

BUKU SDP

Projek program LMS (Learning Management System)

PT.ORELA SHIPYARD



Oleh:

Reynard Kamadjaja 2221154

Richard Hadiyanto 2221155

Yosua Christian 222117069

PROGRAM SARJANA

PROGRAM STUDI INFORMATIKA

FAKULTAS SAINS DAN TEKNOLOGI

INSTITUT SAINS DAN TEKNOLOGI TERPADU SURABAYA

SURABAYA

2025

ABSTRACT

Di era digital yang semakin maju, Learning Management System (LMS) telah menjadi alat penting dalam mendukung proses pendidikan dan pelatihan daring, baik di lingkungan akademik maupun profesional. Namun, banyak perusahaan yang masih bergantung pada LMS pihak ketiga, yang seringkali menimbulkan sejumlah tantangan. Tantangan ini meliputi keterbatasan fleksibilitas dalam penyesuaian fitur, biaya langganan yang tinggi, serta risiko terkait keamanan data dan privasi pengguna. Ketergantungan pada solusi eksternal juga sering membatasi kemampuan perusahaan untuk mengadaptasi sistem pembelajaran sesuai dengan kebutuhan unik organisasi mereka.

Menghadapi kendala tersebut, PT Orela Shipyard memutuskan untuk mengembangkan sistem LMS internal yang independen sebagai solusi alternatif. Langkah ini bertujuan untuk memberikan fleksibilitas yang lebih besar dalam merancang pelatihan karyawan, sekaligus memastikan bahwa sistem tersebut sesuai dengan standar internal perusahaan. LMS internal ini dirancang untuk dapat beradaptasi dengan cepat terhadap perubahan kebutuhan pelatihan, sekaligus memungkinkan pengintegrasian fitur-fitur kustom yang relevan dengan operasional perusahaan. Selain itu, LMS ini akan mendukung pengelolaan materi pelatihan yang meliputi modul, video, dan dokumen, serta menyediakan alat evaluasi seperti soal pilihan ganda, esai, dan praktik.

Pengembangan LMS mandiri ini tidak hanya diharapkan mampu meningkatkan efisiensi operasional, tetapi juga memperkenalkan pengalaman belajar yang lebih interaktif dan personal bagi karyawan. Dengan fitur-fitur seperti forum diskusi, pelacakan kemajuan belajar, dan penilaian kinerja yang terintegrasi, sistem ini memberikan nilai tambah bagi proses pelatihan di PT Orela Shipyard. Selain itu, LMS internal ini memberikan perusahaan kendali penuh atas keamanan data, mengurangi ketergantungan pada teknologi pihak ketiga, dan menurunkan biaya operasional dalam jangka panjang.

Melalui pengembangan ini, PT Orela Shipyard berkomitmen untuk mendukung peningkatan kompetensi dan produktivitas karyawannya, sekaligus mempersiapkan mereka untuk menghadapi tantangan dan peluang di industri yang semakin kompetitif. Dengan sistem pembelajaran yang lebih terstruktur, fleksibel, dan aman, perusahaan berharap dapat menciptakan lingkungan kerja yang inovatif dan berkelanjutan.

KATA PENGANTAR

Puji syukur ke hadirat Tuhan Yang Maha Esa atas segala rahmat dan karunia-Nya, sehingga kami dapat menyusun dan menyajikan informasi mengenai PT Orela Shipyard.

PT Orela Shipyard adalah perusahaan yang bergerak di bidang pembangunan dan perbaikan kapal serta layanan galangan yang berfokus pada profesionalisme, inovasi, dan kualitas kerja tinggi. Berlokasi strategis dengan fasilitas yang lengkap, PT Orela Shipyard telah menjadi salah satu pelaku industri perkapalan yang terpercaya dan diakui di Indonesia. Dengan komitmen untuk mendukung sektor maritim nasional, PT Orela Shipyard berperan aktif dalam menyediakan layanan unggul yang dapat memenuhi kebutuhan pelanggan dalam bidang perkapalan, baik di dalam negeri maupun mancanegara.

Penyusunan informasi ini dimaksudkan untuk memberikan gambaran yang jelas mengenai visi, misi, layanan, serta kontribusi PT Orela Shipyard dalam sektor perkapalan dan kemaritiman. Kami berharap, kata pengantar ini dapat menjadi pintu awal dalam memahami lebih dalam mengenai komitmen PT Orela Shipyard dalam mendukung pengembangan industri maritim Indonesia dan menciptakan dampak positif bagi masyarakat luas.

Kami menyampaikan terima kasih kepada semua pihak yang telah membantu dalam penyusunan dan penyajian informasi ini. Semoga tulisan ini bermanfaat bagi seluruh pihak yang memerlukan, dan dapat memberikan inspirasi bagi perkembangan industri maritim yang lebih baik di masa depan.

Surabaya, November 2024

Penulis

DAFTAR ISI

| | Halaman |
|---|----------------|
| HALAMAN JUDUL..... | i |
| ABSTRACT..... | ii |
| KATA PENGANTAR | iii |
| DAFTAR ISI..... | iv |
| DAFTAR GAMBAR | vi |
| DAFTAR TABEL..... | vii |
| DAFTAR SEGMENT PROGRAM | viii |
| BAB I PENDAHULUAN..... | 1 |
| 1.1 Tujuan | 1 |
| 1.2 Ruang Lingkup..... | 2 |
| 1.3 Arsitektur Sistem..... | 3 |
| 1.4 Estimasi Waktu Pengerjaan..... | 4 |
| BAB II ANALISA..... | 6 |
| 2.1 Fact Finding | 6 |
| 2.2 Perbandingan Sistem LMS dengan Coursera..... | 8 |
| BAB III SISTEM DESAIN | 17 |
| 3.1 Desain Interface | 17 |
| 3.2 Activity Diagram..... | 18 |
| 3.3 Database Design..... | 21 |
| 3.4 Mockup Design | 22 |
| BAB IV IMPLEMENTASI | 40 |
| 4.1 konfigurasi | 40 |
| 4.2 Konfigurasi backend untuk sambung database | 43 |
| 4.3 Registrasi User | 44 |
| 4.4 Buat Kursus Baru | 47 |
| 4.5 Buat Pengumuman | 49 |

| | | |
|------------|------------------------------------|-----|
| 4.6 | JawabSoal/uploadJawaban..... | 51 |
| 4.7 | JawabSoal/uploadJawaban ke 2 | 53 |
| 4.8 | Chat | 54 |
| 4.9 | Buat Modul | 58 |
| 4.10 | Buat Soal | 59 |
| 4.11 | Buat Soal Bagian ke 2 | 61 |
| 4.12 | Absensi | 62 |
| 4.13 | Absensi Bagian ke 2..... | 63 |
| BAB V | TESTING | 66 |
| 5.1 | List Test Case..... | 66 |
| BAB VI | PENUTUP | 76 |
| 6.1 | Kesimpulan | 76 |
| 6.2 | Saran..... | 76 |
| LAMPIRAN A | KUESIONER | A-1 |
| LAMPIRAN B | TAMPILAN PROGRAM..... | B-1 |

DAFTAR GAMBAR

| Gambar | Halaman |
|---|---------|
| 2.1 UseCase Anak Magang | 10 |
| 2.2 UseCase Mentor | 12 |
| 2.3 UseCase Admin | 13 |
| 3.1 Activity Diagram Menjawab Modul | 19 |
| 3.2 Activity Diagram Membuat Modul | 20 |
| 3.3 Database Diagram | 21 |
| 3.4 Profile Halaman..... | 23 |
| 3.5 Halaman Materi..... | 24 |
| 3.6 Halaman Tugas..... | 24 |
| 3.7 Halaman Chat | 25 |
| 3.8 Halaman Kerja Tugas (Quiz) | 25 |
| 3.9 Halaman Edit Profile | 26 |
| 3.10 Halaman upload Tugas (Esai) | 26 |
| 3.11 Halaman Pengumuman Anak Magang..... | 27 |
| 3.12 Halaman Absensi..... | 28 |
| 3.13 Halaman Buat Soal..... | 28 |
| 3.14 Halaman Profile Mentor..... | 28 |
| 3.15 Halaman Materi Mentor | 29 |
| 3.16 Halaman Murid | 29 |
| 3.17 Halaman Chat | 30 |
| 3.18 Halaman Detail Murid..... | 30 |
| 3.19 Halaman jawaban Anak Magang | 31 |
| 3.20 Halaman Upload Materi/Soal..... | 31 |
| 3.21 Halaman Upload Soal Quiz..... | 32 |
| 3.22 Halaman Buat Modul | 32 |

| | | |
|------|----------------------------------|----|
| 3.23 | Halaman Pengumuman Mentor..... | 33 |
| 3.24 | Halaman Jadwal Pembelajaran..... | 34 |
| 3.25 | Halaman List Mentor | 35 |
| 3.26 | Halaman List Anak Magang..... | 35 |
| 3.27 | Halaman Detail Mentor | 35 |
| 3.28 | Halaman Detail Anak Magang | 36 |
| 3.29 | Halaman Detail Kursus | 36 |
| 3.30 | Halaman List Kursus | 37 |
| 3.31 | Halaman Tambah Mentor..... | 37 |
| 3.32 | Halaman Tambah Anak Magang..... | 38 |
| 3.33 | Halaman Tambah Kursus | 38 |
| 3.34 | Halaman Buat Pengumuman | 39 |
| 4.1 | Form Registrasi User..... | 41 |
| 4.2 | Form Kursus Baru | 46 |
| 4.3 | Halaman Add Announcement | 48 |
| 4.4 | Halaman homework | 52 |
| 4.5 | Halaman chat..... | 54 |
| 4.6 | Halaman buat modul | 57 |
| 4.7 | Halaman Buat Soal..... | 62 |

DAFTAR TABEL

| Tabel | | Halaman |
|-------|--------------------------------|---------|
| 2.1 | Kelebihan dan Kekurangan | 6 |
| 5.1 | Test Case UserData | 8 |
| 5.2 | Test Case Admin | 10 |
| 5.3 | Test Case Mentor | 12 |

| | | |
|-----|----------------------------|----|
| 5.4 | Test Case AnakMagang | 12 |
|-----|----------------------------|----|

DAFTAR SEGMENT PROGRAM

| Tabel | Halaman |
|--|---------|
| 4.1 Konfigurasi Code Untuk setup backend..... | 41 |
| 4.2 Konfigurasi backend untuk sambung database | 42 |
| 4.3 Registrasi user baru | 44 |
| 4.4 Buat kursus baru | 46 |
| 4.5 Buat pengumuman..... | 47 |
| 4.6 jawab soal/uploadjawaban..... | 50 |
| 4.7 jawab soal/upload bagian ke 2..... | 53 |
| 4.8 Halaman chat..... | 54 |
| 4.9 Halaman buat modul | 57 |
| 4.10 buat soal..... | 59 |
| 4.11 buat soal bagian ke 2 | 61 |
| 4.12 Absensi | 62 |
| 4.13 Absensi bagian ke 2..... | 64 |

BAB I

PENDAHULUAN

Learning Management System (LMS) merupakan solusi digital yang dirancang untuk mendukung proses pembelajaran secara mandiri dan terstruktur. LMS memungkinkan pengelolaan modul pelatihan, penilaian, dan komunikasi antara mentor dan anak magang dalam satu platform yang terintegrasi. Sistem ini dirancang untuk memudahkan pengguna dalam mengakses materi pelajaran, memantau kemajuan, serta menerima umpan balik secara langsung, sehingga menciptakan pengalaman pembelajaran yang efisien dan efektif.

