner berupa *google form* kepada empat mahasiswa aktif Program Studi Informatika – Program Sarjana, Fakultas Teknologi Industri, Universitas Islam Indonesia yang telah melakukan magang serta satu dosen Program Studi Informatika – Program Sarjana, Fakultas Teknologi Industri, Universitas Islam Indonesia. Berikut pertanyaan yang diberikan beserta penjelasan dan jawaban dari pertanyaan tersebut:

a. Apa masalah yang harus diselesaikan dengan web magang yang lama (web Sekawan)?

Maksud tujuan dari pertanyaan tersebut adalah untuk menemukan dan menjabarkan tentang masalah yang perlu diselesaikan pada web magang sebelumnya yaitu web Sekawan. Dengan memahami masalah yang perlu diselesaikan ini, suatu kelemahan web magang lama ditemukan dan dapat menentukan suatu perubahan yang diperlukan. Jika masalah yang ada pada web magang yang lama dapat diatasi, maka diharapkan web magang yang baru dapat memenuhi kebutuhan dari pengguna dengan baik. Tabel 3.1 merupakan jawaban dari pertanyaan pertama.

Tabel 3.1 Jawaban Pertanyaan 1

Pengguna	Jawaban
1	- Detail dari nilai sangat lama untuk diperbarui. - Logbook belum terlalu terpakai karena maasiswa mengisi logbook pada luar web Sekawan
2	Tidak bermasalah dengan Sekawan
3	*Bug* tidak tampil pada halaman profil
4	- Perlu penambahan fitur yang mendukung kurikulum jalur magang - Fitur kurang fleksibel terkait input dan pengolahan nilai diseminasi magang dan magang belum bisa mengakomodir sistem penilaian berdasarkan RPS
5	Kurangnya fitur untuk mendukung magang seperti: - Fitur informasi lowongan magang - Forum konsultasi magang

b. Secara umum, apa tujuan dari sebuah web magang dikembangkan?

Pertanyaan tersebut dibuat untuk mendapatkan suatu pemahaman yang lebih baik mengenai tujuan umum perangkat lunak web magang dikembangkan. Tujuan dari pertanyaan tersebut adalah untuk mengetahui tujuan umum dibalik pengembangan perangkat lunak berbasis web dalam dunia magang. Penting untuk dipahami bagaimana pengembangan

berangkat lunak berbasis web ini dapat membantu dan mendukung tujuan dari magang. Tabel 3.2 merupakan jawaban dari pertanyaan kedua.

Tabel 3.2 Jawaban Pertanyaan 2

Pengguna	Jawaban
1	Untuk mengelola suatu kegiatan magang
2	Sistem manajemen untuk mahasiswa yang mengambil penjaluran magang
3	Dapat mengelola kegiatan magang dengan baik
4	Memfasilitasi mahasiswa magang dengan dosen magang
5	Mempermudah para pemagang dalam mendapatkan suatu informasi dan menemukan informasi, mengupload progress laporan magang, dan hal-hal yang menunjang keperluan magang.

c. Harapannya, pengguna dari web magang ini siapa saja?

Tujuan dari pertanyaan tersebut adalah untuk mengidentifikasi dan memahami kelompok dari pengguna yang diharapkan atau yang dituju pada penggunaan web magang. Dengan pengguna yang diketahui, tim pengembang dapat mengembangkan fitur dan fungsi sesuai web magang yang dapat sesuai dengan kebutuhan dari pengguna. Pengguna yang diketahui juga agar dapat mengarahkan pengembangan web magang yang bermanfaat secara maksimal kepada pengguna yang akan dituju. Tabel 3.3 merupakan jawaban dari pertanyaan ketiga.

Tabel 3.3 Jawaban Pertanyaan 3

Pengguna	Jawaban
1	Dosen dan mahasiswa
2	Mahasiswa penjaluran magang
3	Pelaku magang
4	Mahasiswa dan dosen
5	Mahasiswa, dosen pembimbing, dosen pembimbing akademik

d. Bagaimana proses bisnis yang diinginkan oleh pengguna?

Pada pertanyaan ini, tujuannya adalah untuk mendapatkan tentang pemahaman tentang langkah-langkah, interaksi, dan aliran informasi yang diharapkan oleh pengguna dalam menjalankan aktivitas bisnis pada web magang. Proses bisnis yang dipahami dapat menjadi suatu pemahaman yang baik untuk merancang fungsionalitas, fitur yang diinginkan oleh pengguna dan web magang dapat mempermudah proses bisnis. Tabel 3.4 merupakan jawaban dari pertanyaan keempat.

Tabel 3.4 Jawaban Pertanyaan 4

Pengguna	Jawaban
1	- Pengajuan magang - Perubahan status penerimaan mahasiswa magang
2	Sistem yang dapat menunjang kegiatan magang mahasiswa
3	Mudah digunakan
4	Simple, mudah, efektif, dan efisien
5	Simple dan sesuai dengan harapan dan kebutuhan dari pengguna

e. Kebutuhan sistem apa saja yang diinginkan oleh pengguna?

Tujuan dari pertanyaan tersebut adalah untuk menentukan kebutuhan dari web magang yang diinginkan oleh pengguna, sehingga perancangan dan pengembangaan web magang dapat sesuai dengan harapan dan kebutuhan dari pengguna dengan memahami kebutuhan sistem tersebut. Kebutuhan fungsional dan nonfungsional termasuk dalam kategori ini. Tabel 3.5 merupakan jawaban dari pertanyaan kelima.

Tabel 3.5 Jawaban Pertanyaan 5

Pengguna	Jawaban
1	- Sistem dapat mengajukan permohonan magang dengan formulir pendataan disertai dengan status permohonan ditolak/diterima. - Sistem dapat memperlihatkan profil perushaan yang menyediakan lowongan magang.
2	- Sistem dapat memperlihatkan profil perusahaan yang menyediakan lowongan magang. - Sistem dapat digunakan untuk mendaftarkan perusahaan
3	- Sistem dapat mencatat kegiatan harian magang - Sistem dapat memperlihatkan informasi magang - Sistem dapat memperlihatkan profil perusahaan penyedia lowongan magang
4	- Sistem dapat digunakan untuk penilaian - Sistem dapat digunakan untuk mengupload dokumen diseminasi magang
5	- Sistem dapat digunakan untuk menjadi forum konsultasi - Sistem dapat memperlihatkan berita lowongan magang.

3.1.3 Kendala yang Dihadapi

Ada beberapa kendala dalam kegiatan magang pada Program Studi Informatika – Program Sarjana, Fakultas Teknologi Industri, Universitas Islam Indonesia, yaitu tidak adanya

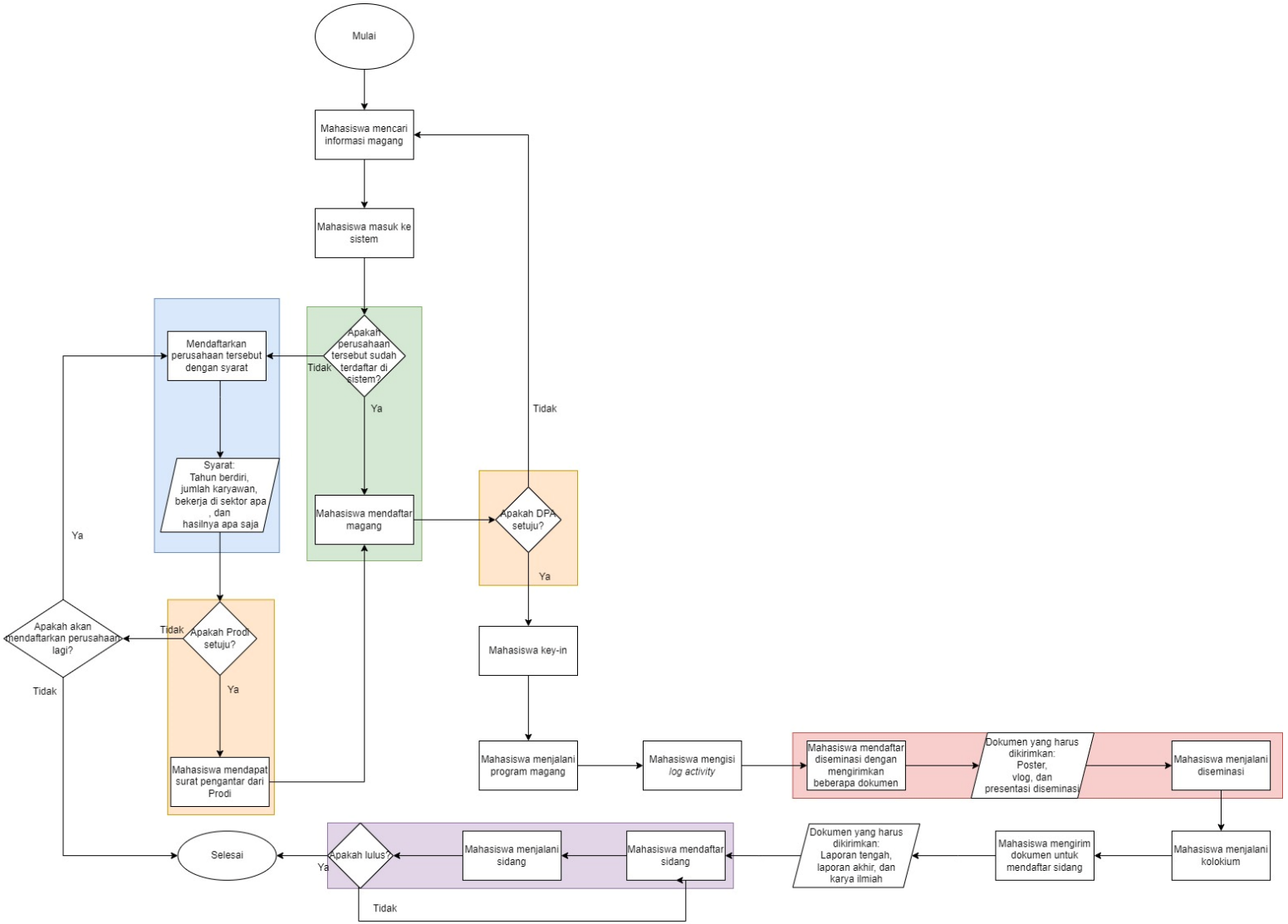
sistem tersendiri untuk kegiatan magang. Sistem yang terdahulu belum memiliki fitur seperti informasi lowongan magang, fitur yang kurang sesuai dengan kurikulum jalur magang, masih ada *bug* seperti data yang tidak tampil pada halaman profil, detail nilai sangat lama untuk diperbarui. Terdapat perusahaan yang kurang layak untuk menyediakan lowongan magang, termasuk kurang layaknya mahasiswa yang akan melakukan magang yang mungkin seharusnya mahasiswa tersebut bisa melamar ke perusahaan atau posisi magang yang sesuai.

3.1.4 Solusi yang Ditawarkan

Dari masalah tersebut, diperlukan sebuah sistem yang dapat membantu menangani masalah yang ada. Diawali dengan mahasiswa masuk ke sistem untuk menawarkan perusahaan-perusahaan yang menyediakan lowongan magang. Perusahaan yang ditawarkan harus memenuhi persyaratan seperti tahun berdiri, jumlah karyawan, bekerja di sektor apa, dan hasilnya apa saja. Mahasiswa juga bisa masuk ke halaman perusahaan yang sudah didaftarkan oleh mahasiswa yang lain, hal ini untuk menghindari duplikat perusahaan yang sama.

Bagian penanggung jawab dan program studi bisa melakukan persetujuan magang, seperti perusahaan yang didaftarkan layak atau tidak untuk dijadikan tempat magang dan mahasiswanya layak atau tidak untuk melakukan magang di perusahaan tersebut. Setelah disetujui, mahasiswa yang bersangkutan menjalani program magang dan ditambahkan dosen pembimbing. Pada saat mahasiswa menjalankan program magang, mahasiswa mengisi log activity. Catatan proses bimbingan dapat disimpan, dapat mengatur jadwal bimbingan, dan dosen bisa melakukan persetujuan antara bimbingan atau tidak.

Berikutnya, mahasiswa mendaftar diseminasi magang dan ada dokumen yang dikirim untuk menjadi persyaratan yaitu poster, vlog, dan presentasi diseminasi. Setelah diseminasi, ada beberapa dokumen yang harus dikirimkan lagi yaitu laporan tengah, laporan akhir, dan karya ilmiah. Setelah kolokium, mahasiswa dapat mendaftar sidang pendadaran dan kemudian dosen dapat memutuskan mahasiswa yang sudah sidang pendadaran lulus atau tidak dan kemudian mahasiswa mendapat nilai. Gambar 3.1 merupakan alur dari Sistem Informasi Jalur Magang.



Gambar 3.1 Bagan Alur dari Sistem Informasi Jalur Magang

3.1.5 Tingkat Kelayakan

Pada Informatika Universitas Islam Indonesia sudah terdapat website Sekawan, namun belum terdapat kebutuhan untuk memenuhi proses bisnis magang, maka dikembangkan sistem informasi manajemen baru jalur magang untuk memenuhi proses bisnis jalur magang. Sistem informasi manajemen jalur magang yang baru merupakan aplikasi website. Pengembangan aplikasi website sistem informasi manajemen jalur magang menggunakan bahasa pemrograman PHP dan framework Laravel. Pengembangan dilakukan secara tim menggunakan metodologi *agile* yang rekayasa kebutuhan perangkat lunaknya menggunakan *user stories*. Tim terdiri dari Muhammad Fariz Iqbal dan Adi Setiawan. Muhammad Fariz Iqbal membuat *database*, sedangkan Adi Setiawan membuat desain antarmuka dan implementasi.

Melihat banyaknya jumlah mahasiswa Informatika Universitas Islam Indonesia yang membutuhkan informasi terkait magang untuk melakukan tugas akhir didapatkan sebuah ide untuk membuat sistem informasi yang menyediakan lowongan magang. Dengan begitu peluang yang didapatkan sangat besar, sehingga sistem yang dibuat dapat menguntungkan pengguna maupun pengembang.

3.2 Elisitasi dan Analisis Kebutuhan (*Requirements Elicitation and Analysis)*

Elisitasi dan analisis kebutuhan ini bertujuan unuk mengumpulkan informasi dan memahami kebutuhan pengguna dari perangkat lunak yang dikembangkan. Pada tahap ini didapatkan *user stories* dan pengguna terhadap Sistem Informasi Jalur Magang. *User stories* yang didapatkan merupakan gambaran dari fungsionalitas Sistem Informasi Jalur Magang serta pengguna yang didapatkan pada tahap ini adalah pengguna yang menggunakan Sistem Informasi Jalur Magang.

3.2.1 Elisitasi

Elisitasi digunakan untuk mengumpulkan peran dan kebutuhan dari pengguna. Pengumpulan peran dan kebutuhan pengguna ini mengacu pada studi kelayakan yang telah dilakukan. Dari pengumpulan ini telah teridentifikasi pengguna dan fungsionalitas dari Sistem Informasi Jalur Magang.

3.2.2 Daftar Peran

Ada beberapa pelaku/*role* di dalam sistem ini yang teridentifikasi dari proses bisnis dan wawancara yang telah dilakukan, yaitu:

a. Mahasiswa, yaitu mahasiswa aktif Informatika Universitas Islam Indonesia yang melaksanakan tugas akhir jalur magang.
b. Dosen pembimbing akademik (DPA), yaitu dosen yang menjadi wali untuk setiap mahasiswa, yang membantu mahasiswa dalam mengambil keputusan terkait pelaksanaan studi di universitas, termasuk dalam pemilihan tugas akhir.
c. Dosen pembimbing, yaitu dosen yang diputuskan oleh program studi untuk membimbing mahasiswa melaksanakan tugas akhir.
d. Program studi, yaitu pengelola program studi Informatika Universitas Islam Indonesia yang mengelola pelaksanaan tugas akhir mahasiswa.