1.1 Tujuan

Tujuan dari Learning Management System (LMS) adalah untuk memfasilitasi anak magang dalam menyelesaikan modul-modul pelatihan secara mandiri, dengan fokus pada pengembangan keterampilan dan pengetahuan yang diperlukan untuk menyelesaikan tugas dan proyek di PT. Orelashipyard. LMS akan menyediakan akses kepada anak magang untuk mengikuti modul pembelajaran yang dirancang sesuai dengan kebutuhan industri. Selain itu, sistem ini akan mendokumentasikan seluruh proses pembelajaran anak magang secara terstruktur, memungkinkan pemantauan kemajuan dan evaluasi kinerja secara sistematis. Setelah menyelesaikan pelatihan, anak magang akan menerima sertifikat sebagai bukti bahwa mereka telah menjalani program magang di PT. Orelashipyard.

Pengguna utama sistem LMS ini mencakup beberapa aktor dengan peran yang jelas. Admin akan berfungsi untuk mengelola keseluruhan sistem, termasuk pembuatan dan pengaturan kursus, pembaruan materi pelatihan, serta pengelolaan data pengguna, baik itu mentor maupun anak magang. Mereka juga bertanggung jawab untuk

memantau perkembangan peserta, menyusun laporan, dan memastikan sistem berjalan sesuai dengan standar yang telah ditetapkan. Mentor, di sisi lain, akan mengawasi dan mendampingi perkembangan anak magang melalui evaluasi kinerja, pemberian umpan balik, serta penugasan proyek. Mereka juga memiliki akses ke modul pelatihan yang relevan dengan program magang dan dapat berkomunikasi langsung dengan anak magang melalui forum atau diskusi dalam LMS.

Selain itu, anak magang akan menggunakan LMS untuk mengikuti kursus dan pelatihan yang disesuaikan dengan kebutuhan mereka selama magang. Mereka memiliki akses penuh ke materi pelajaran, tugas, serta penilaian, dan dapat memantau kemajuan mereka dalam menyelesaikan tugas serta menerima umpan balik dari mentor. Tim IT akan berperan dalam menjaga stabilitas dan keamanan sistem LMS, dengan memastikan integrasi LMS dengan infrastruktur teknologi perusahaan yang lain, seperti sistem manajemen pengguna. Mereka juga bertanggung jawab untuk melakukan pemeliharaan sistem, pembaruan, dan menangani masalah teknis yang muncul, serta memastikan keamanan data dan pengelolaan akses pengguna dalam sistem.

1.2 Ruang Lingkup

Program Learning Management System (LMS) yang akan dikembangkan menggunakan teknologi modern berbasis web ini akan menyediakan berbagai fitur untuk mempermudah proses pelatihan dan evaluasi. Sistem ini akan memungkinkan registrasi kandidat secara mandiri oleh karyawan atau melalui administrator. Pengguna akan dikelompokkan berdasarkan jabatan, divisi, atau jenis pelatihan yang diikuti, serta dapat melacak kemajuan mereka selama pelatihan, dengan sistem yang mencatat aktivitas dan hasil evaluasi untuk setiap peserta.

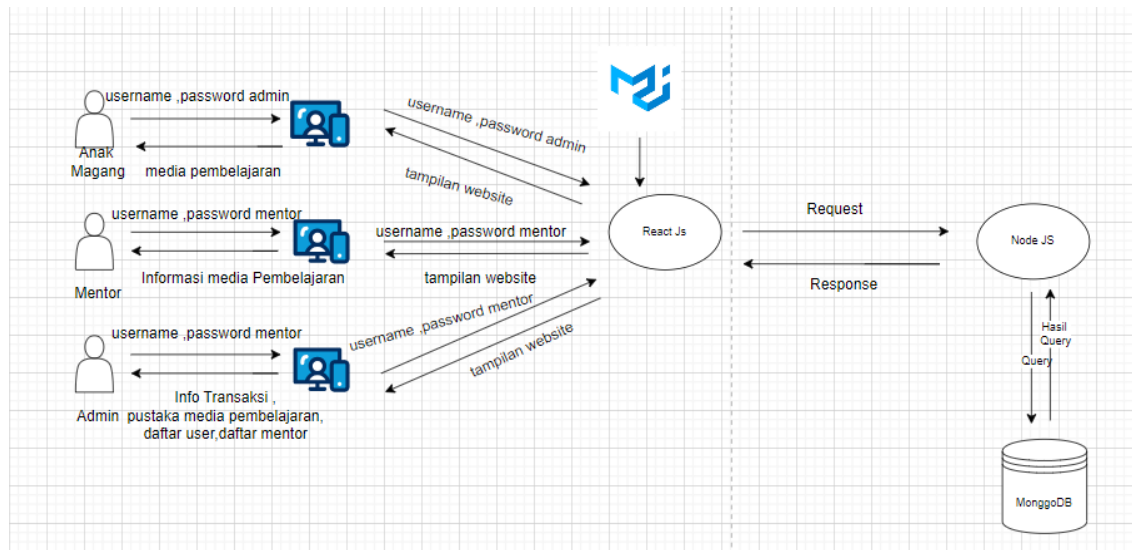
Dalam hal media pembelajaran, LMS akan menyediakan penyimpanan materi dalam berbagai format seperti video, PDF, dokumen, dan presentasi yang dapat diakses

dengan mudah oleh peserta pelatihan. Sistem ini juga mendukung media interaktif seperti simulasi, kuis, dan video tutorial, yang dirancang untuk memperkaya pengalaman belajar dan meningkatkan pemahaman peserta tentang materi pelatihan yang disampaikan.

LMS juga dilengkapi dengan fitur manajemen penilaian dan sertifikasi, di mana tes dan ujian dapat dibuat untuk mengevaluasi pemahaman peserta. Setelah menyelesaikan pelatihan dan lulus ujian, peserta akan menerima sertifikat digital yang terintegrasi langsung dengan profil mereka. Selain itu, sistem menyediakan dashboard untuk admin dan mentor untuk memantau kemajuan peserta secara individu atau keseluruhan, serta menghasilkan laporan kinerja yang dapat digunakan oleh manajemen untuk menganalisis efektivitas pelatihan dan kinerja karyawan.

1.3 Arsitektur Sistem

Pada bagian ini akan dijelaskan mengenai sistem yang akan dibuat, fungsi sistem Learning Management System dan arsitektur sistem dari Proposal SDP. Pada arsitektur ini akan dijelaskan mengenai cara kerja sistem secara garis besar beserta penjelasannya. Arsitektur sistem juga menjelaskan sistem setiap aktor yang terlibat. Terdapat 4 aktor yang saling berhubungan dalam website ini. Aktor tersebut yaitu admin, mentor, dan user. Setiap aktor mulai dari admin dan anak magang memiliki hak akses yang berbeda. Berikut merupakan arsitektur sistem dari Proposal SDP



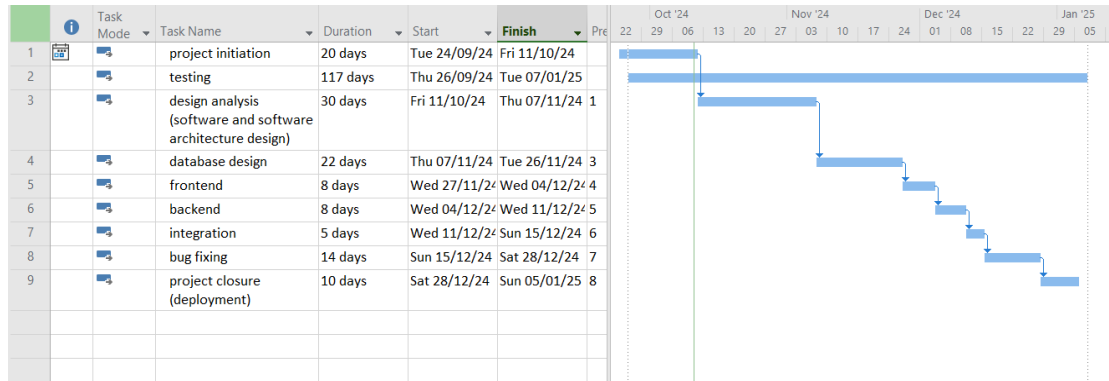
Gambar 1.3

Arsitektur Sistem

Pada gambar 1.3 dapat dilihat arsitektur sistem yang akan dibuat. Website ini membutuhkan koneksi internet yang nantinya akan berhubungan dengan web service untuk mengirimkan data ke dalam database. Website memiliki tampilan yang responsive sehingga dapat diakses melalui website komputer maupun smartphone. Dengan bantuan Progressive Web App, user dapat menginstall website ke smartphone. Tampilan dibantu oleh framework material ui yang memberikan kemudahan dalam membuat design website yang cantik dan responsive.

1.4 Estimasi Waktu Pengerjaan

Proyek ini diperkirakan akan selesai dalam waktu 2-3 bulan dengan rincian sebagai berikut:



Gambar 1.4

Gantt Chart Perkiraan Waktu Pengerjaan

Proyek ini dimulai dengan tahap inisiasi proyek selama 20 hari, dari 24 September hingga 11 Oktober 2024. Tahapan ini diikuti oleh analisis desain yang mencakup desain perangkat lunak dan arsitektur selama 30 hari, hingga 7 November 2024. Setelah analisis desain, tim akan mengembangkan desain basis data selama 22 hari hingga 26 November 2024. Proses desain ini membangun dasar bagi tahap pengembangan selanjutnya.

Pengembangan antarmuka dimulai dengan frontend development selama 8 hari, diikuti oleh backend development selama 8 hari berikutnya, dengan kedua fase ini berlangsung dari 27 November hingga 11 Desember 2024. Setelah backend selesai, proyek masuk ke tahap integrasi selama 5 hari hingga 17 Desember 2024, memastikan semua komponen yang telah dikembangkan bisa bekerja secara bersamaan. Perbaikan bug dilakukan setelah integrasi selama 14 hari, dari 15 hingga 28 Desember 2024, untuk memastikan bahwa aplikasi bebas dari masalah sebelum diterapkan.

Proyek diakhiri dengan tahap penutupan proyek (deployment) selama 10 hari, dari 28 Desember 2024 hingga 5 Januari 2025. Secara keseluruhan, proyek ini memakan waktu sekitar 3,5 bulan dari awal hingga akhir, dengan durasi terpanjang pada fase pengujian yang berjalan secara berkelanjutan selama 117 hari, mulai 26 September 2024 hingga 7 Januari 2025.

BAB II

ANALISA

Sistem pelatihan di PT. ORELA SHIPYARD dirancang untuk meningkatkan efektivitas pelatihan dan mendukung kinerja mentor serta peserta. Temuan utama meliputi penyusunan materi berbentuk modul atau video, pembuatan soal ujian, dan integrasi Learning Management System (LMS) untuk memudahkan pengelolaan. Durasi pelatihan yang disarankan adalah tiga hingga enam bulan, dengan evaluasi berkala dan fitur penilaian otomatis untuk soal pilihan ganda. Dengan sistem ini, perusahaan dapat menyelenggarakan pelatihan terstruktur yang memenuhi kebutuhan tenaga kerja berkualitas di setiap divisi.

2.1 Fact Finding

Dalam pengembangan sistem pelatihan untuk mentor di PT. ORELA SHIPYARD, beberapa temuan penting telah diidentifikasi. Materi pelatihan dapat berupa modul, video, atau dokumen lain yang disusun oleh mentor atau admin perusahaan. Materi ini disampaikan melalui platform seperti Google Classroom atau media internal yang mendukung proses pengajaran. Pembuatan soal ujian dilakukan oleh mentor dan mencakup tipe soal pilihan ganda, esai, maupun praktik. Penilaian soal pilihan ganda dapat dilakukan secara otomatis oleh sistem, sedangkan penilaian esai dan praktik memerlukan evaluasi manual dari mentor.

Durasi pelatihan bagi peserta magang direncanakan berlangsung antara tiga hingga enam bulan, di mana mentor akan memberikan evaluasi berkala berdasarkan kinerja peserta. Evaluasi ini dapat berupa penilaian harian atau mingguan, disertai dengan umpan balik untuk membantu pengembangan keterampilan peserta. Setiap divisi di PT. ORELA SHIPYARD memiliki mentor khusus yang memberikan bimbingan sesuai bidang spesialisasi masing-masing, memastikan peserta mendapatkan pelatihan yang relevan.

Sistem pelatihan ini juga mengintegrasikan Learning Management System (LMS) yang sebelumnya telah menggunakan Google Classroom, dengan harapan dapat menyempurnakan fitur manajemen materi, ujian, dan pelaporan perkembangan peserta. Sebagai tambahan, materi magang terdahulu akan tersedia untuk membantu mentor dalam memberikan referensi yang sesuai selama pelatihan berlangsung. Temuan ini menjadi dasar yang penting untuk menyusun sistem pelatihan yang lebih efektif dan sesuai kebutuhan perusahaan.

a. Wawancara

Dalam wawancara ini, LMS untuk anak magang di PT. Orela Shipyard dirancang untuk mendukung pembelajaran mandiri dan mencakup divisi-divisi seperti safety, IT, SCM, HCM, legal, dan lainnya. Materi pelatihan diunggah secara online oleh admin melalui sistem, dan peserta magang belajar selama lima hari sebelum mengerjakan tugas pada hari ke-6. Pelatihan ini melibatkan mentor yang dipilih dari karyawan perusahaan.

Sistem LMS akan mencakup pengelolaan materi, tugas, kuis, dan laporan perkembangan. Ujian dapat berupa pilihan ganda atau esai, dengan pilihan ganda menyediakan empat opsi. Penilaian menggunakan Google Forms dan dilakukan oleh mentor. Pembelajaran berlangsung secara online, menggunakan Microsoft Teams, dengan pertemuan online dua kali seminggu dan durasi pelatihan empat jam per minggu.

Selain materi yang disediakan oleh perusahaan, mentor juga dapat mengunggah modul pengajaran khusus. LMS akan diintegrasikan dengan media eksternal seperti Google Classroom, jika memungkinkan. Durasi pelatihan ditentukan, dan ada tes praktik yang harus dilalui peserta magang sebelum menyelesaikan pelatihan. Penilaian dilakukan secara mingguan oleh mentor, yang juga bertanggung jawab atas skema penilaian.

b. Observasi

Dalam observasi ini, kami mengamati fasilitas, manajemen, dan proses kerja yang ada di PT Orela Shipyard. Fasilitas galangan dilengkapi dengan teknologi modern yang memungkinkan pekerjaan dilakukan dengan efisien dan presisi tinggi. Selain itu, perusahaan menerapkan prosedur keselamatan kerja yang ketat untuk memastikan lingkungan kerja yang aman bagi para karyawannya, sekaligus melindungi aset perusahaan dan pelanggannya. Fasilitas-fasilitas ini termasuk dok kering, peralatan pengangkatan berat, ruang penyimpanan, serta fasilitas pemeliharaan yang memungkinkan perusahaan menangani berbagai jenis dan ukuran kapal.

Dalam aspek manajemen, PT Orela Shipyard menunjukkan komitmen terhadap peningkatan kualitas dan kepuasan pelanggan melalui pelatihan berkelanjutan bagi karyawan, baik di tingkat operasional maupun manajerial. Perusahaan juga mengutamakan kolaborasi antara divisi teknis dan administrasi untuk memastikan kelancaran proyek dari tahap awal hingga penyelesaian. Pengamatan menunjukkan bahwa PT Orela Shipyard menjalankan sistem manajemen terpadu yang berfokus pada efisiensi waktu dan pengelolaan sumber daya yang optimal, memungkinkan proyek dapat diselesaikan sesuai jadwal dan standar kualitas yang telah ditentukan.

Hasil observasi ini mencerminkan komitmen PT Orela Shipyard dalam menjaga standar profesionalisme yang tinggi, menjaga integritas dalam industri perkapalan, dan memperkuat posisinya sebagai mitra terpercaya dalam sektor maritim. Dengan infrastruktur, teknologi, dan tenaga ahli yang kompeten, PT Orela Shipyard siap mendukung kebutuhan industri perkapalan baik di dalam negeri maupun internasional.

2.2 Perbandingan Sistem LMS dengan Coursera

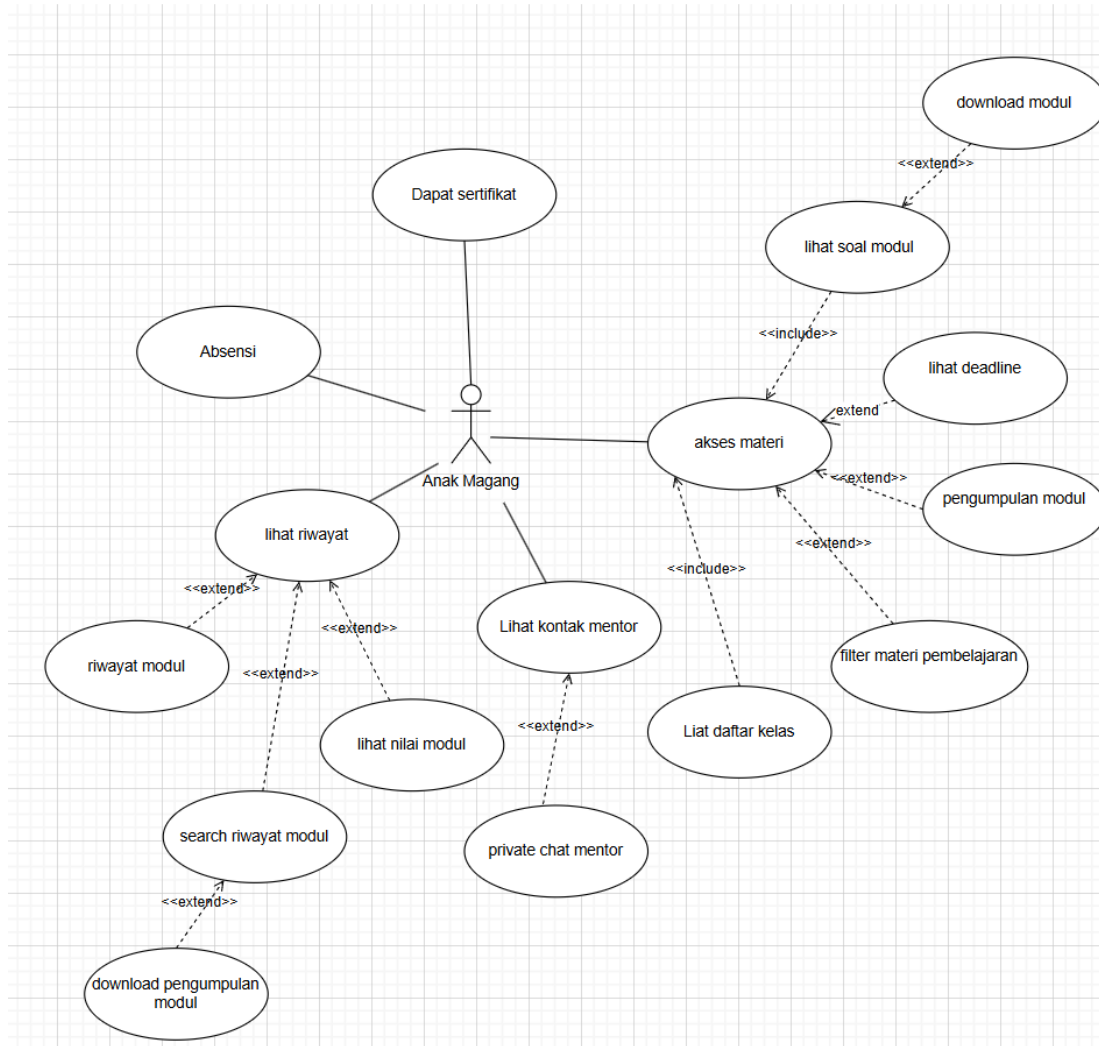
Perbandingan antara LMS PT Orela Shipyard dan Coursera menunjukkan perbedaan signifikan dari segi tujuan, fitur, dan pendekatan. LMS PT Orela Shipyard dirancang khusus untuk memenuhi kebutuhan internal perusahaan dalam melatih dan mengevaluasi karyawan di bidang perkapalan. Platform ini menyediakan materi pelatihan yang relevan dengan pekerjaan teknis, SOP, dan protokol keamanan, yang diakses oleh karyawan, mentor, dan manajemen. Sistem ini juga mengintegrasikan

penilaian otomatis yang langsung terkait dengan keterampilan pekerjaan, sehingga hasilnya lebih spesifik dan sesuai dengan kebutuhan operasional perusahaan. Monitoring kemajuan karyawan dilakukan melalui laporan yang terperinci, memungkinkan manajer untuk memantau keterampilan teknis secara langsung. LMS PT Orela Shipyard juga dilengkapi notifikasi otomatis terkait jadwal pelatihan, ujian, atau hasil penilaian, sehingga pengguna selalu up-to-date.

Sementara itu, Coursera adalah platform pembelajaran online global yang menawarkan kursus dari berbagai universitas dan institusi ternama di bidang teknologi, sains, bisnis, dan seni. Platform ini dirancang untuk pengguna yang mencari pengembangan keterampilan secara mandiri, seperti mahasiswa dan profesional dari berbagai industri. Penilaian di Coursera menggunakan format umum seperti kuis dan tugas, yang kurang spesifik terhadap suatu industri tertentu, tetapi cukup efektif untuk keperluan akademis dan keterampilan umum. Progres pengguna diukur berdasarkan penyelesaian kursus dan sertifikat, dan disertai dengan portofolio proyek untuk mendukung pengembangan keterampilan yang luas. Coursera berfokus pada aksesibilitas pembelajaran global dan memiliki katalog kursus yang luas, namun tidak dirancang untuk memenuhi kebutuhan pelatihan spesifik dalam suatu perusahaan seperti yang dilakukan oleh LMS PT Orela Shipyard.

a. Use Case :

Agar sistem dalam suatu proyek terstruktur dan mudah dipahami maka perlu dibuat design sistem. Salah satu contoh metode untuk menggambarkan design sistem adalah dengan use case. Use case sangat membantu dalam menjelaskan kegunaan dari role-role yang ada. Pada Gambar 2.1 berikut adalah beberapa use case yang telah dibuat berdasarkan yang terjadi di PT. Selaras Inti Usaha.