3.2.3 Daftar *User Stories*

Proses bisnis diperoleh dengan wawancara dari salah satu pengguna dengan mendengarkan cerita dari koordinator jalur magang dan kuisoner yang diberikan kepada empat mahasiswa dan satu dosen Program Studi Informatika – Program Sarjana, Fakultas Teknologi Industri, Universitas Islam Indonesia. Pada wawancara tersebut, dliakukan pemahaman dan mengidentifikasi proses bisnis. Dari cerita proses bisnis yang telah dikaji sebelumnya, diperoleh daftar *user stories* yang akan menjadi suatu panduan dalam pengembangan Sistem Informasi Jalur Magang, sebagai berikut:

a. Sebagai mahasiswa saya ingin mendaftar magang sehingga saya bisa mendapatkan pengalaman magang.
b. Sebagai mahasiswa saya ingin melakukan daftar perusahaan tempat magang agar saya dapat magang pada perusahaan yang saya daftarkan.
c. Sebagai mahasiswa saya ingin melihat perusahaan penyedia lowongan yang telah didaftarkan oleh mahasiswa lain sehingga saya bisa langsung daftar magang dan tidak perlu mendaftarkan perusahaan yang sama.
d. Sebagai mahasiswa saya ingin dapat mengisi log book agar kegiatan selama magang tercatat.
e. Sebagai mahasiswa saya ingin dapat daftar diseminasi magang sehingga saya bisa melakukan diseminasi magang.
f. Sebagai mahasiswa saya ingin dapat melakukan pendaftaran sidang pendadaran sehingga program studi dapat menjadwalkan sidang pendadaran saya.
g. Sebagai mahasiswa saya ingin dapat mengunggah dokumen-dokumen seperti laporan tengah, laporan akhir, dan karya ilmiah sehingga tidak mengunggah dokumen pada website yang berbeda.

h. Sebagai mahasiswa saya ingin dapat melihat laporan tengah, laporan akhir, dan karya ilmiah karya dari mahasiswa lain sehingga saya bisa belajar dari laporan-laporan yang ada.
i. Sebagai dosen pembimbing akademik saya ingin dapat memantau kelayakan dari perusahaan yang didaftarkan oleh mahasiswa sehingga saya dapat melakukan suatu persetujuan.
j. Sebagai dosen pembimbing saya dapat mengunggah nilai mahasiswa yang melakukan program magang sehingga nilai yang saya berikan bisa menjadi tolak ukur dari hasil mahasiswa yang menjalani magang.
k. Sebagai dosen pembimbing saya ingin dapat melihat laporan-laporan yang sudah diunggah oleh mahasiswa sehingga saya bisa belajar dari laporan-laporan yang ada.
l. Sebagai dosen pembimbing saya ingin dapat memetakan judul-judul dokumen dari waktu yang lama hingga terbaru sehingga dokumennya tertata rapih.
m. Sebagai dosen pembimbing akademik saya ingin melakukan approval permagangan sehingga mahasiswa dapat diproses lebih lanjut oleh program studi.
n. Sebagai Program Studi saya ingin dapat melakukan approval terhadap perusahaan magang sehingga perusahaan yang telah didaftarkan oleh mahasiswa bisa terdaftar di sistem dan mahasiswa bisa mendaftar magang.
o. Sebagai Program Studi saya ingin dapat menambahkan dosen pembimbing untuk mahasiswa yang melakukan magang, sehingga mahasiswa yang magang tersebut mendapat bimbingan dari dosen pembimbing.
p. Sebagai dosen pembimbing atau mahasiswa saya ingin dapat mencatat proses bimbingan kemudian menyimpannya sehingga saya tidak melupakan proses bimbingan.

3.3 Spesifikasi Kebutuhan (*Requirements Specification*)

Pada spesifikasi kebutuhan dilakukan prioritisasi terhadap *user stories*. Hasil dari prioritisai berupa *task* yang akan dikerjakan oleh pengembang. Proses prioritiasasi dilakukan dengan menggunakan aplikasi *Trello*.

3.3.1 Alat Bantu

Alat bantu yang digunakan untuk melakukan suatu prioritas dan pembagian *task* adalah *Trello*. *Trello* adalah salah satu aplikasi yang sangat popular dan dapat digunakan secara luas sebagai platform untuk kolaborasi tim. Antarmuka dari *Trello* yang mudah digunakan dan sederhana dapat digunakan sebagai fasilitas untuk kerja sama antar tim ataupun dengan divisi lain secara tim. Aplikasi *Trello* cocok untuk tim yang menerapkan pengembangan perangkat lunak yang menerapkan prinsip dari *agile*.

Fitur utama dari *Trello* yaitu *board*, *list*, dan *card*, dapat memberikan keleluasaan bagi suatu organisasi atau tim pengembang untuk dapat mengatur dan memprioritaskan proyek yang dikerjakan dengan menarik, bermanfaat, dan fleksibel. Pada *Trello*, *board* digunakan untuk membuat suatu lingkungan kerja yang dapat diorganisir secara jelas. *List* dan *board* adalah sebuah fitur yang dapat membantu untuk mengelompokkan dan mengatur tugas.

Fitur yang lain dari *Trello* adalah komentar, batas waktu, notifikasi dan lampiran yang dapat membantu tim pengembang melakukan komunikasi dan melakukan kolaborasi. *Trello* dapat memungkinkan pengguna menghubungkan alat kerja secara mudah dan juga dapat meningkatkan produktivitas dari pengguna. Pengguna dapat memberikan label, ceklis, dan lainnya pada *card* untuk memberikan lebih banyak tentang detail dan informasi yang dibutuhkan.

Secara keseluruhan, *Trello* merupakan suatu alat yang berguna untuk tim yang menggunakan pendekatan yang kolaboratif dan adaptif pada saat pengembangan proyek. Dengan menggunakan fitur yang tersedia pada *Trello*, tim dapat mengatur, menyelesaikan, dan mengelola tugas dengan baik, serta dapat meningkatkan kerjasama antar tim dalam pencapaian kesuksesan proyek.

3.3.2 Prioritasi & Alokasi

Tahapan prioritasi & alokasi dilakukan untuk menemukan kebutuhan yang penting dan mana yang tidak penting. Terdapat dua kriteria yakni kriteria berdasarkan kepentingan dan kriteria berdasarkan urgensi. Kriteria berdasarkan kepentingan berisi elemen *Important* dan *Not Important*. Elemen *Important* menunjukkan pada *task* yang sangat penting dan harus terdapat dalam proses bisnis, sedangkan elemen *not important* adalah *task* yang bersifat tambahan atau tidak krusial pada proses bisnis. Kriteria berdasarkan urgensi terdapat tiga elemen dalam pengelompokan prioritas, yaitu *high, medium,* dan *low*. *Task* dengan elemen *high* harus dikerjakan terlebih dahulu kemudian dilanjutkan *task* dengan elemen *medium* serta yang terakhir mengerjakan *task* dengan elemen *low*. *Task* dengan elemen *low* akan dikerjakan setelah semua *task* yang bersifat *medium* telah selesai dikerjakan (Iqbal, 2021).

Pengembang menggunakan suatu aplikasi yang bernama *Trello* untuk mengatur dan mengelola tugas yang akan dikerjakan dalam pengembangan perangkat lunak. *Trello* merupakan suatu aplikasi yang berguna untuk memudahkan berkomunikasi dan berkolaborasi antar tim.

Setelah *user stories* telah terbagi menjadi tiga *task* utama dalam pengembangan perangkat lunak, selanjutnya menyerahkan tugas-tugas tersebut kepada anggota tim

pengembang yang bertanggung jawab untuk melakukannya. *Task* analisis desain dan implementasi diberikan kepada Adi Setiawan dan Muhammad Fariz Iqbal diberikan tugas untuk mengerjakan rancangan tabel.

Pada aplikasi *Trello,* tim pengembang dapat membuat suatu *board* khusus yang terdapat pada *Trello* untuk mengatur dan melacak setiap dari tugas yang dikerjakan. Setiap anggota tim pengembangan dapat untuk melihat daftar dari tugas-tugas, memperbarui status dari tugas, dan dapat memberikan suatu komentar tentang bagaimana perkembangan tugas tersebut.

Tim pengembang juga dapat menggunakan aplikasi *Trello* untuk memprioritaskan setiap tugas berdasarkan *urgensi* dan tingkat dari kepentingan tugas tersebut. *Trello* juga dapat mengatur *deadline,* mengatur suatu tabel, dan dapat menambahkan *checklist* agar dapat dipastikan proses pengembangan perangkat lunak dapat berjalan dengan lancar.

Untuk melakukan komunikasi dan koordinasi pada setiap task di *Trello,* dapat digunakan fitur komentar untuk dapat berkomunikasi, berbagi informasi, dan memberikan suatu tanggapan. *Trello* membuat suatu pengelolaan tugas pengembangan perangkat lunak lebih sistematis, efektif, dan terorganisir. Aplikasi *Trello* ini dapat memudahkan untuk mengawasi, mengatur, dan mengoordinasikan tahapan pengembangan perangkat lunak yang memungkinkan tim pengembang bekerja sama dan mencapai tujuan dengan lebih baik.

Melalui pendekatan yang dilakukan ini, tim pengembang dapat menjalankan suatu proses pengembangan perangkat lunak dengan efisien dan terstruktur. Pembagian *user stories* menjadi *task* yang terorganisir, dan prioritisasi, tim pengembang dapat fokus pada *task* yang paling penting dan memastikan bahwa fitur yang paling penting di dalam proses bisnis diutamakan.

Gambar 3.2 Contoh Task yang Didapatkan

Gambar 3.2 merupakan contoh *task* yang dikerjakan. Adapun keterangan dari warna-warna yang berada dalam *task*. Warna merah menunjukkan *task* yang prioritasnya *important high*, warna orange meunjukkan prioritas *important medium*, warna kuning menunjukkan *important low*, dan terakhir warna hitam menunjukkan *task* yang memiliki prioritas *not important medium* dan *not important low*. Selain itu, setiap dari *task* juga terdapat suatu deskripsi yang menjelaskan tentang rincian dari *task* tersebut. Dengan menggunakan kode warna dan deskripsi yang digunakan, tim pengembang dapat jelas memahami tentang prioritas dan konteks dari setiap *task* yang dikerjakan.

3.3.3 Detil Spesifikasi

Setelah 16 *user stories* teridentifikasi dari proses elisitasi dan analisis kebutuhan, langkah selanjutnya yang dilakukan adalah dengan melakukan spesifikasi kebutuhan. *User stories* yang telah didapatkan kemudian dibagi menjadi 3 *task* utama yang perlu diselesaikan pada pengembangan perangkat lunak.

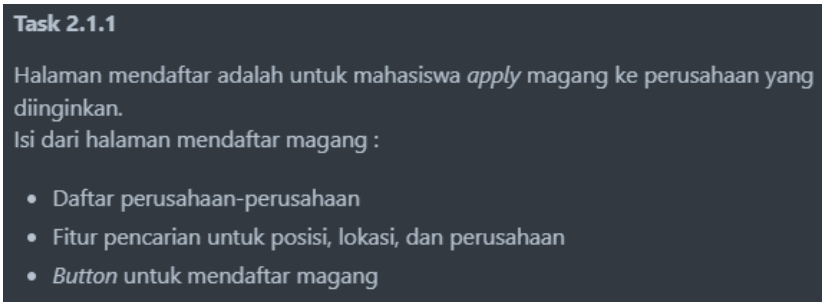
Pertama, terdapat *task* analisis desain yang melibatkan suatu pemahaman yang mendalam untuk kebutuhan pengguna dan merancang solusi yang tepat. *Task* analisis desain fokusnya adalah pengembangan desain perangkat lunak yang efektif dan dapat memenuhi kebutuhan yang telah teridentifikasi.

Yang kedua, teradapat *task* merancang tabel. Merancang tabel penting untuk membangun suatu struktur dan hubungna relasi antar data. Pada *task* merancang tabel ini berguna untuk menyimpan suatu informasi yang penting pada perangkat lunak.

Ketiga adalah *task* implementasi. *Task* implementasi melibatkan penerjemahan suatu desain dan kebutuhan yang didapatkan menjadi sebuah kode program yang dapat digunakan. *Task* ini mencakup penulisan kode program dan integrasi dengan komponen lain pada pengembangan perangkat lunak.

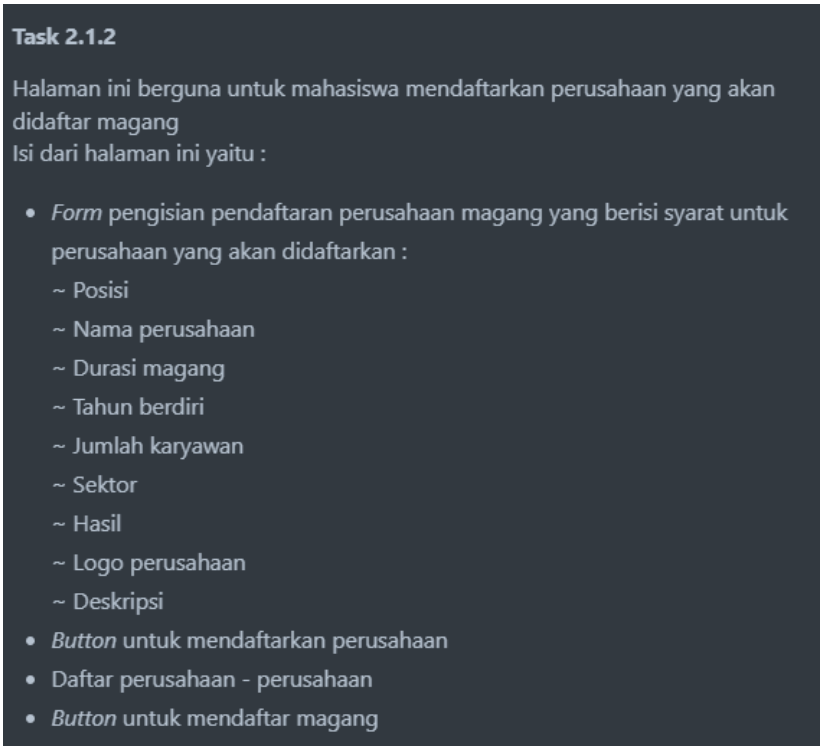
Dengan demikian, total *task* yang telah diperoleh adalah sebanyak 41 *task*. Setiap dari *task* memiliki suatu peran yang penting pada pengembangan perangkat lunak dan *task* ini perlu diselesaikan bertahap dan terkoordinasi agar dapat mencapai perangkat lunak yang dikembangkan sesuai pada kebutuhan.

Penyelesaian dari *task* yang sudah diperoleh menjadi suatu prioritas pada proses pengembangan perangkat lunak, dikarenakan setiap dari tugas yang diselesaikan membuat pengembangan perangkat lunak menjadi lebih dekat dengan tujuan akhir. Dengan mendeskripsikan dan membagi *user stories* menjadi *task* yang nantinya akan menjadi tugas pengembang, proses untuk pengembangan perangkat lunak menjadi lebih terstruktur dan dijalankan dengan tepat. Berikut merupakan Gambar 3.3, Gambar 3.4, Gambar 3.5, Gambar 3.6, Gambar 3.7, Gambar 3.8, Gambar 3.9, Gambar 3.10, dan Gambar 3.11 yang merupakan contoh deskripsi dari masing masing task analisis desain, merancang tabel, dan implementasi.



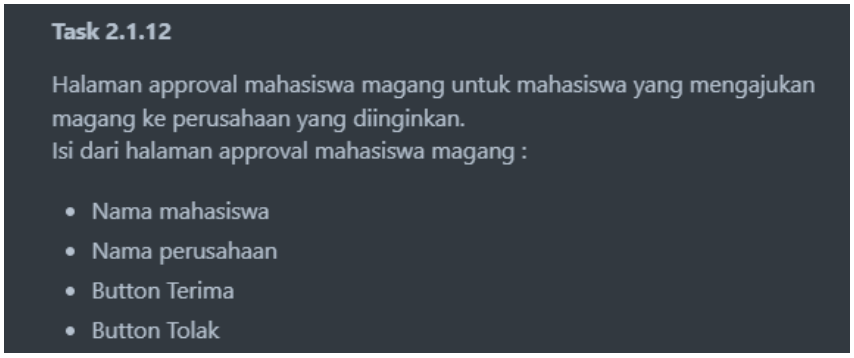
Gambar 3.3 Deskripsi *Task* Merancang Antarmuka Halaman Mendaftar Magang

Gambar 3.3 merupakan *task* pertama dalam pengembangan perangkat lunak dengan kriteria kepentingan *Important* serta kriteria urgensi *High*. *Task* ini membuat rancangan antarmuka halaman mendaftar magang untuk *role* mahasiswa.



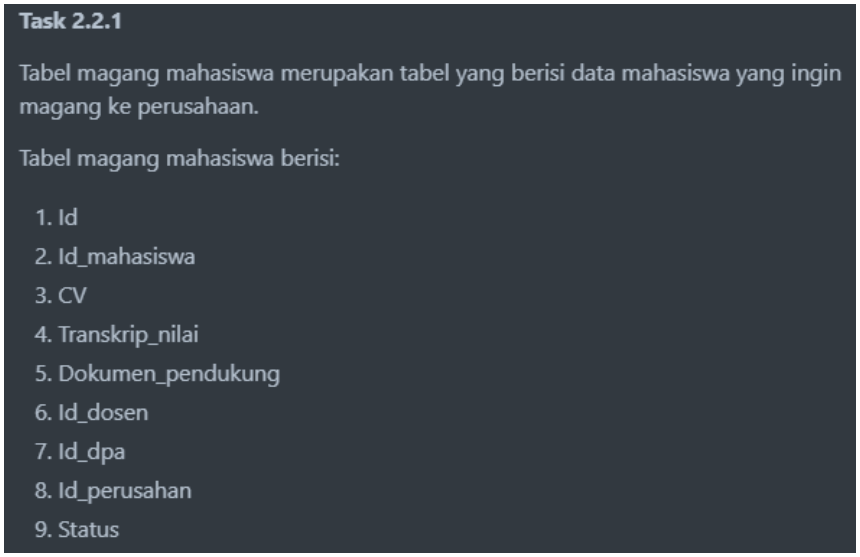
Gambar 3.4 Deskripsi Task Merancang Antarmuka Halaman Pendaftaran Perusahaan Tempat Magang

Gambar 3.4 merupakan *task* kedua dalam pengembangan perangkat lunak dengan kriteria kepentingan *Important* serta kriteria urgensi *High*. *Task* ini membuat rancangan antarmuka halaman pendaftaran perusahaan tempat magang untuk *role* mahasiswa.



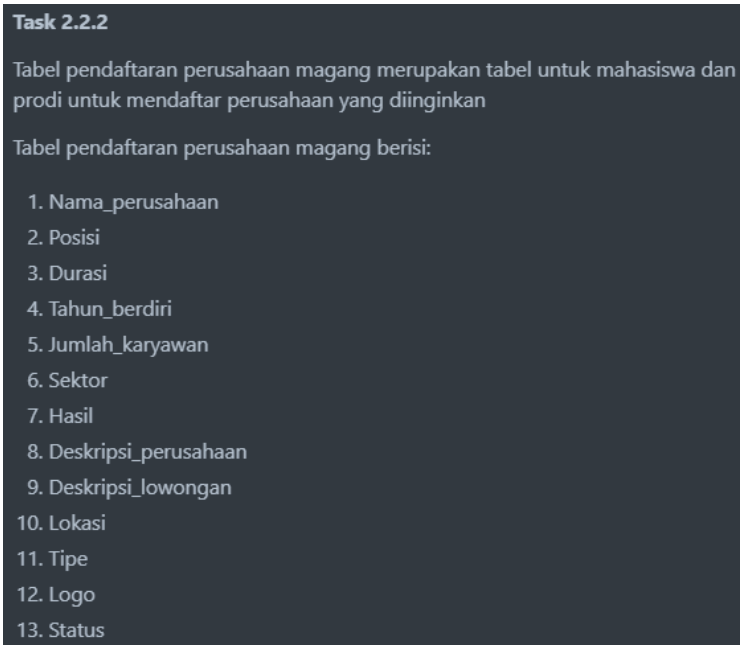
Gambar 3.5 Deskripsi *Task* Merancang Antarmuka Halaman *Approval* Mahasiswa Magang

Gambar 3.5 merupakan *task* dalam pengembangan perangkat lunak dengan kriteria kepentingan *Important* serta kriteria urgensi *High*. *Task* ini membuat rancangan antarmuka halaman *approval* mahasiswa magang untuk *role* mahasiswa.



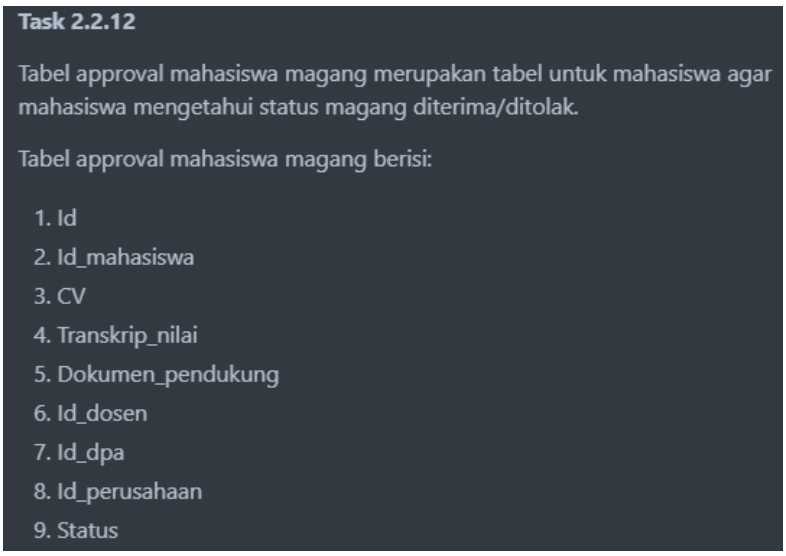
Gambar 3.6 Deskripsi *Task* Merancang Tabel Magang Mahasiswa

Gambar 3.6 merupakan *task* dalam pengembangan perangkat lunak dengan kriteria kepentingan *Important* serta kriteria urgensi *High*. *Task* ini membuat rancangan tabel mendaftar magang untuk *role* mahasiswa.



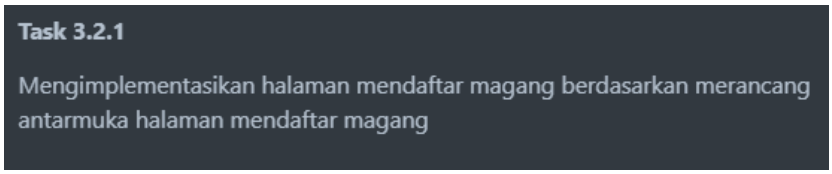
Gambar 3.7 Deskripsi Task dari Merancang Tabel Pendaftaran Perusahaan Magang

Gambar 3.7 merupakan *task* dalam pengembangan perangkat lunak dengan kriteria kepentingan *Important* serta kriteria urgensi *High*. *Task* ini membuat rancangan tabel pendaftaran perusahaan magang untuk *role* mahasiswa.



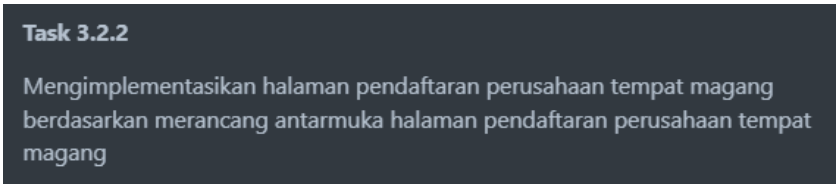
Gambar 3.8 Deskripsi Task dari Merancang Tabel *Approval* Mahasiswa Magang

Gambar 3.8 merupakan *task* dalam pengembangan perangkat lunak dengan kriteria kepentingan *Important* serta kriteria urgensi *High*. *Task* ini membuat rancangan tabel *approval* mahasiswa magang untuk *role* mahasiswa.



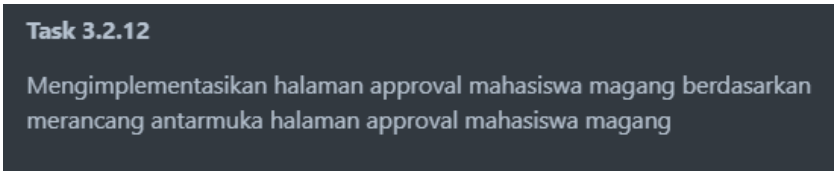
Gambar 3.9 Deskripsi *Task* Implementasi Halaman Mendaftar Magang

Gambar 3.9 merupakan *task* dalam pengembangan perangkat lunak dengan kriteria kepentingan *Important* serta kriteria urgensi *High*. *Task* ini merupakan implementasi rancangan antarmuka halaman mendaftar magang serta rancangan tabel mendaftar magang untuk *role* mahasiswa.



Gambar 3.10 Deskripsi *Task* Implementasi Halaman Pendaftaran Perusahaan Tempat Magang

Gambar 3.10 merupakan *task* dalam pengembangan perangkat lunak dengan kriteria kepentingan *Important* serta kriteria urgensi *High*. *Task* ini merupakan implementasi rancangan antarmuka halaman pendaftaran perusahaan tempat magang serta rancangan tabel pendaftaran perusahaan tempat magang untuk *role* mahasiswa.



Gambar 3.11 Deskripsi Task Implementasi Halaman *Approval* Mahasiswa Magang

Gambar 3.11 merupakan *task* dalam pengembangan perangkat lunak dengan kriteria kepentingan *Important* serta kriteria urgensi *High*. *Task* ini merupakan implementasi rancangan antarmuka halaman *approval* mahasiswa magang serta rancangan tabel *approval* mahasiswa magang untuk *role* mahasiswa.

BAB IV
VALIDASI KEBUTUHAN

Dalam tahap validasi, yang pertama kali dilakukan adalah pengecekan konsistensi (*consistency*) dari pengguna sistem. Dalam *user stories* terdapat empat role yaitu, mahasiswa, dosen, dosen pembimbing akademik, dan program studi. *User stories* untuk role mahasiswa sebanyak sembilan, *user stories* dosen pembimbing sejumlah empat, *user stories* dosen pembimbing akademik sejumlah tiga, dan jumlah *user stories* program studi berjumlah dua. Hal ini sesuai dengan proses bisnis dari pelaksanaan magang yaitu mahasiswa yang ingin menjalani program magang mendaftar terlebih dahulu dan perusahaan yang akan menjadi tempat magang juga didaftarkan. Setelah mahasiswa dan perusahaan terdaftar maka akan dilakukan persetujuan oleh dosen pembimbing akademik dan prodi. Perusahaan yang didaftarkan akan diseleksi oleh prodi kemudian mahasiswa yang mendaftar di perusahaan tersebut akan dilakukan proses persetujuan oleh dosen pembimbing akademik. Langkah selanjutnya setelah mahasiswa dan perusahaan terdaftar disetujui, mahasiswa menjalani program magang. Diseminasi magang dilakukan oleh mahasiswa yang menjalani program magang yang kemudian dilakukan kolokium setelah diseminasi magang. Selanjutnya mahasiswa melakukan sidang, kelulusan akan diberitahukan dosen.

4.1 Pengecekan Konsistensi

Selanjutnya dilakukan pengecekan konsistensi untuk fungsionalitas sistem. Semua u*ser stories* tepat, disesuaikan dengan alur proses bisnis. Tanpa pengecekan konsistensi, semua tidak bisa diyakinkan. Pengecekan konsistensi tidaklah terlalu sulit, dikarenakan pengecekan konsistensi ini relatif sederhana (Laplante, 2016). Meskipun terdapat variabel logis yang banyak, tidak akan mempengaruhi evaluasi proposisi gabungan. Pengecekan konsistensi dapat dilakukan menggunakan alat pemeriksa konsistensi otomatis atau menggunakan *spreadsheet*. Akan tetapi, pembacaan informal sudah cukup untuk pengecekan konsistensi dikarenakan pembacaan informal tidak menyatakan adanya ketidak konsistenan yang jelas.

Pengecekan konsistensi terdapat pada Tabel 4.1. Dari pengecekan tersebut, secara umum setiap satu proses bisnis dimodelkan ke dalam satu *user stories*, kecuali proses bisnis nomor 3 dan nomor 6, karena satu proses bisnis dimodelkan menjadi beberapa *user stories*. Terdapat 3 *user stories* pada proses bisnis nomor 3 dan 2 *user stories* pada proses bisnis nomor 6.

Tabel 4.1 Pengecekan Konsistensi

Proses Bisnis	***User Stories***	**Konsisten?**
1. Mahasiswa melakukan pendaftaran magang	Sebagai mahasiswa saya ingin daftar magang sehingga saya bisa mendapatkan pengalaman magang.	✓
2. Mahasiswa masuk ke sistem untuk menawarkan perusahaan-perusahaan yang menyediakan lowongan magang.	Sebagai mahasiswa saya ingin mendaftarkan perusahaan tempat magang agar saya dapat magang pada perusahaan tersebut.	✓
3. Bagian penanggung jawab dan program studi melakukan seleksi terhadap mahasiswa dan perusahaan yang menyediakan lowongan magang	- Sebagai dosen saya dapat memantau kelayakan mahasiswa yang akan mendaftar magang di perusahaan agar saya dapat melakukan persetujuan. - Sebagai dosen pembimbing akademik saya ingin melakukan *approval* permagangan sehingga mahasiswa dapat diproses kelanjutannya oleh program studi. - Sebagai Program Studi saya ingin dapat melakukan *approval* terhadap perusahaanmagang sehingga perusahaan yang didaftarkan oleh mahasiswa bisa terdaftar di sistem dan mahasiswa bisa mendaftar magang.	✓
4. Mahasiswa menjalani program magang dandosen pembimbing ditambahkan.	Sebagai Program Studi saya ingin dapat menambahkan dosen pembimbing sehingga mahasiswa yangmagang mendapat bimbingan dari dosen pembimbing.	✓
5. Mahasiswa mengisi *log activity*.	Sebagai mahasiswa saya ingin dapat mengisi *log activity* agar kegiatan magang tercatat.	✓
6. Mahasiswa melakukan pendaftaran diseminasi magang dan mengirimkan beberapa dokumen.	- Sebagai mahasiswa saya ingin mendaftar diseminasi magang sehingga saya dapat melakukan diseminasi magang.	✓

	- Sebagai mahasiswa saya ingin dapat mengunggah dokumen-dokumen seperti laporan tengah, laporan akhir, dan karya ilmiah sehingga tidak mengunggah dokumen pada website yang berbeda.	
7. Mahasiswa mendaftar sidang pendadaran dan dosen memberikan nilai kepada mahasiswa.	Sebagai mahasiswa saya ingin dapat melakukan pendaftaran sidang pendadaran sehingga program studi dapat menjadwalkan sidang pendadaran saya.	✓

Dapat disimpulkan dari Tabel 4.1 bahwa pemeriksaan yang dilakukan sesuai dengan kebutuhan. Setiap proses bisnis konsisten dengan *user stories*, ini artinya setiap dari aktivitas yang ada pada proses bisnis, harus sejalan dengan *user stories* yang ditentukan. Setiap dari aktivitas yang dilakukan dalam proses bisnis harus sesuai dengan tujuan, nilai, dan kebutuhan dari pengguna yang telah diungkapkan dalam *user stories*. Konsistensi antara proses bisnis dan *user stories* dijaga agar setiap aktivitas yang berada pada proses bisnis harus sesuai dengan *user stories*.

4.2 Pengecekan Kebenaran dan Kelengkapan

Selanjutnya dilakukan pengecekan atas kebenaran dan kelengkapan (*correctness* and *completeness*), pada tahap analisis ini. Tahapan ini dilakukan pengecekan atas definisi masalah, tujuan sistem, identifikasi pengguna sistem, fungsionalitas utama, fungsionalitas pendukung, interaksi pengguna-fungsionalitas, *user stories*, kegunaan *user stories*, dan *task*. Berikut penjelasan dari aspek-aspek tersebut:

a. Definisi Masalah

Definisi masalah membantu tim pengembang untuk mendapat pemahaman tentang masalah yang ingin diselesaikan. Hal ini penting dikarenakan pengecekan terhadap definisi masalah tersbut memverifikasi adanya permasalahan yang akan diatasi, memastikan pemahaman yang akurat tentang masalah, dan relevansinya dengan kebutuhan pengguna yang terlibat. Pengecekan definisi masalah ini berperan untuk memastikan tim pengembang memiliki suatu landasan yang kuat dalam mengembangkan solusi yang tepat dan juga efektif, fokusnya terhadap kebutuhan nyata dan memcahkan suatu masalah yang signifikan.

b. Tujuan Sistem

Pengecekan kebenaran dan kelengkapan terhadap tujuan sistem untuk memastikan bahwa tujuan pengembangan perangkat lunak telah ditetapkan jelas. Pengcekan ini diperlukan untuk memahami tim pengembang tentang sasaran dan visi yang ingin dicapai, sehingga pengembang dapat mengarahkan pengembangan perangkat lunak secara tepat. Tim pengembang mengambil keputusan yang tepat, dapat mengalokasikan sumber daya secara efisien, merancang dan pengimpletasian fitur yang sesuai. Pengecekan terhadap tujuan sistem juga dapat meminimalisirkan risiko dari pengembangan yang tidak mengarah dan tidak sesuai dengan tujual awal, sehingga perangkat lunak yang dihasilkan dapat lebih efektif dan sesuai harapan pengguna.

c. Identifikasi Pengguna Sistem

Pengecekan kebenaran dan kelengkapan terhadap identifikasi pengguna dari sistem dapat membantu untuk memastikan semua pemangku kepentingan yang relevan telah diidentifikasi secara tepat. Pengembang dapat memiliki suatu pemahaman yang jelas tentang siapa yang akan menggunakan sistem tersebut dan dapat memahami kebutuhan dari pengguna. Dengan informasi yang dihasilkan dari identifikasi pengguna sistem ini, tim pengembang dapat merancang suatu pengalaman pengguna yang sesuai, dapat memenuhi kebutuhan dan harapan dari pengguna. Pengecekan identifikasi pengguna sistem juga dapat membantu untuk pencegahan agar tidak salah dalam mengidentifikasi pemangku kepentingan yang tidak sesuai. Dengan ini, pengembangan perangkat lunak dapat lebih terarah yang menyebabkan pengembangan solusi dapat lebih baik dan sesuai dengan kebutuhan dari pengguna.