Gambar 2.1
Use Case Anak Magang

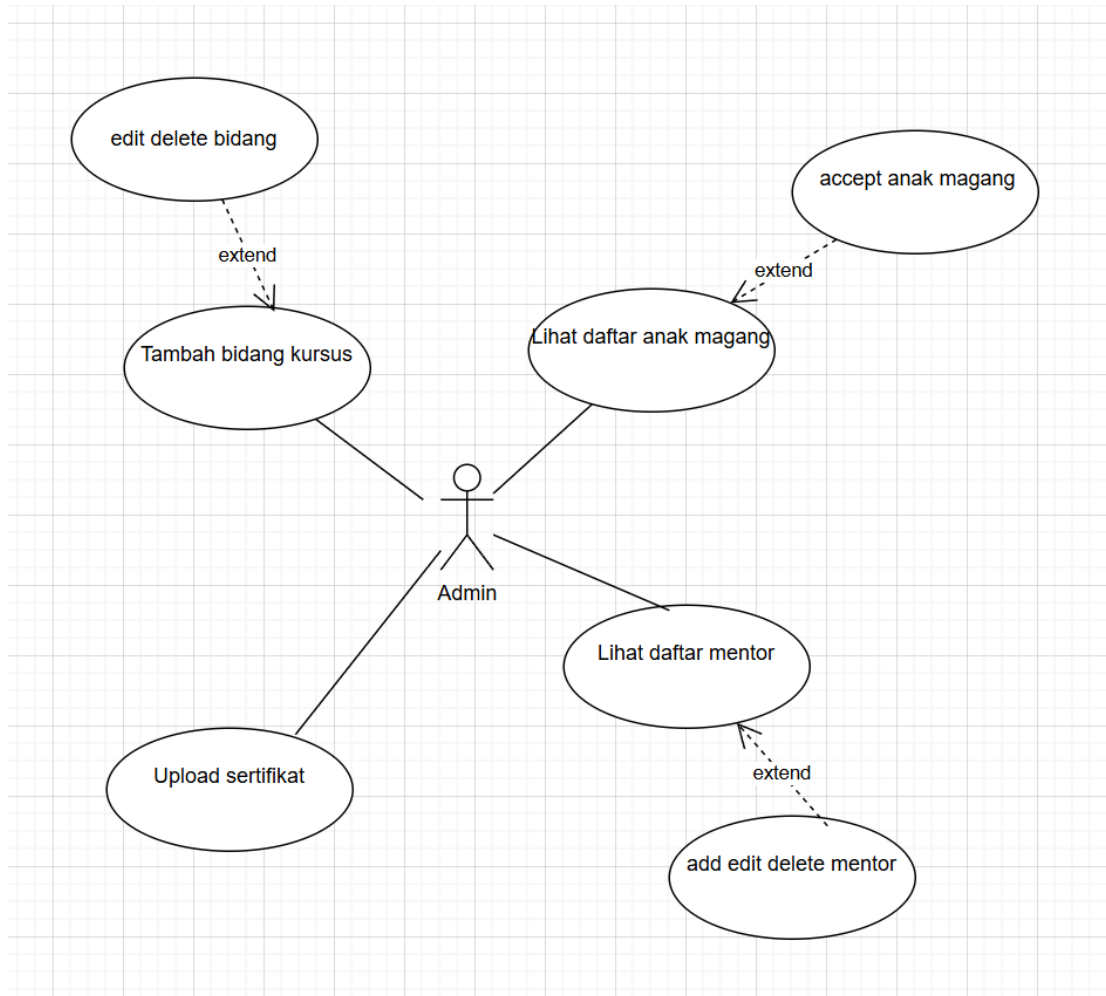
Gambar ini menunjukkan diagram use case untuk sistem manajemen pembelajaran yang berfokus pada fitur-fitur yang dapat diakses oleh seorang pengguna berperan sebagai "Anak Magang". Diagram ini memuat berbagai use case atau fungsionalitas yang dapat dilakukan oleh pengguna, seperti login, akses pembelajaran, melihat kontak mentor, dan riwayat pembelajaran. Setiap use case terhubung dengan fitur-fitur lain yang bisa diakses atau dikembangkan, yang ditandai dengan panah bertuliskan

"include" dan "extend". Hal ini menunjukkan adanya fungsionalitas tambahan atau ketergantungan fitur lain yang saling terkait dalam sistem ini.

Di bagian atas, beberapa use case dasar seperti "Login", "Dapat Sertifikat", dan "Absensi" menunjukkan fungsionalitas utama yang diperlukan oleh pengguna. Setelah berhasil login, pengguna dapat mengakses modul pembelajaran, yang mencakup fitur tambahan seperti melihat soal modul, melihat deadline, dan melakukan pengumpulan modul. Diagram ini juga memperlihatkan fitur "Lihat kontak mentor", yang memungkinkan pengguna untuk berkomunikasi dengan mentor. Fitur ini dapat diperluas dengan opsi private chat, yang memungkinkan komunikasi lebih langsung antara pengguna dan mentor.

Selanjutnya, fitur riwayat pembelajaran memungkinkan pengguna untuk melihat sejarah belajar mereka, termasuk nilai modul yang telah dicapai dan riwayat modul yang telah diakses. Terdapat pula fitur pencarian riwayat modul yang membantu pengguna menemukan modul tertentu dalam riwayat pembelajaran mereka. Diagram ini dengan fungsionalitas tambahan yang mendukung pengalaman belajar pengguna. menunjukkan alur akses fitur yang cukup komprehensif dalam sistem manajemen pembelajaran, memperlihatkan bagaimana tiap fitur bisa diakses serta dihubungkan.

Gambar ini adalah diagram use case yang menggambarkan peran dan fungsi yang dapat dilakukan oleh seorang admin dalam suatu sistem magang atau kursus. Pada diagram ini, admin ditampilkan sebagai aktor utama yang memiliki akses untuk mengelola berbagai aktivitas terkait kursus, mentor, dan anak magang. Admin dapat mengakses berbagai fungsi inti, seperti menambah bidang kursus, mengelola daftar mentor, melihat daftar anak magang, dan mengunggah sertifikat.

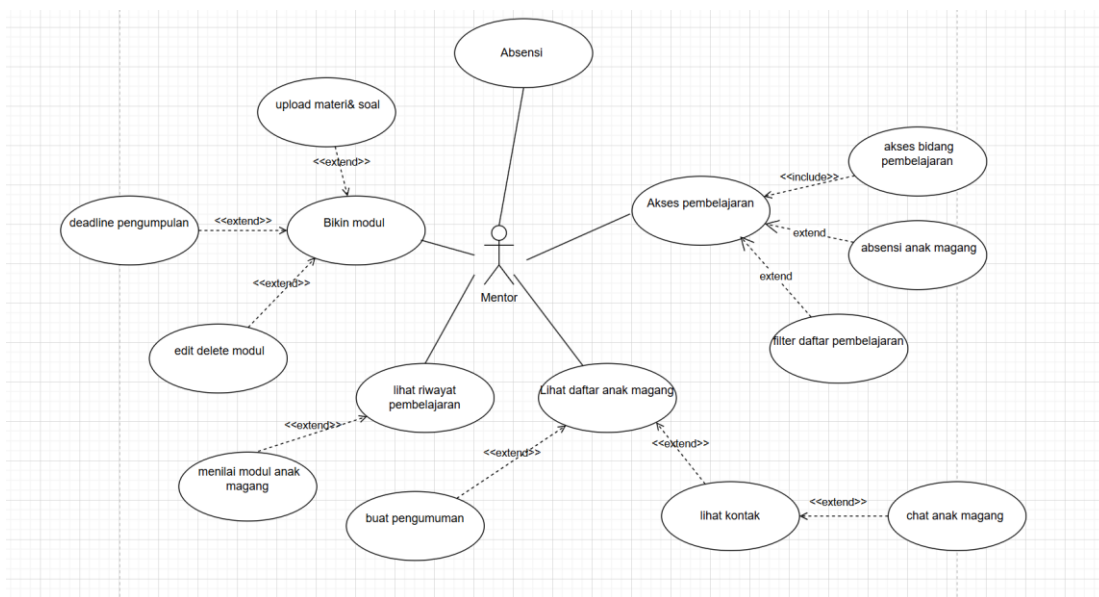


Gambar 2.2
Use Case Admin

Beberapa use case memiliki relasi "extend," yang menunjukkan fungsi tambahan yang dapat diperluas dari fungsi utama. Misalnya, untuk "Tambah bidang kursus," terdapat use case tambahan berupa "Extend edit delete bidang," yang memungkinkan admin tidak hanya menambah bidang, tetapi juga mengedit atau menghapus bidang yang sudah ada. Begitu pula dengan "Lihat daftar anak magang," yang memiliki fungsi extend "accept anak magang," sehingga admin dapat menerima anak magang dari daftar yang ditampilkan. Pada "Lihat daftar mentor," ada fungsi

tambahan untuk "Extend add edit delete mentor," di mana admin dapat menambah, mengedit, atau menghapus mentor dalam daftar.

Diagram ini menunjukkan bagaimana sistem ini dirancang untuk memberikan kontrol penuh kepada admin atas manajemen kursus dan mentor. Admin tidak hanya dapat menambah atau mengubah data bidang, mentor, dan anak magang, tetapi juga memastikan kelengkapan data dengan mengunggah sertifikat. Struktur use case seperti ini memastikan bahwa sistem memiliki fleksibilitas yang memungkinkan admin untuk melakukan berbagai tindakan sesuai kebutuhan dalam mengelola program magang atau kursus.



Gambar 2.3
Use Case Mentor

Gambar ini adalah diagram use case yang menggambarkan fitur-fitur yang dapat diakses oleh pengguna dengan peran "Mentor" dalam sistem manajemen pembelajaran. Diagram ini menunjukkan bagaimana seorang mentor dapat melakukan berbagai aktivitas, seperti absensi, membuat modul, mengakses pembelajaran, melihat

riwayat pembelajaran, dan melihat daftar anak magang. Setiap use case utama memiliki beberapa fitur tambahan yang dijelaskan dengan panah bertuliskan "include" dan "extend", yang menunjukkan bahwa ada fitur yang perlu dimasukkan atau dapat diperluas.

Use case "Bikin modul" merupakan salah satu fitur penting untuk mentor, yang mencakup kemampuan untuk mengunggah materi dan soal, serta menetapkan deadline pengumpulan. Selain itu, mentor juga dapat memperluas fungsionalitas ini dengan opsi untuk mengedit atau menghapus modul yang telah dibuat. Kemudian, mentor dapat melihat "Riwayat pembelajaran" dari anak magang, yang termasuk fitur untuk memberi penilaian pada modul-modul yang telah diselesaikan oleh mereka. Hal ini menunjukkan peran mentor dalam mengelola konten pembelajaran dan evaluasi terhadap kemajuan anak magang.