d. Fungsionalitas Utama dan Pendukung

Pengecekan kebenaran dan kelengkapan pada aspek fungsionalitas utama dan pendukung dapat membantu untuk memastikan suatu identifikasi akurat terhadap semua fitur dan kemampuan perangkat lunak yang diharapkan. Pada proses ini, tim pengembang merancang suatu arsitektur sistem yang sesuai dan mengembangkan suatu fungsionalitas yang tepat dengan kebutuhan dari pengguna. Dengan memastikan setiap fitur yang penting telah diidentifikasi tepat, tim pengembang dapat menggunakan sumber daya secara efisien dan menghindari fitur yang tidak sesuai. Selain dari itu, pengecekan fungsionalitas pengguna ini dapat memastikan suatu sistem memenuhi suatu ekspektasi dari pengguna, menghasilkan suatu solusi yang dapat memberikan manfaat dan dapat meningkatkan pengguna.

e. Interaksi Pengguna-Fungsionalitas

Aspek interaksi pengguna-fungsionalitas perlu pengecekan dikarenakan penting untuk memastikan jika alur kerja dan antarmuka pengguna dirancang tepat. Pada proses ini

melibatkan suatu pemeriksaan apakah fungsionalitas yang terdapat pada perangkat lunak mendukung aktivitas yang diharapkan pengguna, apakah interaksi antara pengguna dan sistem dapat dengan mudah dipahami, efisien, dan intuitif. Dengan melakukan pengecekan ini, tim pengembang memastikan jika perangkat lunak memberikan suatu pengalaman yang memuaskan dan juga tidak membingungkan. Pengecekan interaksi pengguna-fungsionalitas juga dapat membantu untuk mengidentifikasi dan mengatasi potensi suatu kesalahan pada interaksi pengguna, sehingga dapat memungkinkan pengguna untuk berinteraksi dengan perangkat lunak dengan lebih produktif dan efektif.

f. *User Stories*

Sangat penting memastikan kebutuhan pengguna telah diungkapkan dan telah dipahami secara akurat oleh tim pengembang melalui pengecekan kebenaran dan kelengkapan terhadap *user stories*. *User stories* harus memiliki suatu kejelasan, serta spesifikasi, dan dapat diukur sehingga menjadi panduan yang efektif dalam pengembangan perangkat lunak. Pengecekan terhadap *user stories* ini memastikan jika setiap aspek kebutuhan dari pengguna telah tercakup pada *user stories*, sehingga tim pengembang fokus terhadap pengembangan fitur yang sesuai dan dapat memberikan suatu nilai tambah. Kemudian, pengecekan *user stories* juga dapat menghindari suatu ambiguitas pada saat memahami kebutuhan dari pengguna, akibatnya dapat meminimalkan suatu kesalahan dan kegagalan pada saat pengembangan perangkat lunak.

g. Kegunaan *User Stories*

Pengecekan kebenaran dan kelengkapan pada aspek kegunaan *user stories* dapat memastikan jika *user stories* relevan terhadap kebutuhan pengguna dan dapat membimbing pengembang mengembangkan perangkat lunak dengan benar. Pada pengecekan ini melibatkan suatu pemeriksaan apakah *user stories* dapat menggambarkan situasi nyata yang akan dihadapi oleh pengguna dan apakah *user stories* dapat memenuhi kebutuhan yang memiliki nilai penting bagi pengguna. Pengembang dapat memastikan untuk fokus terhadap suatu fitur dan fungsionalitas yang penting untuk pengguna, sehingga dapat menghasilkan suatu solusi yang bermanfaat. Pengcekan kebenaran dan kelengkapan pada aspek *user stories* ini juga dapat membantu mencegah suatu penyimpangan kebutuhan pengguna, sehingga perangkat lunak yang dikembangkan dapat memenuhi harapan pengguna dan dapat memberikan suatu nilai yang signifikan.

Suatu sistem dikatakan benar, jika sistem tersebut menghasilkan *output* yang sesuai dengan *input* yang dilakukan dan berakhir dengan sempurna. Sebuah *asersi* diperlukan untuk melakukan pengecekan kebenaran, *asersi* adalah hubungan yang sesuai pada saat program

dieksekusi. *Asersi* yang terletak sebelum pernyataan disebut prekondisi, sedangkan *asersi* yang terletak pada sesudah pernyataan adalah postkondisi.

Pendapat dari Boehm (1984) bahwa kebutuhan dikatakan lengkap apabila semua bagian-bagian hadir dan dari setiap bagian dikembangkan sepenuhnya. Menurut Menzel (2010), kelengkapan yaitu suatu rasa apakah suatu spesifikasi mengandung semua dari kebutuhan sistem dan apakah semua kebutuhan yang ada di dalam spesifikasi sudah ditentukan dengan lengkap. Menurut IEEE 29148, kebutuhan tunggal dapat dikatakan lengkap apabila kebutuhan tersebut dapat diukur dan dapat menjelaskan dari kemampuan dan karakteristik suatu sistem untuk memenuhi kebutuhan dari *stakeholder*. Dokumen SRS yang lengkap tidak ada suatu fungsi yang hilang, yang berarti semua perilaku dari sistem yang tidak diinginkan dan diinginkan sudah ditentukan. Jika terdapat perilaku yang tidak diinginkan, maka jelas itu dilarang masuk ke dalam sistem (Laplante, 2016).

Dari aspek yang telah dijelaskan, berikut Tabel 4.2 adalah tabel pengecekan kebenaran dan kelengkapan.

Tabel 4.2 Pengecekan Kebenaran dan Kelengkapan

Pemeriksaan	Ada/tidak	Keterangan
Definisi masalah	Ada	Belum ada sistem informasi manajemen khusus untuk jalur magang.
Tujuan sistem	Ada	Untuk pengajuan magang di Program Studi Informatika Universitas Islam Indonesia.
Identifikasi pengguna sistem	Ada	Pengguna sistem meliputi mahasiswa, dosen pembimbing, dosen pembimbing akademik,dan program studi.
Fungsionalitas utama	Ada	Sistem informasi yang menyediakan lowongan magang.
Fungsionalitas pendukung	Ada	Pengisian *log activity*, pendaftaran diseminasi,dan pendaftaran sidang pendadaran.
Interaksi pengguna-fungsionalitas	Ada	Interaksi pengisian formulir : ● Mahasiswa mendaftarkan perusahaan ● Mahasiswa mendaftar diseminasi ● Mahasiswa mendaftar sidang
User Stories	Ada	1 *user stories* didapatkan dari proses bisnis
Kegunaan *user stories*	Ada	Terdapat 17 kegunaan *user stories*

Dari Tabel 4.2 disimpulkan bahwa kebutuhan-kebutuhan dari pengembangan Sistem Informasi Jalur Magang telah lengkap. Tidak ada kebutuhan yang tidak diinginkan pada proses pengecekan kebenaaran dan kelengkapan. Dengan melakukan pengcekan kebenaran dan kelengkapan berdasarkan aspek-aspek yang disebutkan, dapat dipastikan bahwa definisi masalah, tujuan sistem, identifikasi pengguna sistem, fungsionalitas utama, fungsionalitas pendukung, interaksi pengguna-fungsionalitas, *user stories*, dan kegunaan *user stories* telah dilakukan pengecekan secara akurat. Hal ini membantu memastikan bahwa sistem yang dikembangkan telah memenuhi persyaratan yang diharapkan dan dapat memberikan suatu solusi yang lengkap dan tepat dengan kebutuhan pengguna.

4.3 Verifikasi *User Stories*

User stories yang telah didapatkan kemudian diverifikasi. Verifikasi dilakukan pada indikator terhadap *user stories* yang telah dikumpulkan dan perancangan antarmuka yang didasarkan pada *user stories*. Dengan melakukan verifikasi, dapat dipastikan bahwa setiap dari *user stories* terverifikasi dengan baik dan sesuai kebutuhan dari pengguna. Selain itu, *user stories* membantu menentukan bagaimana interaksi antar pengguna dengan sistem.

4.3.1 Indikator *User Stories*

Pengecekan *verifiability* perlu dilakukan terhadap *user stories*. Pengecekan ini dilakukan untuk mengetahui bagaimana selesainya *user stories*. Pengecekan *verifiability* adalah proses penting untuk memastikan kepuasan dari kebutuhan yang telah didapatkan. Dalam pengecekan ini, setiap kebutuhan harus dipastikan dapat terpenuhi melalui pengukuran atau metode yang lain namun jelas dan dapat diverifikasi. Kualitas ini sangat penting karena jika kebutuhan tidak dapat dibuktikan, maka artinya kebutuhan tersebut belum terpenuhi. Ketika suatu kebutuhan sulit untuk diukur, maka pemenuhan kebutuhan tersebut menjadi sulit dan sering menimbulkan masalah. Manfaat dari pengecekan *verifiability* yaitu dapat memperoleh bukti secara nyata mengenai pemenuhan suatu kebutuhan. Hal ini memungkinkan untuk mengidentifikasi dan menangani kesenjangan antara kebutuhan yang diharapkan dan memenuhi kebutuhan yang sebenarnya. Selain itu, pengecekan *verifiability* dapat membantu proses perbaikan dan evaluasi yang sistematis. Mengukur suatu kepuasan dari setiap pengguna dengan cara yang jelas dapat mengidentifikasi suatu kekurangan atau ketidaksesuaian pada pemenuhan kebutuhan tersebut, ini memungkinkan untuk mengambil suatu tindakan yang diperlukan yang tujuannya adalah memperbaiki dan meningkatkan kualitas pemenuhan kebutuhan tersebut. Adanya bukti yang dapat diverifikasi, membuat pihak terkait dapat melihat dan memahami dengan jelas sejauh mana kebutuhan sudah terpenuhi (Laplante, 2016). Tabel 4.3 merupakan pengecekan *verifiability*.

Tabel 4.3 Pengecekan *Verifiability*

No	*User stories*	Indikator Berhasil	*Verified*
1.	Sebagai mahasiswa saya ingin mendaftar magang sehingga saya mendapatkan pengalaman magang.	Mahasiswa terdaftar dalam Sistem Informasi Jalur Magang	*Verified*

2.	Sebagai mahasiswa saya ingin melakukan daftar perusahaan tempat magang agar saya dapat magang pada perusahaan yang saya daftarkan	Perusahaan terdaftar dalam sistem informasimagang.	*Verified*
3.	Sebagai mahasiswa saya ingin melihat perusahaan penyedia lowongan yang telah didaftarkan oleh mahasiswa lain sehingga saya bisa langsung daftar magang dan tidak perlu mendaftarkan perusahaan yang sama.	Perusahaan yang telah terdaftar ditampilkan dalam sistem.	*Verified*
4.	Sebagai mahasiswa saya ingin dapat mengisi*log book* agar kegiatan selama magang tercatat. .	Kegiatan magang mahasiswa tercatat pada sistem.	*Verified*
5.	Sebagai mahasiswa saya ingin dapat daftar diseminasi magang sehingga saya bisa melakukan diseminasi magang.	Data mahasiswa yang telah mendaftar diseminasi akan ditampilkan pada halaman diseminasi magang beserta jadwal yang telah ditentukan oleh prodi	*Verified*
6.	Sebagai mahasiswa saya ingin dapat melakukan pendaftaran sidang pendadaran sehingga program studi dapat menjadwalkansidang pendadaran saya.	Data mahasiswa yang telah mendaftar magang akan ditampilkan pada halaman sidang besertajadwal yang ditentukanoleh prodi.	*Verified*
7.	Sebagai mahasiswa saya ingin dapat mengunggah dokumen-dokumen seperti laporan tengah, laporan akhir, dan karya ilmiah sehingga tidak mengunggah dokumenpada website yang berbeda.	Dokumen laporan yangdiunggah akan tersimpan pada database.	*Verified*
8.	Sebagai mahasiswa saya ingin dapat melihatlaporan tengah, laporan akhir, dan karya ilmiah karya dari mahasiswa lain sehingga saya bisa belajar dari laporan-laporan yang ada.	Mahasiswa hanya bisamelihat dokumen yang sudah diunggah. Dokumen tersimpan pada database sistem	*Verified*

		dan bisa dilihat pada tab dokumen.	
9.	Sebagai dosen pembimbing akademik saya ingin dapat memantau kelayakan dari perusahaan yang didaftarkan oleh mahasiswa sehingga saya dapat melakukan suatu persetujuan.	Perusahaan yang tidak disetujui tidak terdaftarpada sistem dan perusahaan yang disetujui akan terdaftarpada sistem.	*Verified*
10.	Sebagai dosen pembimbing saya dapat mengunggah nilai mahasiswa yang melakukan program magang sehingga nilaiyang saya berikan bisa menjadi tolak ukur dari hasil mahasiswa yang menjalani magang.	Nilai akan ditampilkanpada sistem magang.	*Verified*
11.	Sebagai dosen pembimbing saya ingin dapatmelihat laporan-laporan yang sudah diunggah oleh mahasiswa sehingga saya bisabelajar dari laporan-laporan yang ada.	Dokumen laporan yangsudah dikirim akan disimpan pada database sistem.	*Verified*
12.	Sebagai dosen pembimbing saya ingin dapatmemetakan judul-judul dokumen dari waktuyang lama hingga terbaru sehingga dokumennya tertata rapih.	Judul-judul dokumen dikelompokkan berdasarkan tahun pembuatan dokumen.	*Verified*
13	Sebagai dosen pembimbing akademik saya ingin melakukan *approval* permagangan sehingga mahasiswa dapat diproses lebih lanjut oleh program studi.	Mahasiswa yang tidakdisetujui tidak akan lanjut magang dan mahasiswa yang disetujui akan lanjut magang dan data mahasiswa tidak terdaftar pada sistem.	*Verified*
14.	Sebagai Program Studi saya ingin dapat melakukan *approval* terhadap perusahaan magang sehingga perusahaan yang telah didaftarkan oleh mahasiswa bisa terdaftar disistem dan mahasiswa bisa mendaftar magang.	Sebelum perusahaanyang didaftarkan disetujui oleh prodi tidak akan tersedia pada sistem.	*Verified*
15.	Sebagai Program Studi saya ingin dapat menambahkan dosen pembimbing untuk mahasiswa yang melakukan magang, sehingga mahasiswa yang magang tersebutmendapat bimbingan dari dosen pembimbing.	Setiap mahasiswa diberikan dosen pembimbing, satu dosen bisa untuk beberapa mahasiswa.	*Verified*
16.	Sebagai dosen pembimbing atau mahasiswasaya ingin dapat mencatat	Catatan proses bimbingan	*Verified*

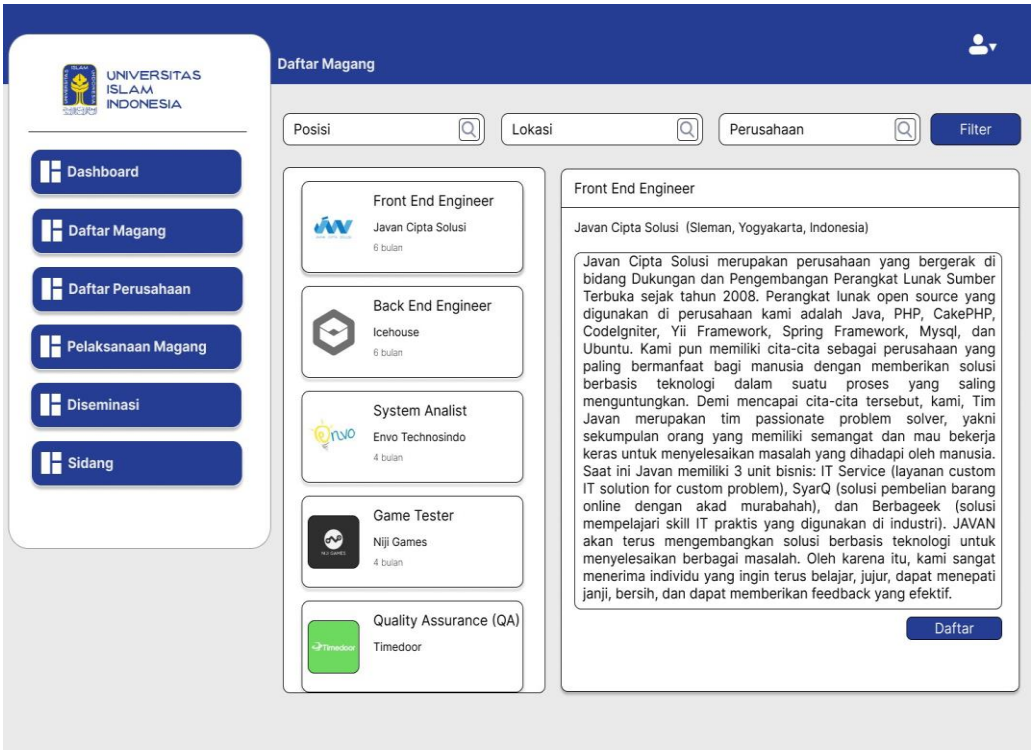
	proses bimbingan kemudian menyimpannya sehingga saya tidak melupakan proses bimbingan	tersimpanpada sistem.	

Dari Tabel 4.3 disimpulkan bahwa setiap *user stories* memiliki indikator keberhasilannya masing-masing. Sebagai contoh, *user stories* nomor 1 yaitu sebagai mahasiswa, saya ingin mendaftar magang sehingga saya bisa mendapatkan pengalaman magang, indikator berhasilnya dari *user stories* tersebut adalah mahasiswa terdaftar dalam Sistem Informasi Jalur Magang telah *verified*. Artinya, indikator keberhasilan tersebut telah memenuhi kebutuhan *user stories* nomor 1.

4.3.2 Rancangan User Stories

Dari 16 *user stories* yang didapat, kemudian dikembangkan menjadi desain. Dari desain yang sudah dibuat, kemudian dijadikan *prototype* menggunakan aplikasi *Figma*. *Prototype* akan diujikan kepada *user* untuk mengetahui bagaimana pendapat tentang *prototype* Sistem Informasi Manajemen Jalur Magang ini.

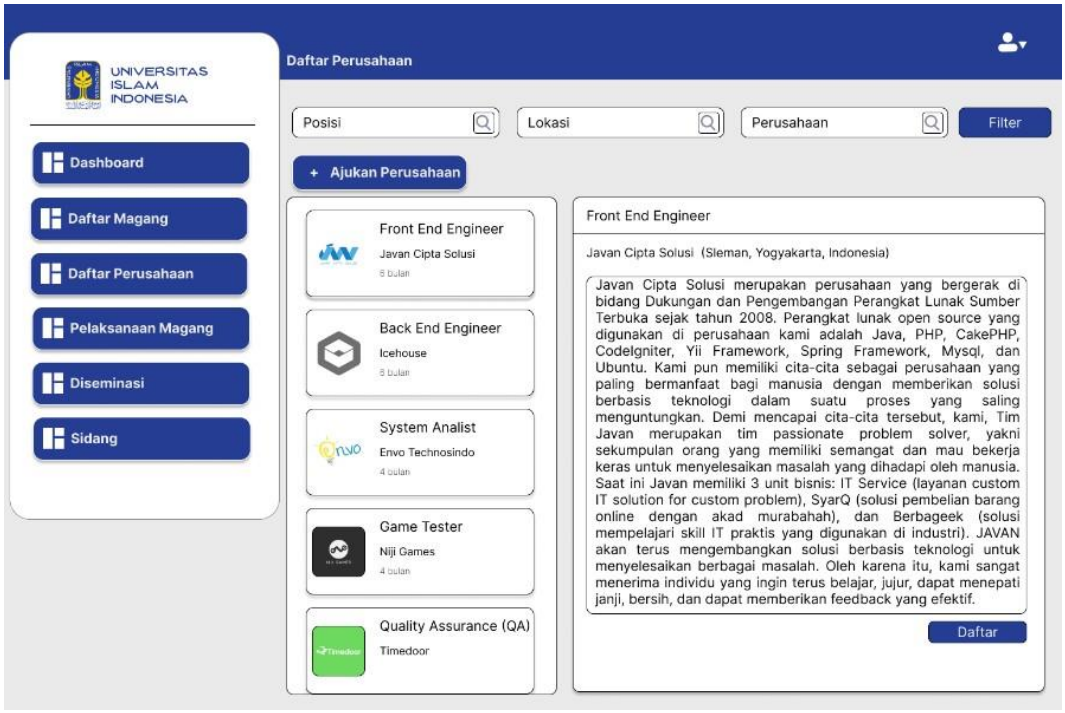
a. *User Stories* Nomor 1 dan 3 (Sebagai mahasiswa saya ingin mendaftar magangsehingga saya bisa mendapatkan pengalaman magang dan sebagai mahasiswa saya ingin melihat perusahaan penyedia lowongan magang yang telah didaftarkan oleh mahasiswa lain sehingga saya bisa langsung daftar magang dan tidak perlu mendaftarkan perusahaan yang sama)



Gambar 4.1 Desain Halaman Daftar Magang

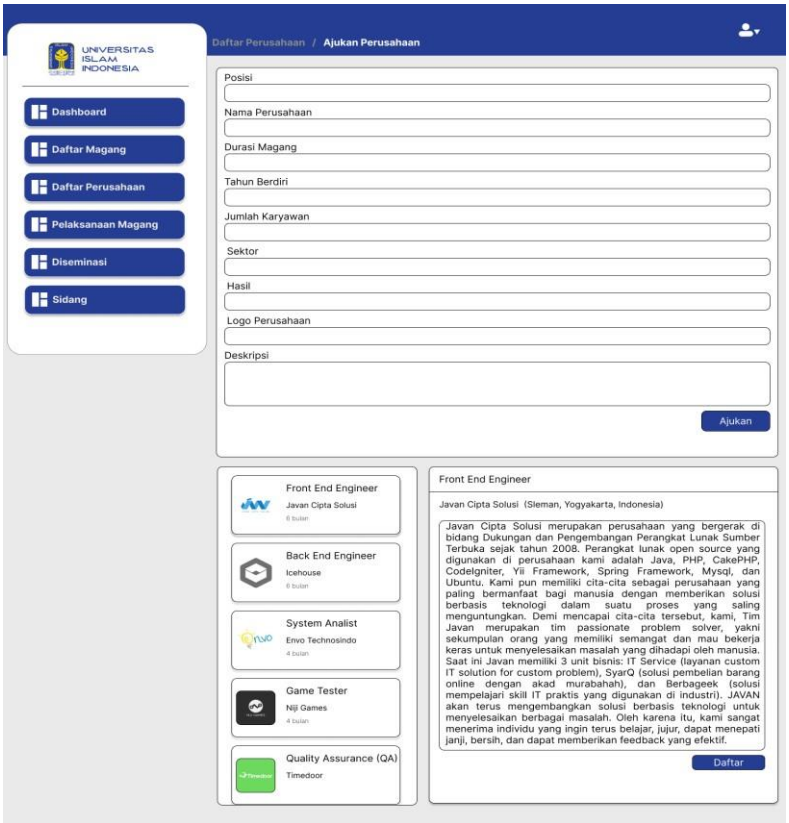
Pada Gambar 4.1 mahasiswa dapat melakukan pendaftaran magang pada perusahaan yang sudah tersedia di sistem. Komponen yang ada pada halaman daftar magang adalah profil perusahaan. Pada halaman daftar magang terdapat *filter* untuk mahasiswa mencari terkait posisi magang, lokasi magang, dan perusahaan. Terdapat *button* untuk mahasiswa mendaftar magang.

b. *User Stories* nomor 2 dan 3 (Sebagai mahasiswa saya ingin melakukan daftar perusahaan tempat magang agar saya dapat magang pada perusahaan yang saya daftarkan dan sebagai mahasiswa saya ingin melihat perusahaan penyedia lowongan magang yang telah didaftarkan oleh mahasiswa lain sehingga saya bisa langsung daftarmagang dan tidak perlu mendaftarkan perusahaan yang sama)



Gambar 4.2 Desain Halaman Daftar Perusahaan

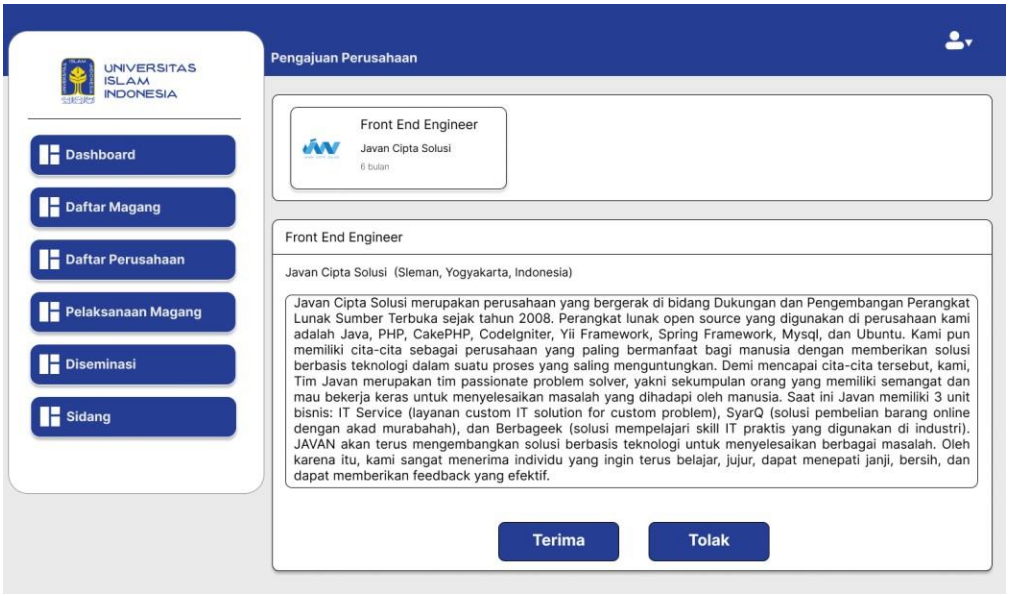
Pada Gambar 4.2, mahasiswa dapat mendaftarkan perusahaan yang belum terdapat di sistem. Komponen yang ada pada halaman daftar perusahaan yaitu terdapat profil perusahaan. Pada halaman daftar magang juga terdapat *button* +Ajukan Perusahaan yang berguna untuk mahasiswa masuk ke halaman ajukan perusahaan dan *button* Daftar untuk mahasiswa mendaftar magang.



Gambar 4.3 Desain Halaman Daftar Perusahaan

Pada Gambar 4.3, mahasiswa dapat mengisikan persyaratan apa saja yang dibutuhkan agar perusahaan penyedia lowongan magang yang akandidaftarkan dapat terdaftar pada sistem. Komponen yang ada di halaman ajukan perusahaan iniadalah *form* untuk pengisian pendaftaran perusahaan dan profil perusahaan. Pada halaman ajukan magang terdapat *button* untuk mendaftarkan perusahaan penyedia lowongan magang dan *button* untuk mahasiswa mendaftar magang.

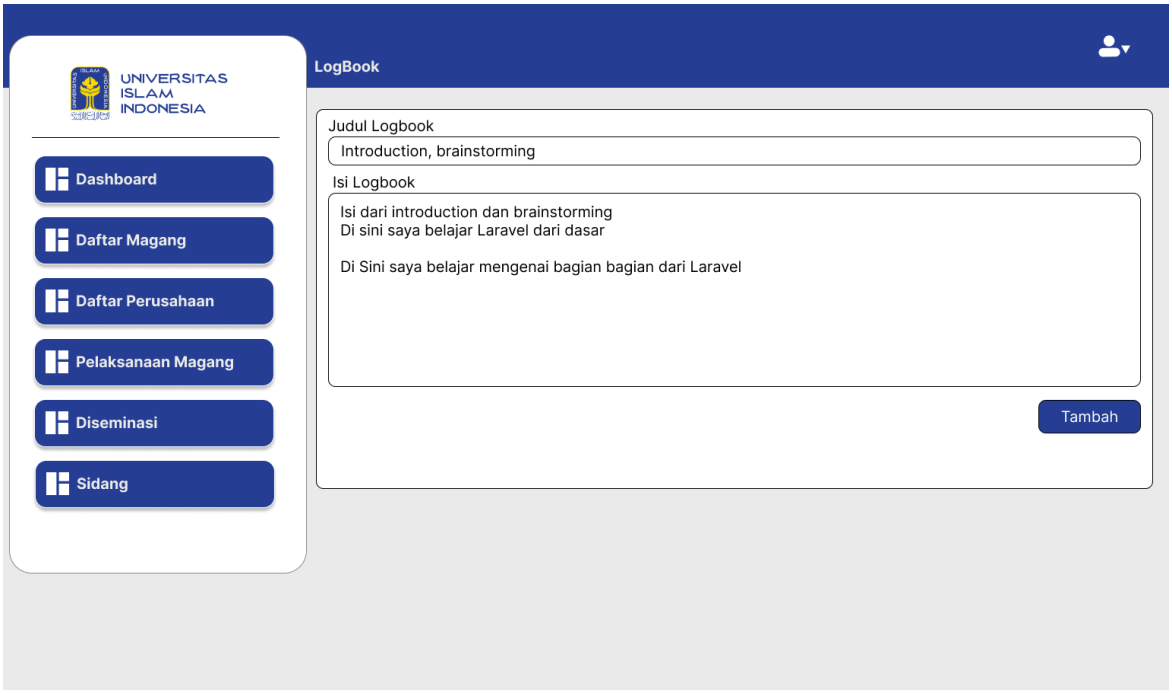
c. *User Stories* nomor 9 dan 15 (Sebagai dosen pembimbing akademik saya ingin dapat memantau kelayakan dari perusahaan yang didaftarkan oleh mahasiswa sehingga saya dapat melakukan persetujuan dan Sebagai program studi saya ingin dapat melakukan *approval* terhadap perusahaan magang sehingga perusahaan yang telah didaftarkan oleh mahasiswa bisa terdaftar di sistem dan mahasiswa bisa mendaftar magang)



Gambar 4.4 Desain Halaman *Approval* Perusahaan

Pada Gambar 4.4, program studi dapat menerima atau menolak perusahaan yang didaftarkan oleh mahasiswa. Komponen yang ada di dalam halaman ini adalah profil perusahaan. Pada halaman *approval* terdapat *button* untuk prodi yangberguna untuk menerima atau menolak perusahaan yang didaftarkan mahasiswa.

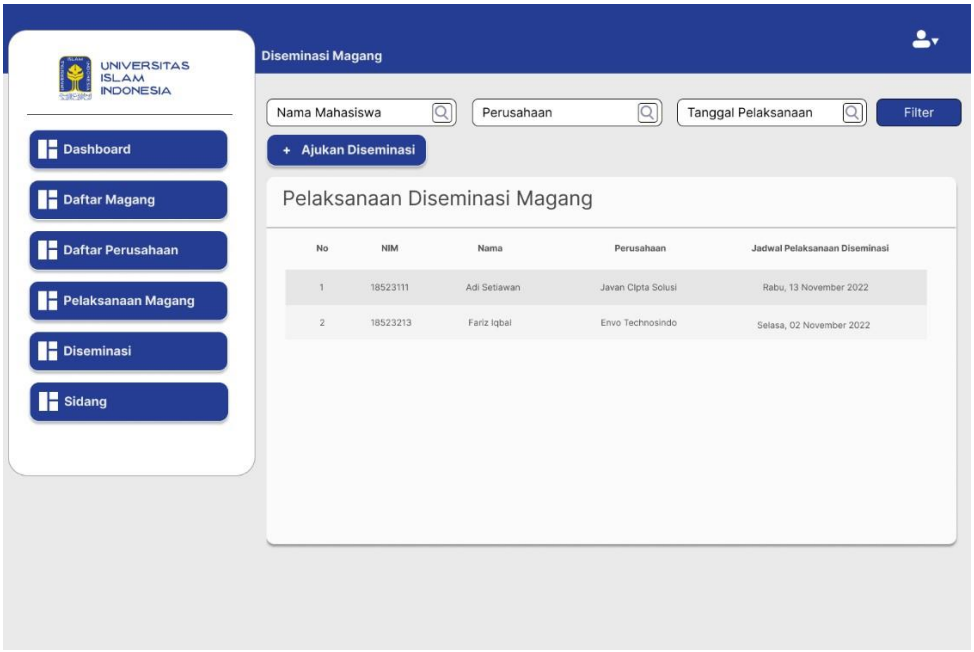
d. *User stories* Nomor 4 dan 17 (Sebagai mahasiswa saya ingin dapat mengisi *log book* agar kegiatan selama magang tercatat dan Sebagai dosen pembimbing atau mahasiswasaya ingin dapat mencatat proses bimbingan kemuudian menyimpannya sehingga sayatidak melupakan proses bimbingan)



Gambar 4.5 Desain Halaman *Logbook* Mahasiswa

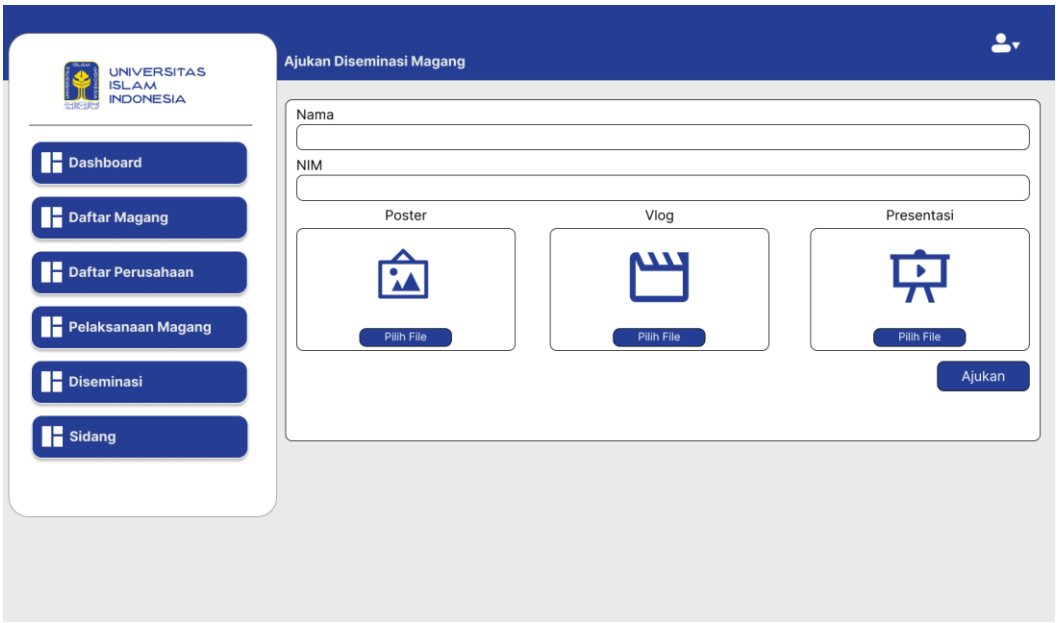
Pada Gambar 4.5, mahasiswa dapat mengisi *log book* yang digunakan untuk mencatat kegiatan magang. Komponen pada halaman ini yaitu berupa *form* untuk mahasiswa mengisi *logbook*. Terdapat *button* Tambah untuk mahasiswa menambahkan *logbook*.