Fitur "Lihat daftar anak magang" memungkinkan mentor untuk memantau daftar peserta magang dan melibatkan beberapa fitur tambahan seperti filter daftar pembelajaran dan akses kontak. Dalam kontak ini, mentor dapat melihat informasi kontak anak magang atau memulai percakapan melalui chat. Selain itu, mentor dapat membuat pengumuman yang dapat dilihat oleh anak magang, yang mendukung komunikasi dan koordinasi dalam program pembelajaran. Diagram ini menggambarkan bagaimana peran mentor dirancang untuk mendukung aktivitas pembelajaran dan pengelolaan peserta dengan fitur yang cukup komprehensif.

b. Kelebihan dan Kekurangan:

Setiap sistem LMS memiliki kelebihan dan kekurangan yang disesuaikan dengan tujuan penggunaannya, sehingga penting bagi PT Orela Shipyard untuk memilih atau mengembangkan LMS yang tepat. Coursera, misalnya, unggul dalam menyediakan kursus dari berbagai institusi ternama, cocok untuk individu yang ingin

belajar mandiri dengan akses fleksibel ke berbagai materi umum dan sertifikasi global. Namun, karena sifatnya yang umum, Coursera tidak mampu memenuhi kebutuhan pelatihan teknis yang spesifik untuk lingkungan kerja perkapalan dan kurang mendukung penilaian serta pemantauan kinerja karyawan secara langsung. Oleh karena itu, sistem LMS yang dirancang khusus untuk PT Orela Shipyard akan fokus pada konten yang relevan dengan pekerjaan, penilaian berbasis keterampilan teknis, dan fitur monitoring yang memungkinkan manajer untuk melihat perkembangan karyawan secara real-time. Dengan pendekatan ini, LMS baru dapat lebih mendukung kebutuhan pelatihan internal yang spesifik dan memastikan ketercapaian standar operasional perusahaan.

| No. | Fitur | Sistem Coursera | Sistem Orela Shipyard |
|-----|-------------------------|-----------------|-----------------------|
| 1 | Pembayaran Kursus | V | X |
| 2 | E-Sertifikat | V | V |
| 3 | Live Chat dengan Mentor | X | V |
| 4 | Group Chat | X | V |
| 5 | Absensi Online | X | V |
| 6 | Pemilihan Kursus | V | X |
| 7 | Course enrollment | V | X |

Tabel 2.1
Tabel kelebihan dan kekurangan

Tabel ini menjelaskan LMS PT Orela Shipyard memiliki kelebihan dalam kustomisasi konten yang sesuai dengan kebutuhan spesifik perusahaan, seperti pelatihan teknis perkapalan, penilaian otomatis yang relevan dengan kinerja karyawan, serta fitur notifikasi dan monitoring kemajuan yang membantu manajemen dalam

memantau perkembangan keterampilan. Namun, LMS ini terbatas pada konten internal, sehingga kurang fleksibel untuk pembelajaran umum dan tidak dapat digunakan di luar lingkungan perusahaan. Sebaliknya, Coursera menawarkan katalog kursus yang sangat luas dari berbagai institusi ternama, cocok untuk pengguna yang ingin mengembangkan keterampilan umum secara mandiri dan mendapatkan sertifikasi yang diakui secara global. Meski demikian, Coursera tidak memiliki fitur yang disesuaikan untuk kebutuhan pelatihan spesifik suatu perusahaan dan kurang dalam pengawasan kinerja langsung, yang membuatnya tidak ideal untuk pelatihan khusus seperti yang dibutuhkan PT Orela Shipyard.

BAB III

SISTEM DESAIN

3. 1 Desain Interface

Desain antarmuka (interface) dari sistem Learning Management System (LMS) akan dirancang dengan fokus pada kemudahan penggunaan dan pengalaman pengguna yang optimal. Tampilan antarmuka akan menggunakan desain yang responsif, memastikan bahwa pengguna dapat mengakses LMS dengan lancar baik melalui perangkat desktop maupun perangkat mobile. Framework seperti Material UI akan digunakan untuk menciptakan elemen desain yang konsisten dan modern, sehingga mempercantik tampilan website dan memudahkan navigasi. Setiap elemen antarmuka akan dirancang untuk mempermudah pengguna dalam mencari informasi, mengakses materi pelatihan, dan menyelesaikan tugas-tugas yang diberikan, tanpa menghadapi kesulitan dalam penggunaan.

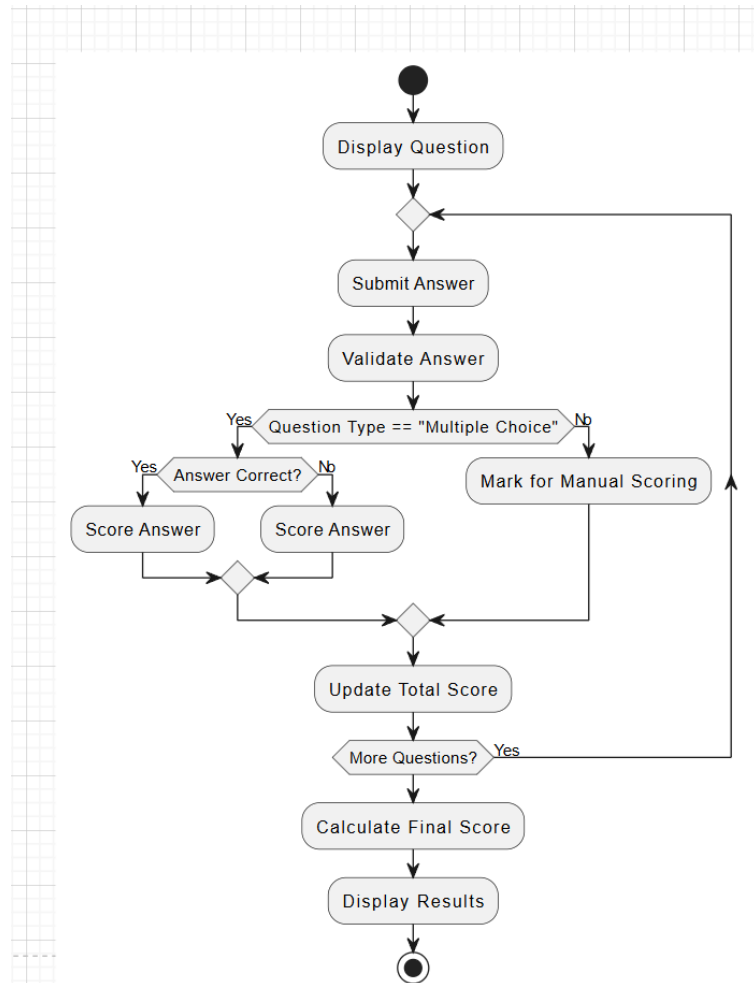
Untuk pengguna seperti admin, mentor, dan anak magang, antarmuka akan disesuaikan dengan peran masing-masing. Bagi admin, tampilan utama akan menyertakan fitur pengelolaan kursus, data pengguna, serta laporan kinerja peserta. Selain itu, admin akan memiliki akses ke dashboard yang menampilkan ringkasan aktivitas LMS secara keseluruhan. Mentor akan memiliki tampilan yang memungkinkan mereka untuk memberikan umpan balik kepada anak magang, mengelola evaluasi, serta mengakses materi pelatihan yang relevan dengan program magang. Untuk anak magang, desain antarmuka akan lebih terfokus pada akses materi, pelacakan kemajuan, dan komunikasi dengan mentor, sehingga memudahkan mereka dalam menyelesaikan tugas dan mengikuti pelatihan dengan efektif.

Antarmuka LMS juga akan dilengkapi dengan elemen-elemen interaktif seperti notifikasi, pop-up, dan formulir interaktif yang memudahkan komunikasi antara pengguna dan sistem. Misalnya, mentor dapat memberikan penilaian atau feedback

langsung kepada anak magang melalui formulir yang muncul di antarmuka. Selain itu, fitur pencarian dan filter akan ditambahkan untuk membantu pengguna menemukan materi atau modul pelatihan yang relevan dengan cepat. Secara keseluruhan, desain antarmuka LMS ini akan memastikan bahwa pengguna dapat berinteraksi dengan sistem dengan cara yang intuitif dan efisien, meningkatkan produktivitas dan pengalaman pembelajaran.

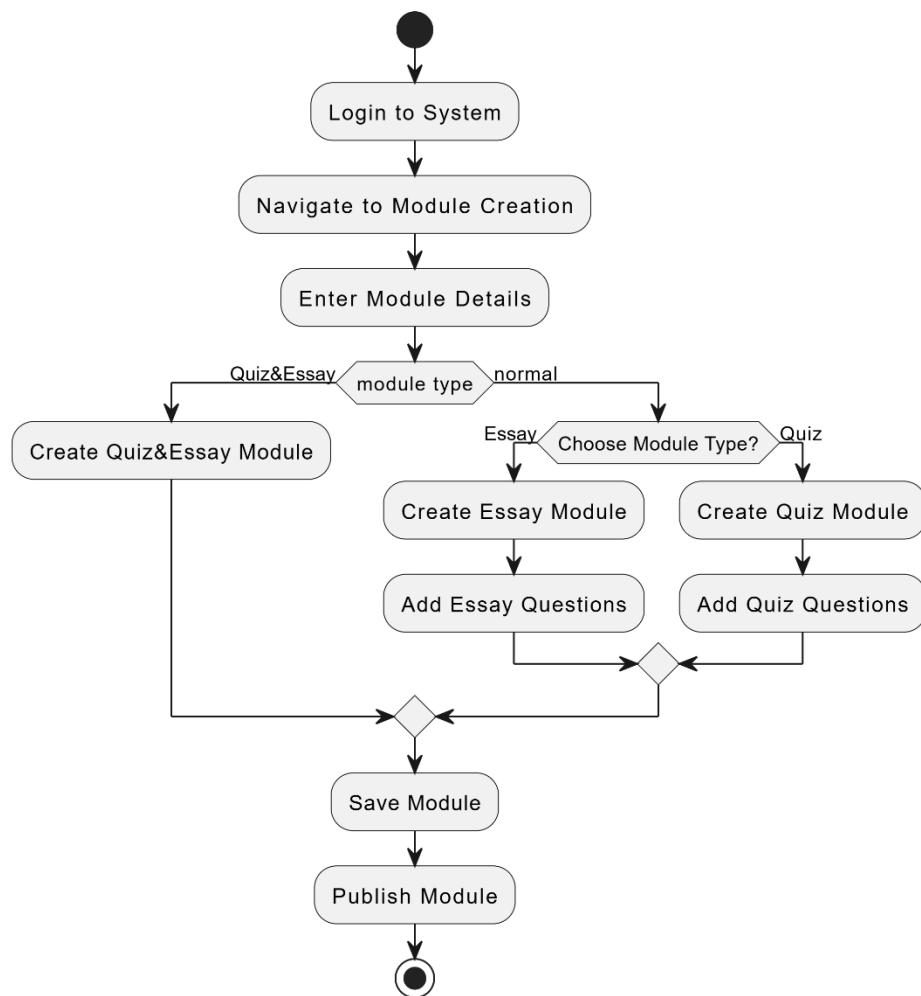
3.2 Activity Diagram

Activity Diagram akan digunakan untuk menggambarkan pemakaian sebuah aplikasi dalam scenario tertentu untuk memperjelas desain antarmuka dari sistem Learning Management System (LMS Desain antarmuka (interface) . pada bab ini Activity diagram yang akan digambarkan adalah activity diagram menjawab Soal dari sisi anak Magang dan membuat soal dari sisi mentor.