e. *User Stories* nomor 5 (Sebagai mahasiswa saya ingin dapat mendaftar diseminasi magang sehingga saya dapat melakukan diseminasi magang)



Gambar 4.6 Desain Halaman Diseminasi Magang

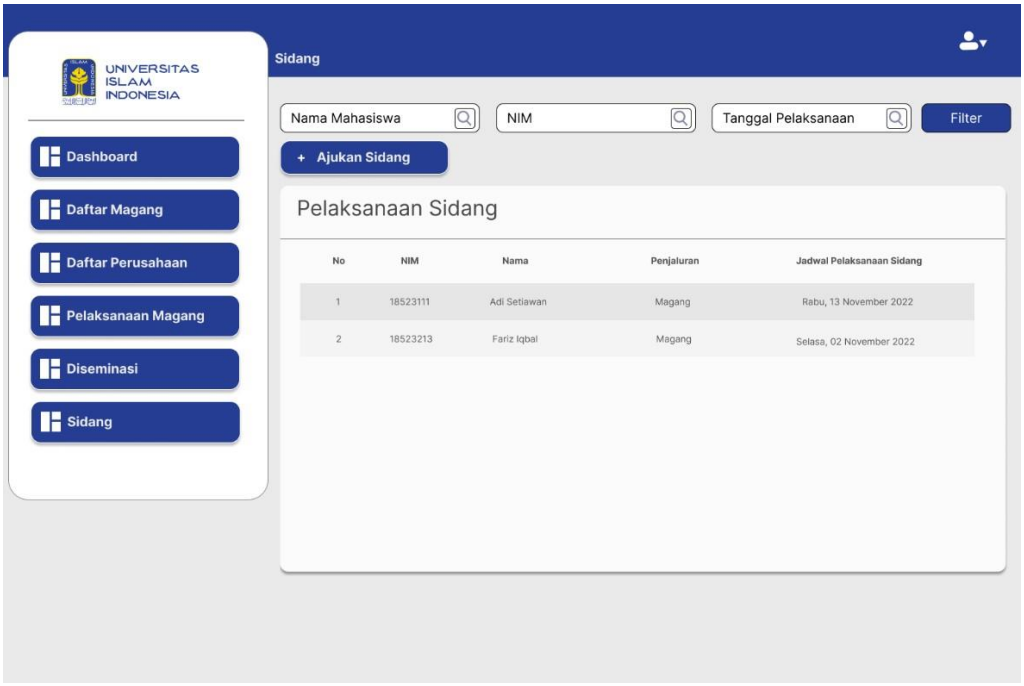
Pada Gambar 4.6, mahasiswa dapat melihat jadwal pelaksanaan diseminasi magang termasuk jadwal diseminasi magang mahasiswa lain. Komponen yang ada pada halaman diseminasi magang adalah tabel jadwal pelaksanaan diseminasi magang. Terdapat *filter* pencarian untuk nama mahasiswa, perusahaan, dan tanggal dari pelaksanaan diseminasi magang. Terdapat *button* +Ajukan Diseminasi yang jika diklik akan ke halaman daftar diseminasi.



Gambar 4.7 Desain Halaman Ajukan Diseminasi Magang

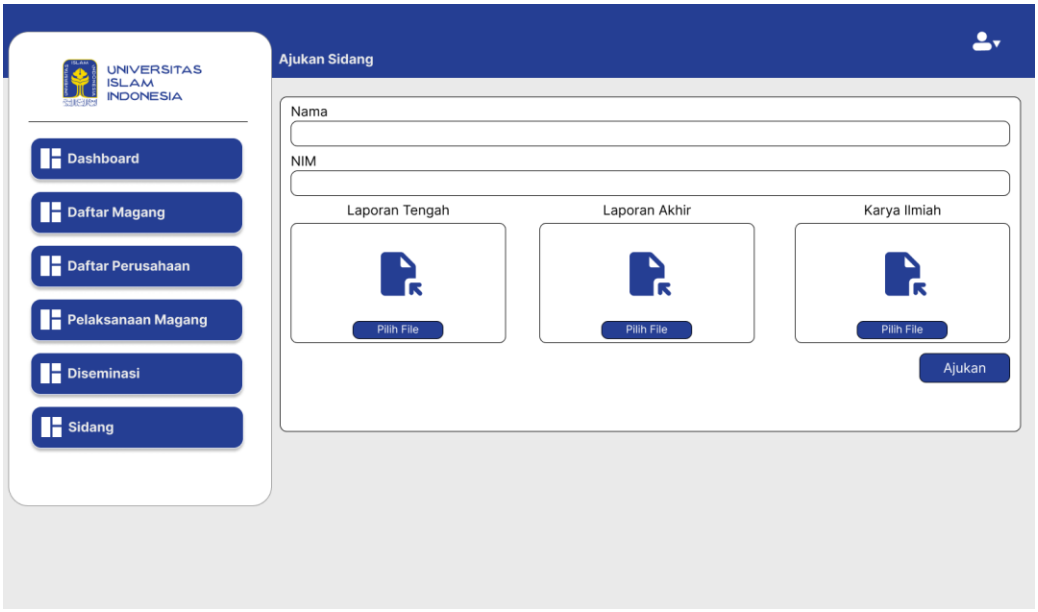
Pada Gambar 4.7, mahasiswa dapat melakukan pendaftaran diseminasi magang. Komponen yang ada pada halaman tersebut adalah *form* untukmahasiswa mendaftar diseminasi magang. Terdapat *button* Ajukan untuk mahasiswa mendaftardiseminasi magang.

f. *User Stories* nomor 6 dan 7 (Sebagai mahasiswa saya ingin dapat melakukan pendaftaran sidang pendadaran sehingga program studi dapat menjadwalkan sidang pendadaran saya dan Sebagai mahasiswa saya ingin dapat mengunggah dokumen- dokumen seperti laporan tengah, laporan akhir, dan karya ilmiah sehingga tidak mengunggah dokumen pada website yang berbeda)



Gambar 4.8 Desain Halaman Ajukan Diseminasi Magang

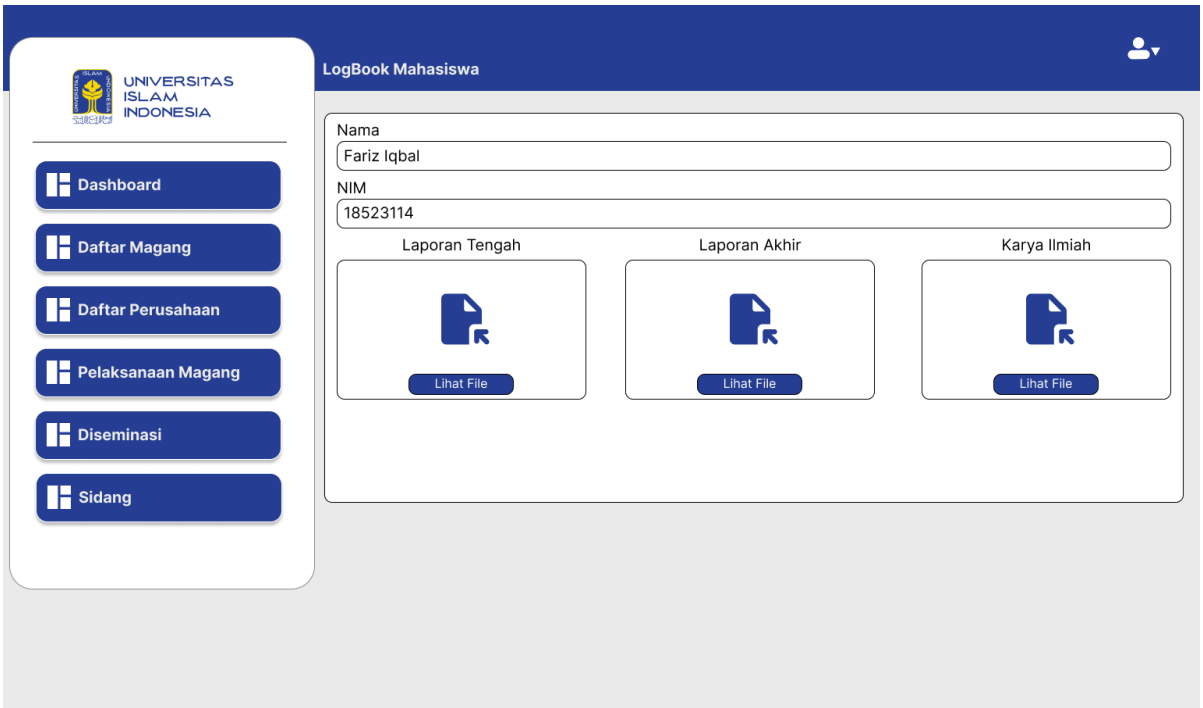
Pada Gambar 4.8, mahasiswa dapat melihat jadwal pelaksanaan sidang mahasiswa termasuk jadwal pelaksanaan sidang mahasiswa lain. Komponen dari halaman ini adalah tabel dari jadwal sidang mahasiswa. Terdapat *filter* untuk pencarian nama mahasiswa, NIM, dan tanggal pelaksanaan sidang. Terdapat juga *button*+Ajukan Magang untuk mahasiswa mendaftar magang.



Gambar 4.9 Desain Halaman Ajukan Sidang

Pada Gambar 4.9, mahasiswa dapat mengajukan sidang dengan mengirim persyaratan untuk mendaftar sidang. Syarat untuk mendaftar sidang yaitu dengan mengirim laporan tengah, laporan akhir, dan karya ilmiah. Komponen dari halaman tersebut adalah *form* untuk pendaftaran sidang. Terdapat *button* Ajukan untuk mahasiswa mendaftar sidang.

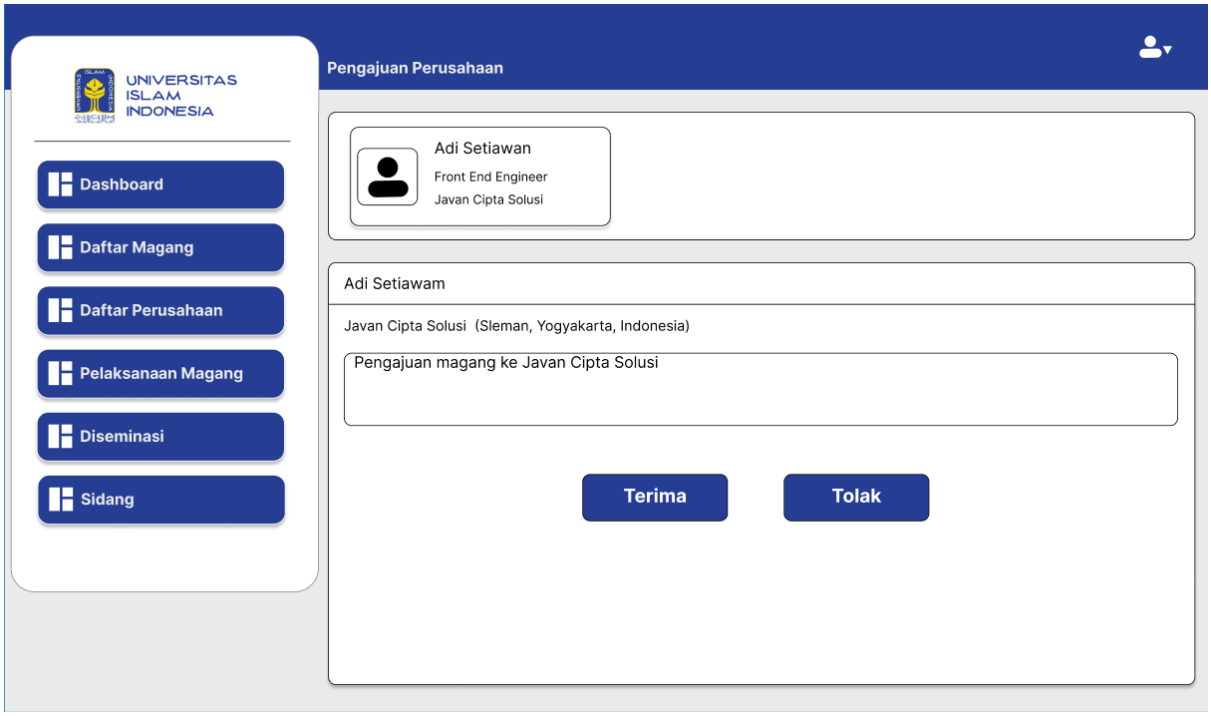
g. *User Stories* nomor 8 (Sebagai mahasiswa saya ingin dapat melihat laporan tengah, laporan akhir, dan karya ilmiah dari mahasiswa lain sehingga saya bisa belajar dari laporan-laporan yang ada)



Gambar 4.10 Desain Halaman *Logbook Mahasiswa*

Pada Gambar 4.10 digunakan mahasiswa untuk melihat karya-karya ilmiah mahasiswa lain. Komponen yang ada pada halaman tersebut adalahtabel nama mahasiswa yang akan dilihat dokumen dan terdapat dokumen-dokumen yang sudahdiunggah oleh mahasiswa lain. Terdapat *filter* yang dapat mencari nama mahasiswa, perusahaan dan status. Dan ada *button logbook* untuk mahasiswa melihat

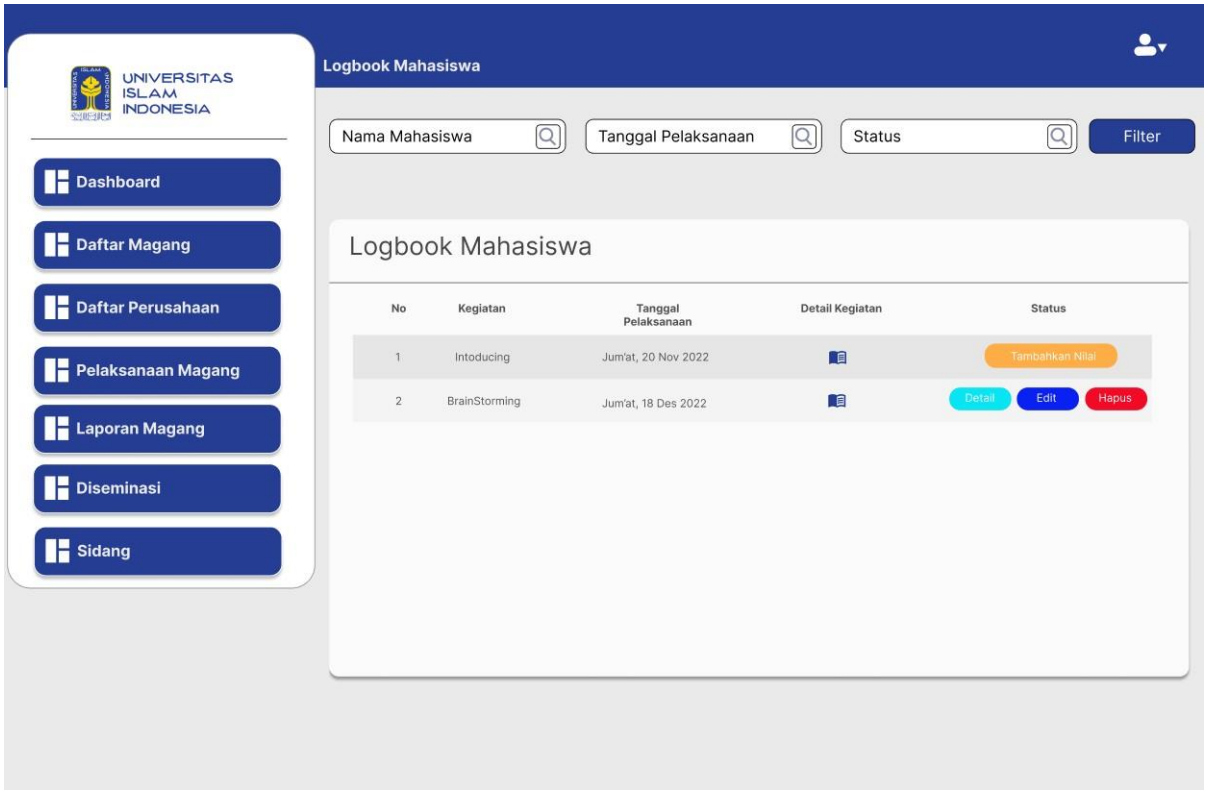
h. *User Stories* nomor 14 (Sebagai dosen pembimbing akademik saya ingin melakukan *approval* permaganan sehingga mahasiswa dapat diproses lebih lanjut oleh program studi)



Gambar 4.11 Desain Halaman Pengajuan Mahasiswa Magang

Pada Gambar 4.11, dosen pembimbing akademik dapat melakukan persetujuan terhadap mahasiswa yang mendaftar magang. Komponen yang terdapat pada halaman tersebut adalah profil mahasiswa yang akan mendaftarmagang. Terdapat *button* untuk dosen pembimbing akademik menerima atau menolak mahasiswa yang mengajukan lowongan magang.

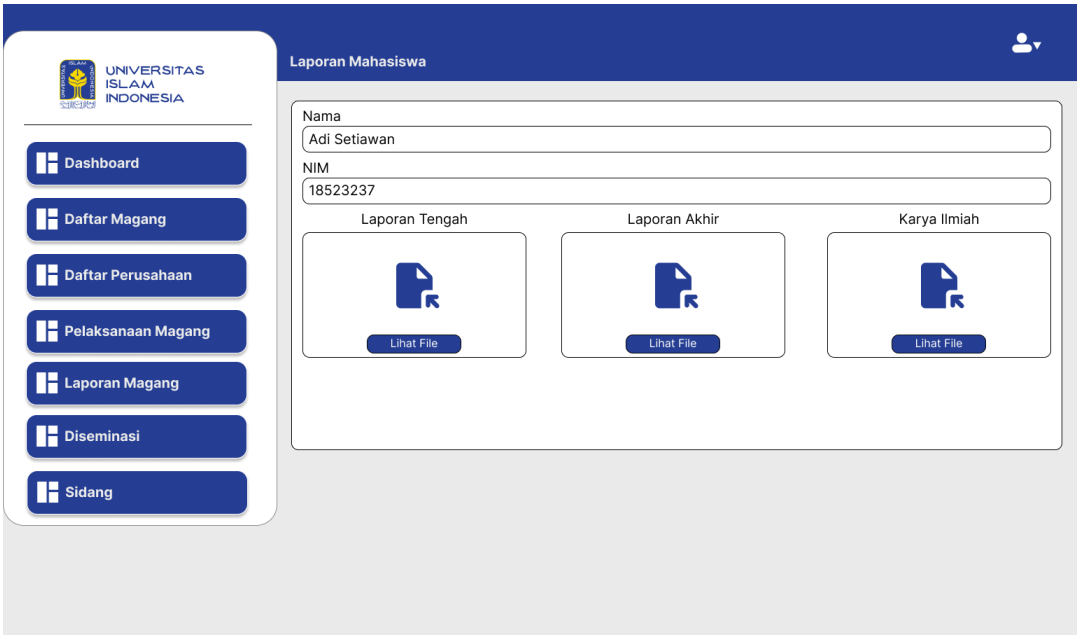
i. *User Stories* nomor 11 (Sebagai dosen saya ingin dapat mengunggah nilai mahasiswa yang melakukan program magang sehingga nilai yang saya berikan menjadi tolak ukurdari hasil mahasiswa yang menjalani magang)



Gambar 4.12 Desain Halaman *Logbook* Mahasiswa pada Dosen

Pada Gambar 4.12, dosen dapat memberikan nilai kepada mahasiswa. Komponen yang terdapat di dalam halaman tersebut adalah daftar kegiatan mahasiswa selama magang. Tedapat *button* Tambahkan Nilai untuk dosen menambahkan nilai, *button* Detail untuk melihat detail dari penilaian tersebu, *button* Edit untuk dosen mengedit penilaian yang diberikan kepada mahasiswa, *button* Hapus untuk dosenmenghapus nilai.

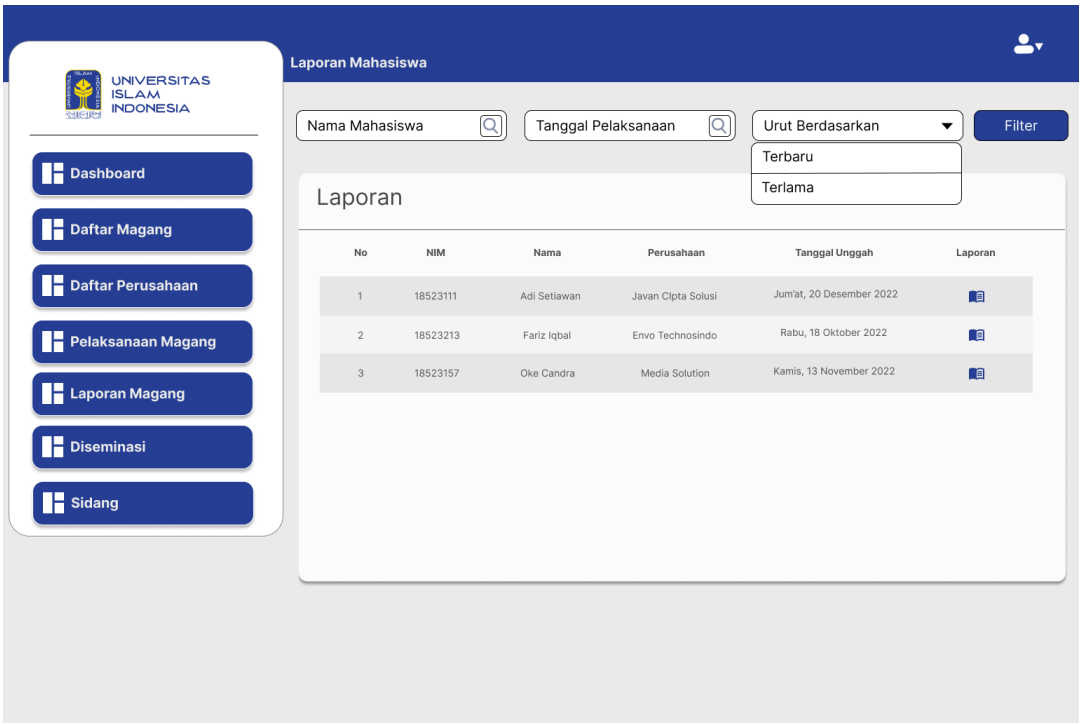
j. *User Stories* nomor 12 (Sebagai dosen pembimbing saya ingin dapat melihat laporan-laporan yang sudah diunggah oleh mahasiswa sehingga saya bisa belajar dari laporan-laporan yang ada)



Gambar 4.13 Desain Halaman Laporan Mahasiswa

Pada Gambar 4.13, dosen dapat melihat dokumen- dokumen yang sudah diunggah oleh mahasiswa. Komponen yang ada di dalam halaman ini adalah profil mahasiswa dan dokumen dokumen yang sudah diunggah oleh mahasiswa.

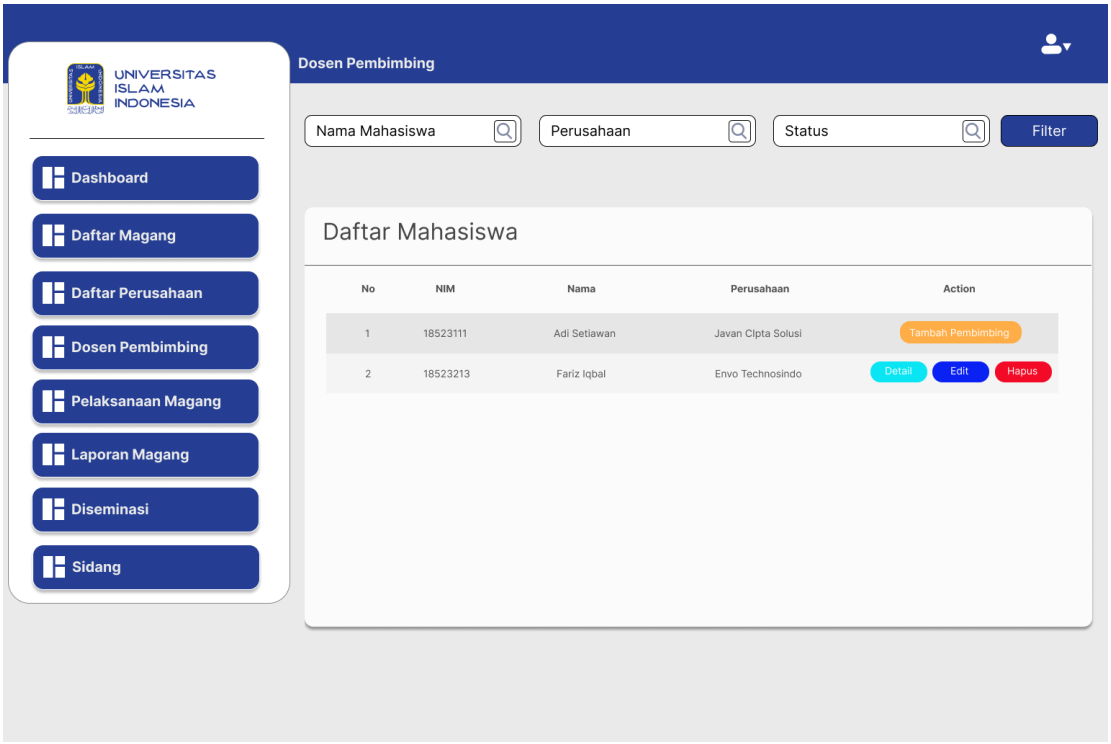
k. *User Stories* nomor 13 (Sebagai dosen pembimbing saya ingin dapat memetakan judul-judul dokumen dari waktu yang lama hingga terbaru sehingga dokumennya rapih)



Gambar 4.14 Desain Halaman Laporan Mahasiswa

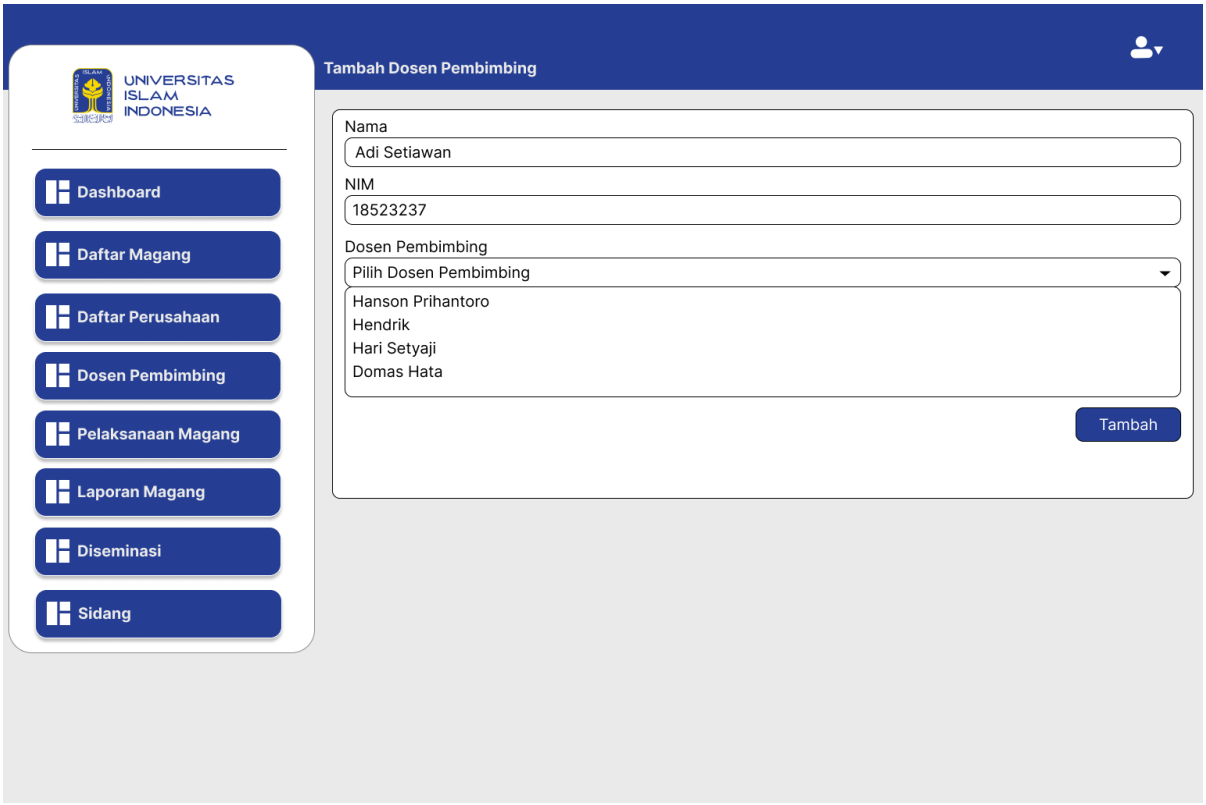
Pada Gambar 4.14, dosen dapat mengurutkan judul dari yang terlama hingga judul terbaru. Komponen yang ada pada halaman ini adalah daftar nama mahasiswa yang sudah mengunggah dokumennya di tanggal tertentu. Terdapat *filter* untuk mencari nama mahasiswa, tanggal pelaksanaan mahasiswa mengunggah dokumen, dan *filter* untuk mengurutkan tanggal pada saat mahasiswa mengunggah dokumen.

l. *User Stories* nomor 16 (Sebagai program studi saya ingin dapat menambahkan dosen pembimbing untuk mahasiswa yang melakukan magang, sehingga mahasiswa yang magang tersebut mendapat bimbingan dari dosen pembimbing)



Gambar 4.15 Desain Halaman Dosen Pembimbing

Pada Gambar 4.15, prodi dapat menambahkan dosen pembimbing kepada mahasiswa. Komponen yang ada pada halaman tersebut adalah tabel yang berisi mahasiswa-mahasiswa yang mendaftar magang. Terdapat *button* untukTambah Pembimbing, Detail, Edit, dan Hapus)



Gambar 4.16 Desain Halaman Tambah Dosen Pembimbing

Pada Gambar 4.16, terusan dari Gambar 4.15 Desain Halaman Dosen Pembimbing ketika prodi mengklik *button* Detail. Komponen yang ada pada halaman ini adalah *form* untuk prodi menambahkan dosen pembimbing kepada mahasiswa. Terdapat *button* Tambah untuk menambahkan dosen pembimbing kepada mahasiswa.

Pengujian dilakukan oleh beberapa calon *user*. Sampel calon *user* tersebut terdiri dari 2 dosen Informatika Universitas Islam Indonesia dan 7 mahasiswa Informatika yang telah melakukan magang. *User* melakukan pengujian terhadap setiap *user stories* yang berada pada Tabel 4.3 Pengecekan *Verifiability*. Pada tahap ini, *user* menjalankan *prototype* sesuai dengan alur bisnis yang telah disampaikan. *User* mengecek setiap fungsi yang terdapat pada *prototype*. Setelah selesai pengecekan, *user* melakukan penilaian terhadap *prototype* yang telah dijalankan dan memberikan kritik dan saran. Hasil dari pengujian adalah bahwa setiap *user stories* yang dihasilkan telah diverifikasi benar.

4.4 Diskusi

Dari langkah-langkah yang telah dilakukan pada rekayasa kebutuhan perangkat lunak telah berhasil didapatkan *user stories* yang kemudian *user stories* tersebut divalidasi, dicek, dan diverifikasi oleh calon pengguna. *User stories* penting dalam pengembangan sistem informasi dikarenakan *user stories* menjadi pemandu untuk para pengembang dalam mengembangkan perangkat lunak terutama pada pengembangan perangkat lunak yang menggunakan metodologi *agile*.

4.4.1 Validasi, Pengecekkan, dan Verifikasi *User Stories*

Pengujian sistem telah dilakukan oleh calon pengguna untuk memvalidasi *user stories* yang telah dibuat. Pengguna melakukan pengujian dengan cara memeriksa setiap dari *user stories* menggunakan *prototype* aplikasi sebagai alat bantu. Pengecekan *user stories* dilakukan dengan langkah-langkah tertentu. Langkah pertama yang dilakukan calon pengguna adalah pengguna diberikan penjelasan mengenai proses bisnis dari Sistem Informasi Jalur Magang yang didapatkan dari analisis kebutuhan. Selanjutnya, pengguna dibimbing untuk menjalankan *prototype* Sistem Informasi Jalur Magang oleh pengembang. Kemudian, pengguna secara langsung menjalankan *prototype* tersebut.

Dari hasil pengujian, ditemukan bahwa *user stories* yang dihasilkan telah terverifikasi dengan benar. Pengguna dapat memastikan bahwa setiap dari fitur dan fungsionalitas yang dijelaskan dalam *user stories* dapat berjalan dengan baik pada *prototype* aplikasi. Hal ini menunjukkan bahwa *user stories* yang didapatkan mendapat validasi dari pengujian yang dilakukan oleh calon pengguna. Dengan demikian, kebutuhan pengguna yang diungkapkan melalui *user stories* telah terverifikasi dan memastikan bahwa pengembangan perangkat lunak dapat sesuai dengan harapan pengguna.