Gambar 3.1
Activity Diagram Menjawab modul

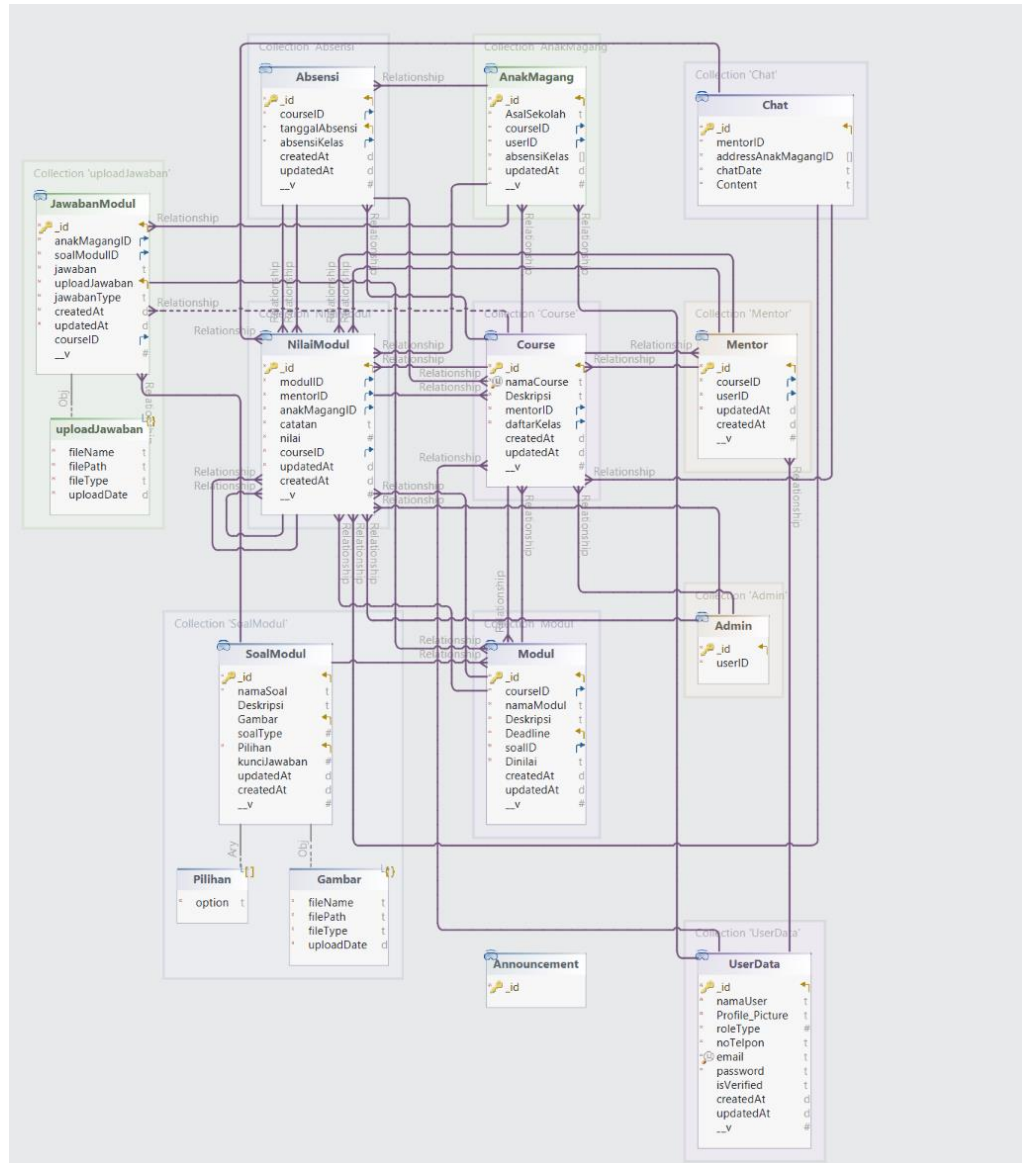
Activity diagram menjawab modul menggambarkan alur proses bagaimana anak Magang menjawab sebuah modul. Ketika anak Magang buka sebuah modul akan ditampilkan sebuah pertanyaan yang bisa bertipe pilihan ganda dan tulisan jika pertanyaan yang di kasih adalah pilgan nilai akan ditambah otomatis sesuai jumlah benar sebuah soal , tetapi jika tipe soal tulisan modul tidak akan langsung dinilai dan ditunggu dari mentor untuk menilai soal tersebut dan akan diulang sampai akhir soal yang lalu akan ditampilkan nilai secara langsung jika semua soal berbentuk pilgan dan nilai pending jika bentuk tulisan.



Gambar 3.2
Activity Diagram Membuat Modul

Activity diagram membuat modul menggambarkan alur proses bagaimana mentor membuat modul di LMS. Mentor dimulai dengan navigasi ke halaman modul Dimana ia sudah login lalu ia mengisi detail modul dari nama , tanggal dan soal yang akan dimasukkan, modul tersebut akan ada 3 tipe , pilgan, soal/esai, campuran. Ketika mentor selesai akan diupload ke web dan disimpan dalam database.

3.3 Database Design



Gambar 3.3
Database Diagram

Gambar database diagram akan menjelaskan bagaimana struktur penyimpanan sebuah data dalam database, dalam diagram ini MongoDB akan digunakan untuk

penyimpanan data user pada web. Pada gambar tersebut akan ada 16 tabel mulai dari userData. Table userData berfungsi untuk menyimpan data user yang pernah register didalamnya , yang berisi username, password , Alamat, tipe user, email, dan terVerifikasi. Table userdata akan menggunakan ini sebagai basis untuk sambungan dari table mentor, anakMagang , dan Admin, Dimana akan ada bidang terverifikasi sebagai sekuritas agar user tidak terotorisasi tidak bisa mengakses web.

Table Course digunakan untuk menyimpan bidang pembelajaran yang ada dengan menyimpan siapa pengajarnya, untuk table daftar kelas juga ada untuk menyimpan siapa saja di terdaftar di kelas , yang berisi id anakmagang dan id course nya. Table Absensi juga ada sebagai cara untuk mencatat kedatangan sebuah anak Magang, dicantumkan tanggal, id anakmagang, dan course id. Adanya juga pada table admin itu bersangkutan dengan table announcement, mentor dan anakMagang Dimana admin bisa simpan pengumuman dan memverifikasi mentor dan anakMagang yang akan disimpan pada table announcement ,mentor dan anakMagang.

Table modul akan berhubungan dengan table soalModul, nilai modul , dan jawaban modul semua berfungsi untuk menyimpan soal dan jawaban modul dalam table modul , dengan soal menyimpan soal yang dibuat mentor dan nilai modul yang dipakai untuk menilai hasil modul dari anak magang, dan jawaban disimpan dari input anakMagang. Table chat akan digunakan untuk menyimpan data percakapan antara mentor dan anakmagang yang disimpan akan menjadi percakapan, tanggal dan siapa mentor nya dan anakmagangnya.

3.4 Mockup Design

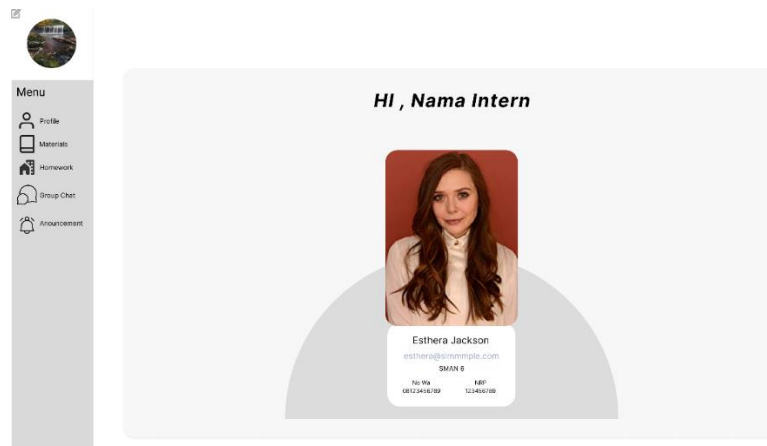
Mockup design berisi Kumpulan gambar dari tampilan pada sistem yang dibuat yang dirancang untuk memberikan gambaran awal mengenai tampilan dan fungsi sistem. Bagian ini bertujuan untuk menjelaskan bagaimana bentuk tampilan website yang sedang dirancang akan ditampilkan serta bagaimana interaksi antara user dengan sistem yang sedang dibuat. Untuk detail lebih lanjut mengenai Mockup

Design bisa dilihat pada link berikut

<https://www.figma.com/design/c2UTgKOI1Luhtth0l7aqNq/Untitled?node-id=104-3552&node-type=frame&t=0BWQP2JrPJhBmcIf-0>

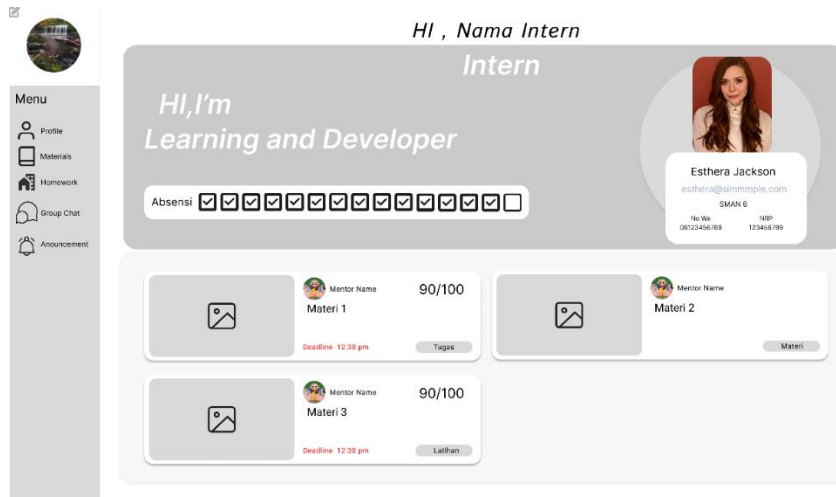
a. Halaman Anak Magang

pada mockup design akan ada 3 kategori halaman, yaitu anak magang , admin dan mentor pada mockup design ini, yang akan mejadi contoh tampilan website proyek ini. Halaman ini akan bersangkutan dengan anak magang, dimana ketika anak magang membuka website yang dijumpai adalah gambar-gambar halaman di sini.



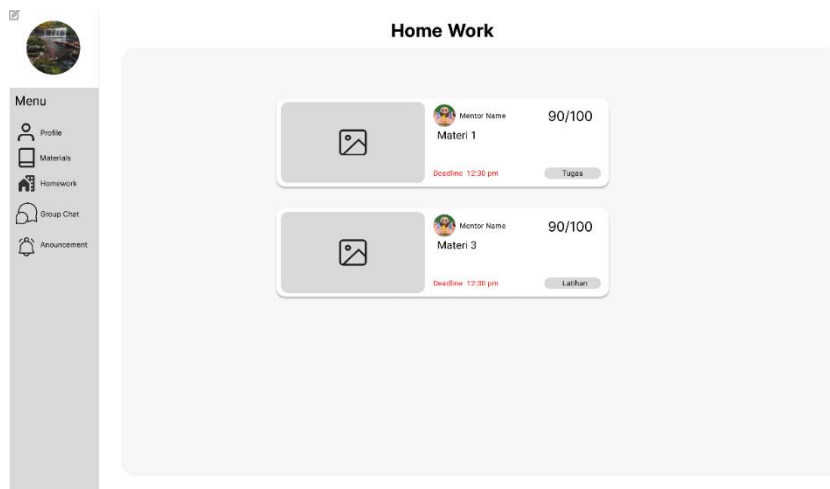
Gambar 3.4
Profile Halaman

Gambar 3.4 menggambarkan profil sebuah anak magang, di mana user bisa melihat biodata mereka , seperti nama , email, nomor telpon dan dll. Halaman ini penting untuk memastikan data yang diinput sudah disimpan.



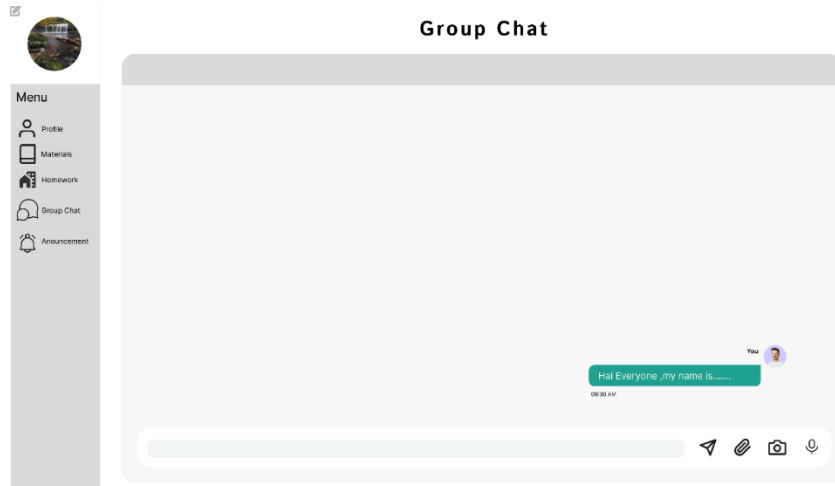
Gambar 3.5
Halaman Materi

Pada gambar 3.5, anak magang bisa lihat list materi mereka. Dengan ditulis nya kursus apa yang user ikuti dan detail profile mereka ditampilkan lalu menampilkan materi yang diupload oleh Mentor dalam kursus tersebut.



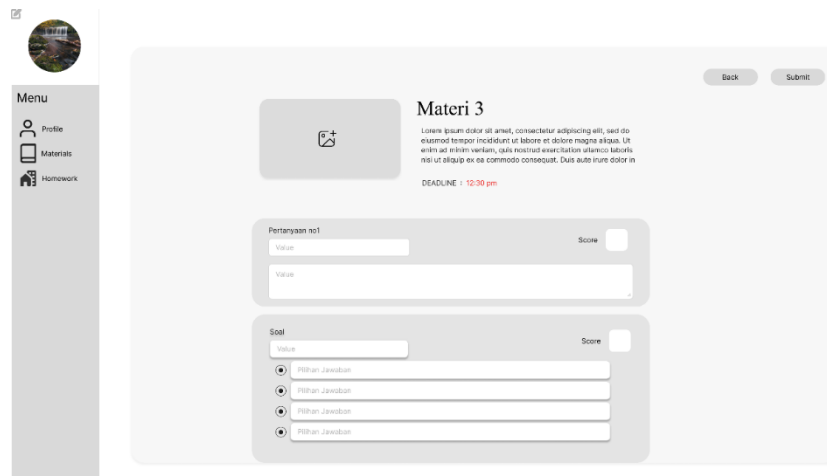
Gambar 3.6
Halaman Tugas

Pada gambar 3.6, anak magang bisa melihat tugas mereka yang sedang aktif dengan tombol kerjakan agar bisa memulai tugas / latihan, yang lalu akan diberi nilai kalau sudah selesai dinilai oleh Mentor.



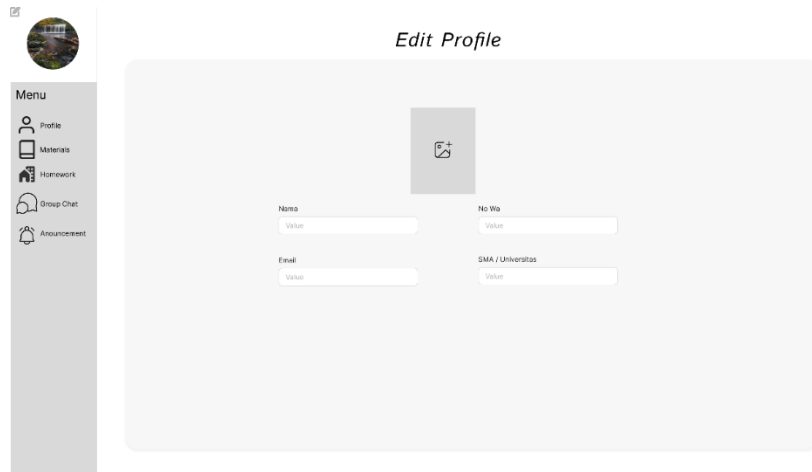
Gambar 3.7
Halaman Chat

Pada gambar 3.7, anak magang bisa melakukan Chatting dengan Mentor jika mau menanyakan sesuatu kepada mentor. Pada halaman ini akan ada button untuk isi gambar dan attachment.



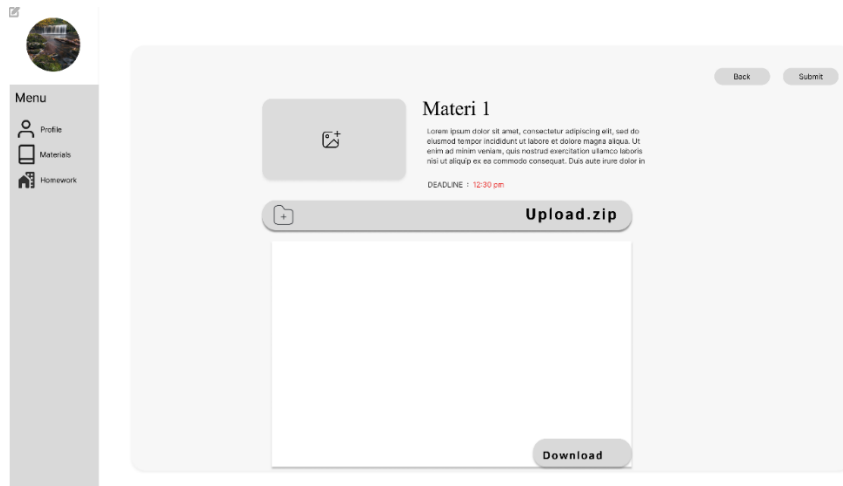
Gambar 3.8
Halaman Kerja Tugas (Quiz)

Pada gambar 3.8, anak magang akan ada halaman untuk mengerjakan tugas yang diberi oleh mentor pada gambar tersebut akan bisa diisi bentuk soal dan quiz,



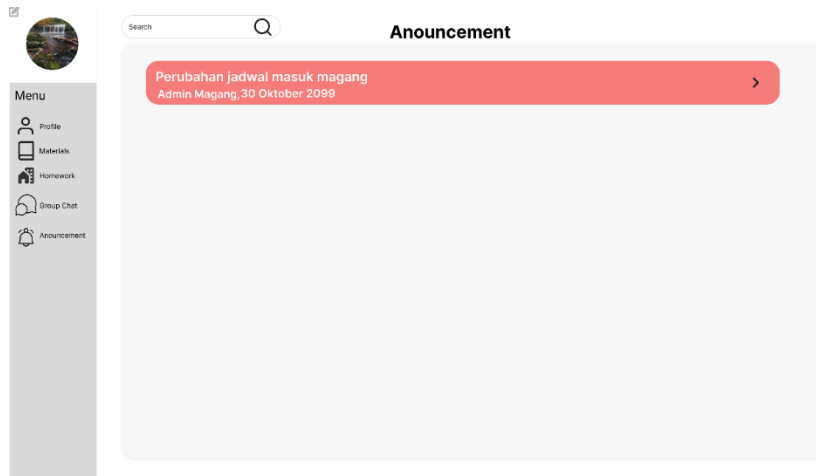
Gambar 3.9
Halaman Edit Profile

Pada gambar 3.9, anak magang akan bisa edit profile mereka sebagai opsi, hanya email yang tidak bisa diedit, tetapi akan bisa mengganti nomor telpon dan biodata lainnya jika perlu.



Gambar 3.10
Halaman upload Tugas (Esai)

Pada gambar 3.10, anak magang akan bisa upload Tugas jika soal tidak dalam bentuk pilihan ganda dan tulis esai. Halaman upload ini akan bisa menerima file pdf yang nantinya diperiksa oleh mentor.

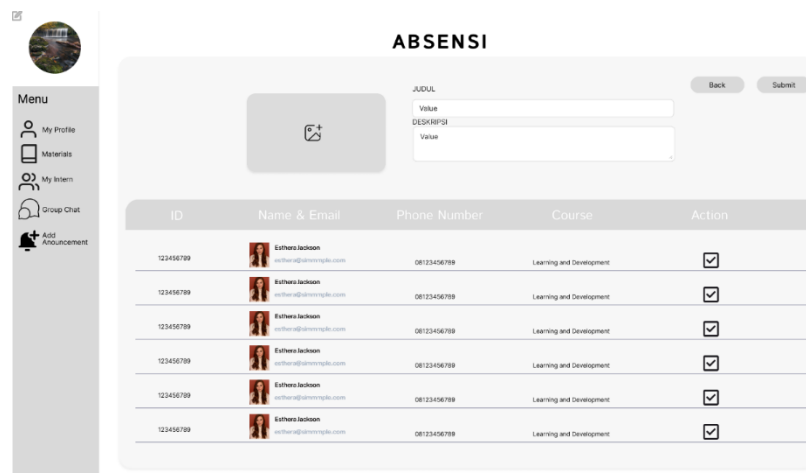


Gambar 3.11
Halaman Pengumuman Anak Magang

Pada gambar 3.11, anak magang akan bisa upload Tugas jika soal tidak dalam bentuk pilihan ganda dan tulis esai. Halaman upload ini akan bisa menerima file pdf yang nantinya diperiksa oleh mentor.

b. Halaman Mentor

Pada Bagian Halaman Mentor, mockup design disini akan semua ditemui oleh mentor ketika ia login, dimana Ia bisa absen murid, tambah soal, buat modul , tambah materi lalu chat dengan murid.



Gambar 3.12

Halaman Absensi

Pada gambar 3.12, Mentor bisa melakukan absensi pada halaman ini , untuk mencatat kehadiran anak magang pada tanggal itu.

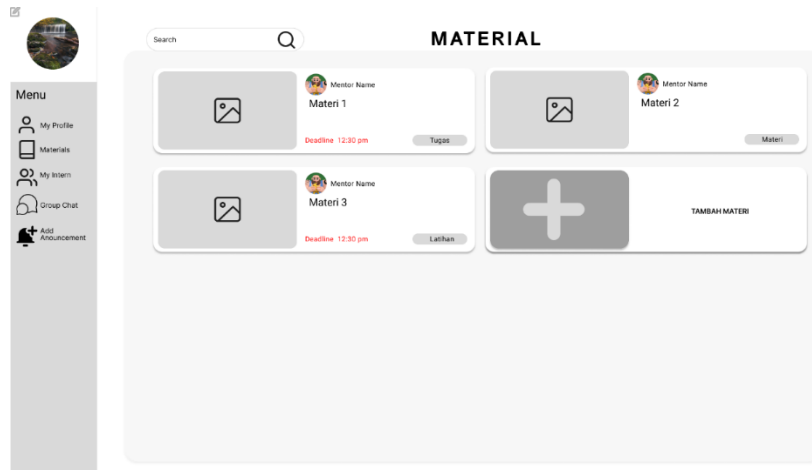
Gambar 3.13
Halaman Buat Soal

Pada gambar 3.13, Mentor bisa melakukan absensi pada halaman ini , untuk mencatat kehadiran anak magang pada tanggal itu.