Pengujian *user stories* dengan menggunakan *prototype* aplikasi memberikan suatu keuntungan dalam memvalidasi kebenaran dan kelayakan *user stories*. Secara langsung, pengguna dapat melihat dan merasakan bagaimana sistem yang dikembangkan akan berfungsi sesuai dengan kebutuhan pengguna. Selain itu, pengujian juga dapat memberikan kesempatan pengguna untuk memberikan suatu *feedback* yang berharga pada tim pengembang, sehingga perbaikan dan penyempurnaan lebih lanjut pada *user stories* dan pengembangan perangkat lunak.

4.4.2 Peran *User Stories* dalam Pengembangan Sistem Informasi

User stories memiliki peran yang penting untuk pengembang dalam memandu pengembangan perangkat lunak, dimulai dari pemahaman bagaimana mendapatkan suatu kebutuhan pengguna. *User stories* merupakan sebuah narasi yang menggambarkan kebutuhan fungsional sistem dari sudut pandang pengguna. Setiap dari *user* stories dapat mewakili satu atau beberapa *task* yang akan dikerjakan oleh pengembang.

Pada proses pengembangan, peran penting *user stories* diperlukan untuk memandu dalam merancang, melakukan pemrograman, dan mengimplementasikan setiap fitur yang diinginkan oleh pengguna. *User stories* membantu dalam mengubah kebutuhan pengguna menjadi *task* yang dapat dipecahkan oleh tim pengembang. *User stories* dapat membantu memberikan penjelasan mengenai harapan apa yang pengguna harapkan untuk sistem yang sedang dikembangkan. Dengan *user stories* sebagai panduan, tim pengembang mendapat pemahaman yang jelas tentang *task* yang dilakukan dan dapat fokus pada pengembangan fitur yang memiliki nilai paling penting bagi pengguna. Selain itu, *user stories* memiliki peran yang penting sebagai suatu alat validasi agar memastikan sistem yang dikembangkan oleh pengembang telah memenuhi kebutuhan pengguna yang ada pada *user stories*. Pada setiap tahap pengembangan, *user stories* dapat digunakan sebagai suatu referensi untuk verifikasi apakah sistem yang dikembangkan sudah sesuai dengan kebutuhan yang ditetapkan sebelumnya oleh pengguna. Ini artinya, *user stories* dapat membantu memastikan pengembangan sistem yang dilakukan telah sesuai harapan dari pengguna.

Pengembang memiliki suatu pedoman jelas dan terstruktur dalam melaksanakan *task* pengembangan dengan adanya *user stories*. Ini juga dapat membantu tim pengembang untuk diarahkan pada jalur yang benar dan memastikan fokus pengembangan tetap pada kebutuhan pengguna yang sudah diidentifikasi sebelumnya. *User stories* membantu menjaga suatu kejelasan dan kesesuaian pengembangan kebutuhan pengguna yang telah ditetapkan. Kemudian, *user stories* memiliki peran untuk memfasilitasi komunikasi antara pengembang dan para *stakeholder*. *User* stories juga dapat digunakan untuk diskusi, negosiasi, dan memperjelas kebutuhan dari pengguna, semua pihak yang terlibat pada pengembangan perangkat lunak memiliki pemahaman yang sama tentang kebutuhan yang harus dipenuhi. Ini memungkinkan sebuah kesepahaman yang baik tercipta antara tim pengembang dan pemangku kepentingan, sehingga pengembangan perangkat lunak dapat berjalan lebih efisien.

4.4.3 Peran *User Stories* dalam Metodologi *Agile*

Dalam praktik yang ada pada *agile*, peran penting *user stories* juga diperlukan dalam pengembangan perangkat lunak secara berulang dan berkembang sedikit demi sedikit secara teratur. *User stories* membantu dalam pengidentifikasian fitur-fitur yang nilainya paling tinggi dan merancang iterasi pengembangan yang fokusnya pada hasil nilai bisnis yang nyata pada setiap siklus pengembangan perangkat lunak. Dengan *user stories*, tim pengembang dapat mengatur prioritas *task* yang telah dibagi menjadi sprint di dalam pengembangan perangkat lunak yang menggunakan metodologi *agile* berdasarkan kepentingan dan kebutuhan dari pengguna, sehingga hasil pada setiap iterasi dapat memberikan suatu manfaat nyata untuk pengguna.

Secara keseluruhan, *user stories* memiliki sebuah peran yang penting pada pengembangan perangkat lunak dalam metodologi *agile*. *User stories* tidak hanya membantu dalam mendapatkan suatu pemahaman yang lebih baik tentang kebutuhan dari pengguna, tetapi juga dapat menjadi sebuah alat bantu dan alat validasi pada setiap tahap pengembangan. Adanya *user stories*, pengembangan perangkat lunak yang dikerjakan dapat secara terstruktur, fokus, dan sesuai dengan kebutuhan pengguna yang sesungguhnya.

BAB V
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Dari penelitian yang dilakukan, telah berhasil didapatkan 16 *user stories* yang diperoleh dari tahap analisis kebutuhan perangkat lunak untuk Sistem Informasi Jalur Magang. Proses pengumpulan *user stories* ini dimulai dengan langkah pertama yaitu studi kelayakan. Dari studi kelayakan diperoleh suatu pemahaman tentang alur dari proses bisnis Sistem Informasi Jalur Magang. Setelah proses bisnis dipahami, selanjutnya dilakukan elisitasi dan analisis kebutuhan untuk mendapatkan *user stories* yang diperlukan pada pengembangan Sistem Informasi Jalur Magang. Tujuan dari dari proses elisitasi dan analisis kebutuhan adalah untuk mengidentifikasi kebutuhan dari pengguna secara spesifik dan prioritas pengguna. Hasil dari tahap elisitasi dan analisis kebutuhan ini adalah 16 *user stories* yang menggambarkan fitur-fitur yang dibutuhkan pengguna pada Sistem Informasi Jalur Magang.

Selanjutnya, *user stories* yang telah teridentifikasi digunakan sebagai dasar untuk merancang suatu spesifikasi kebutuhan yang lebih terperinci. Pada proses ini, melibatkan identifikasi suatu *task* yang akan dilakukan oleh pengembang untuk memenuhi *user story* dalam mengembangkan Sistem Informasi Jalur Magang. Dari spesifikasi kebutuhan yang rinci, dapat menjadi panduan dalam pengembangan perangkat lunak.

Pada tahap terakhir analisis kebutuhan dilakukan validasi kebutuhan dengan menyesuakan proses bisnis yang berada pada Sistem Informasi Jalur Magang. Proses validasi kebutuhan melibatkan verifikasi apakah *user stories* yang telah teridentifikasi, dapat memenuhi persyaratan dan tujuan dari perangkat lunak yang ditetapkan sebelumnya. Penting untuk melakukan validasi kebutuhan agar memastikan perangkat lunak yang dikembangkan sesuai pada kebutuhan bisnis yang ada.

User stories yang telah teridentifikasi kemudian dikembangakan sebuah *prototype* dari Sistem Informasi Jalur Magang. Dari *prototype* ini dapat memungkinkan pengguna untuk melakukan pengujian dan dapat memberikan suatu umpan balik terhadap fungsionalitas yang telah diimplementasikan. Pengguna dilibatkan pada proses verifikasi untuk memastikan antara *user stories* yang diimplementasikan ke perangkat lunak telah sesuai.

5.2 Saran

Dalam pengembangan Sistem Informasi Jalur magang, telah dilakukan analisis kebutuhan hingga pengembangan *prototype* sistem sebagai alat validasinya. Sebagai pekerjaan ke depan, perlu dilakukan pengembangan sistem informasi manajemen jalur magang berbasis *web* agar dapat diakses oleh mahasiswa kapanpun dan di manapun. Sebagai saran penelitian lanjutan, pekerjaan dapat dibandingkan metode analisis kebutuhan dari sistem yang sudah berjalan seperti menggunakan analisis kebutuhan menggunakan *PIECES*. Saran selanjutnya adalah mengkaji *user stories* yang melibatkan proses bisnis dan *user stories* yang tidak melibatkan proses bisnis.

DAFTAR PUSTAKA

Adiguna, A. R., Saputra Chandra, M., & Pradana, F. (2018). Analisis dan Perancangan Sistem Informasi Manajemen Gudang pada PT Mitra Pinasthika Mulia Surabaya. *Jurnal Pengembangan Teknologi Informasi Dan Ilmu Komputer*, *2*(2), 612–621.

Adikara, F., Gunawan, H., & Sandfreni, S. (2018). Pemodelan Hasil Elisitasi Kebutuhan Sistem Penjualan Online Menggunakan Metode Knowledge Acquisition in Automated Specification. *Jurnal Edukasi Dan Penelitian Informatika (JEPIN)*, *4*(2), 108. https://doi.org/10.26418/jp.v4i2.28016

Amelia, H., Irmanda, H. N., Pembangunan, U., Veteran, N., Labu, P., & Selatan, J. (2021). *SISTEM INFORMASI MAGANG PADA UPT TEKNOLOGI INFORMASI DAN KOMUNIKASI UNIVERSITAS PEMBANGUNAN NASIONAL VETERAN JAKARTA 1 Pendahuluan 2 Tinjauan Pustaka 3 Metode Penelitian*. *4221*, 154–163.

Anggraini, R. (2021). Rancang Bangun Sistem Informasi Administrasi Pengelolaan Dana Masjid Berbasis Web (Studi Kasus: Masjid Al-Muttaqin). *Jurnal Teknologi Dan Sistem Informasi (JTSI)*, *2*(3), 109–118. http://jim.teknokrat.ac.id/index.php/JTSI

Arif Setia Sandi A., Soedijono, B., & Nasiri, A. (2020). Pengaruh Perceived Usefulness dan Perceived Ease of Use Terhadap Attitude Toward Using Dengan Metode TAM Pada Sistem Informasi Magang Kerja. *IT Journal Research and Development*, *5*(2), 109–118. https://doi.org/10.25299/itjrd.2021.vol5(2).5287

Baharuddin, M. R. (2021). *Pengembangan Sistem Informasi Manajemen Pelaksanaan Magang FKIP UNCP*. *1*(1), 34–41.

Bingamawa, M. T., & Ahmad, S. (2016). A Review on Requirement Engineering Process in Game Development. *International Journal of Scientific Research in Computer Science, Engineering and Information Technology © 2017 IJSRCSEIT*, *3*(13). https://doi.org/10.13140/RG.2.2.32494.00329

Certuche. (2016). *Agile: The challenge of writing good enough User Stories?* LinkedIn. https://www.linkedin.com/pulse/agile-challenge-writing-good-enough-user-stories-victor-hugo

Cohn, M. (2009). *User Stories Applied For Agile Software Development*.

Ependi, U. (2017). Pengembangan E-Trace Alumni Dengan Menggunakan Pendekatan Metode Agile. *Seminar Nasional Informatika (SEMNASIF)*, *1*(4), 239.

http://tracer.tp.ugm.ac.id/new/content/apa-itu-tracer-study%0Ahttp://jurnal.upnyk.ac.id/index.php/semnasif/article/view/1104

Goodpasture, J. C. (2016). Project Management the Agile way: Making it Work in the Enterprise. In *Management*.

Harahap, S. (2018). STUDI KELAYAKAN BISNIS Pendekatan Integratif. In *FEBI UIN-SU Press*.

Indra Kharisma Raharjana. (2017). *PENGEMBANGAN SISTEM INFORMASI MENGGUNAKAN METODOLOGI AGILE.pdf* (p. 80). Grup Penerbitan CV BUDI UTAMA.

Iqbal, M. F. (2021). *Penerapan Simple Agile Methodology dalam Pengembangan Aplikasi Web*.

Islahuddin, B. N., Wicaksono, S. A., & Purnomo, W. (2020). Pengembangan Sistem Informasi Magang untuk Membantu Proses Administrasi Siswa Magang (Studi pada : Badan Kepegawaian Negara). *Jurnal Pengembangan Teknologi Informasi Dan Ilmu Komputer*, *4*(5), 1480–1489.

Ismail, I., Hasan, H., & Musdalifah, M. (2018). Pengembangan Kompetensi Mahasiswa Melalui Efektivitas Program Magang Kependidikan. *Edumaspul - Jurnal Pendidikan*, *2*(1), 124–132. https://doi.org/10.33487/edumaspul.v2i1.48

Komalasari, D., Salsabilah, A., Studi, P., Informatika, M., Vokasi, F., Darma, U. B., Studi, P., Akuntansi, K., Vokasi, F., Darma, U. B., Studi, P., Informatika, M., Vokasi, F., & Darma, U. B. (2022). *Sistem Informasi Pendaftaran Magang Pada*. *08*(01), 44–51.

Laplante, P. A. (2016). Requirements Engineering for Software and Systems. In *Https://Medium.Com/* (3rd ed.). CRC Press. https://medium.com/@arifwicaksanaa/pengertian-use-case-a7e576e1b6bf

Lau. (2010). Storytelling for User experience: Crafting Stories for Better Design. *Media*, 539.

Lestari, A., & Novita, M. (2019). SISTEM INFORMASI MAGANG BERBASIS WEBSITE PADA DINAS KESEHATAN PROVINSI JAWA TENGAH. *Seminar Nasional Science and Engineering National Seminar*, *1*(1).

Lewenusa, I. (2017). Rekayasa Kebutuhan Perangkat Lunak Pada Perusahaan Skala Kecil Dan Menengah Dengan Pendekatan Soft System Methodology (Ssm) – Studi Kasus Pt Xyz. *Computatio : Journal of Computer Science and Information Systems*, *1*(1), 49. https://doi.org/10.24912/computatio.v1i1.240

Lutfiani, N., Harahap, P., Aini, Q., Dimas, A., Ahmad, A. R., & Rahardja, U. (2020). InfoTekJar : Jurnal Nasional Informatika dan Teknologi Jaringan Attribution-

NonCommercial 4.0 International. Some rights reserved Inovasi Manajemen Proyek I-Learning Menggunakan Metode Agile Scrumban. *InfoTekJar : Jurnal Nasional Informatika Dan Teknologi Jaringan*, *5*(1), 96–101. https://doi.org/10.30743/infotekjar.v5i1.2848

Mäkitalo, N., Taivalsaari, A., Kiviluoto, A., Mikkonen, T., & Capilla, R. (2020). On opportunistic software reuse. *Computing*, *102*(11), 2385–2408. https://doi.org/10.1007/s00607-020-00833-6

Messer Hugo. (2021). Start Agile. In *วารสารวิชาการมหาวิทยาลัยอีสเทิร์นเอเชีย* (Vol. 4, Issue 1).

Mountain Goat Software. (2014). Planning Poker: An Agile Estimating and Planning Technique. *Mountain Goat Software*, 3. https://www.mountaingoatsoftware.com/agile/planning-poker

OKI. (2022). Agile Software Development untuk Meningkatkan Kinerja Perusahaan | Ekipa Indonesia. In *Ekipedia.co.id*. https://www.ekipa.co.id/agile-software-development-untuk-kinerja-perusahaan/

Pratidana, D. (2017). Hak cipta dan penggunaan kembali: Lisensi ini mengizinkan setiap orang untuk menggubah, memperbaiki, dan membuat ciptaan turunan bukan untuk kepentingan komersial, selama anda mencantumkan nama penulis dan melisensikan ciptaan turunan dengan syarat ya. *J. Exp. Psychol. Gen*, *136*(1), 23–42.

Priskila, R. (2018). Pada Perusahaan Karya Cipta Buana Sentosa. *Journal of Computer Engineering System and Science*, *3*(2), 94–99.

Ramadhan, D. W. (2019). PENGUJIAN USABILITY WEBSITE TIME EXCELINDO MENGGUNAKAN SYSTEM USABILITY SCALE (SUS) (sTUDI KASUS: WEBSITE TIME EXCELINDO). *JIPI (Jurnal Ilmiah Penelitian Dan Pembelajaran Informatika)*, *4*(2), 139. https://doi.org/10.29100/jipi.v4i2.977

Ridwan Mohamad et al. (2019). *Sistem Informasi Manajemen*. https://doi.org/10.31227/osf.io/cfy76

RIZALDI, M. O. H. D. W. I. C. (2021). *Pengembangan Sistem Informasi Jadwal Kuliah Untuk Fakultas Kedokteran Universitas Islam Indonesia*.

uBugtrack. (2023). *Simple Agile Methodology S.A.M. Programming*. UBugtrack. https://ubugtrack.com/?page=simple_agile_methodology

Yulianto, Hery Dwi, and R. B. F. (2018). IJIS Indonesian Journal on Information System ISSN 2548-6438. *IJIS-Indonesia Journal on Information System*, *3*(April), 11. https://media.neliti.com/media/publications/260171-sistem-informasi-pengolahan-data-

pembeli-e5ea5a2b.pdf

LAMPIRAN