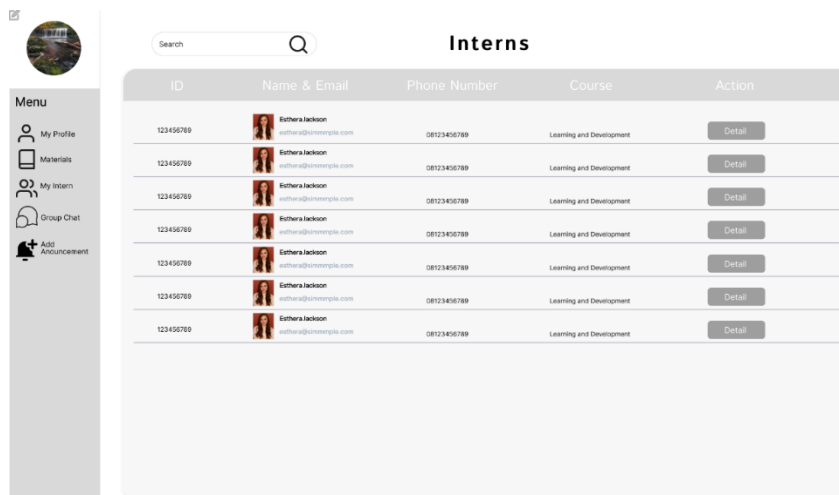
Gambar 3.14
Halaman Profile Mentor

Pada gambar 3.14, Mentor bisa melihat detail profile mereka seperti pada gambar 3.4 halaman profile anak magang. Dimana ia bisa lihat biodata.



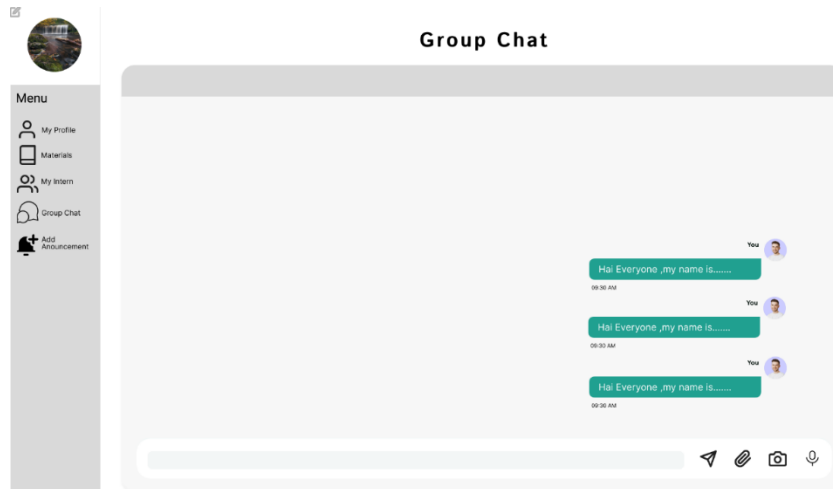
Gambar 3.15
Halaman Materi Mentor

Pada gambar 3.15, Mentor bisa melihat materi yang mereka upload di sini dengan ada search filter untuk memfilter materi yang ini dicarikan agar mudah untuk mencari materi yang ingin dilihat.



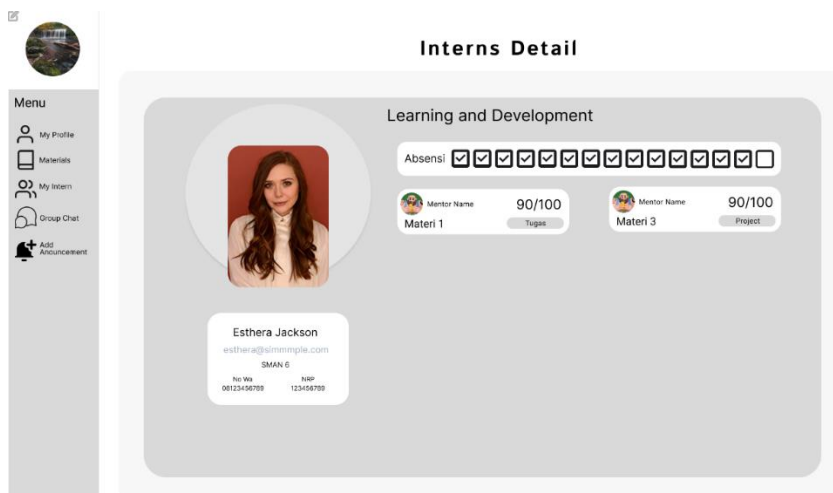
Gambar 3.16
Halaman Murid

Pada gambar 3.16, mentor melihat anak Magang pada halaman murid, dimana ia bisa ketuk tombol detail untuk melihat data anak magang dengan lebih detail. Pada halaman ini hanya id, gambar, nama, nomor telpon dan kursusnya diperlihatkan.



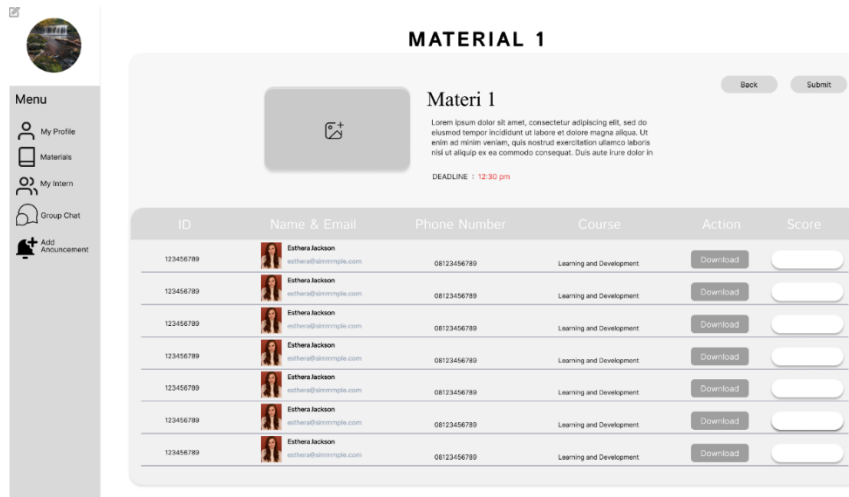
Gambar 3.17
Halaman Chat

Pada gambar 3.17 , mentor bisa chat atau menerima chat dari anak magang jika perlu pada halaman chat, fungsi nya akan bekerja seperti yang ditampilkan sesuai dengan gambar 3.7 halaman chat anak magang.



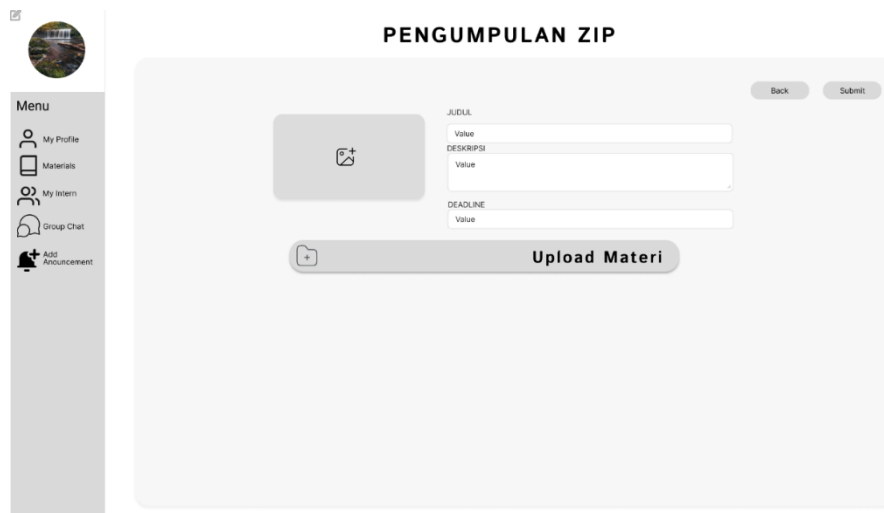
Gambar 3.18
Halaman Detail Murid

Pada gambar 3.18 , mentor melihat detail anak magang yang akan menampilkan detail sekilas seperti pada gambar 3.16 memperlihatkan id, gambar , nama, nomor telpon dan kursusnya, , lalu detail anak magang akan juga memperlihatkan berapa kali Ia absen lalu nilai untuk tugas nya yang sudah dikerjakan dan belum dikerjakan.



Gambar 3.19
Halaman jawaban Anak Magang

Pada gambar 3.19 , mentor melihat halaman Jawaban sebagai perantara untuk download hasil kerja anak magang, seperti pada list anak magang , format berbentuk list hanya ada tombol download untuk hasil kerja mereka jika sudah dikumpul



Gambar 3.20
Halaman Upload Materi/Soal

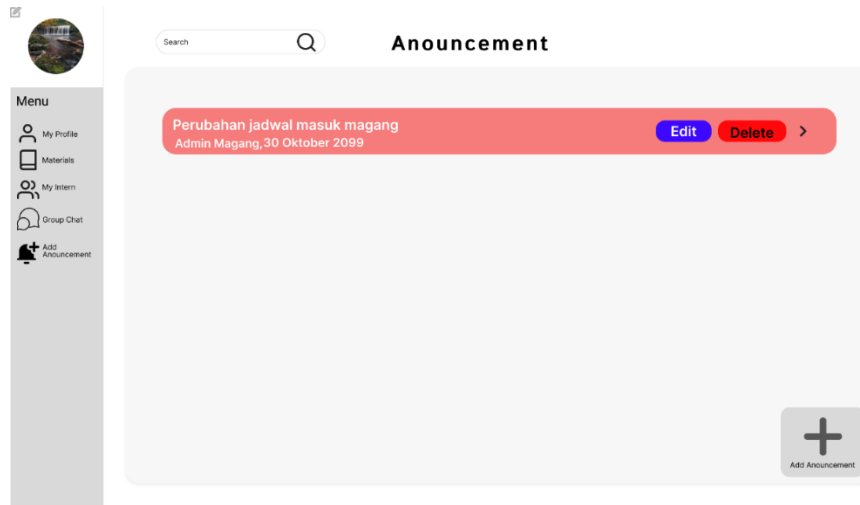
Pada gambar 3.20, mentor bisa upload soal lewat halaman ini jika soal berbentuk file . dimana nantinya pada sisi anak magang bisa di lihat dan didownload untuk tugas.

Gambar 3.21
Halaman Upload Soal Quiz

Pada gambar 3.21, mentor akan mengupload soal bisa berbentuk pilihan ganda atau diketik manual. Agar bisa dibuatnya latihan untuk anak magang.

Gambar 3.22
Halaman Buat Modul

Pada gambar 3.22, mentor akan buat modul dengan isi nya merupakan hasil dari halaman soal pada gambar 3.21 jika diinput dan ditekan tombol submit. Pada halaman modul diberi input untuk soal yang mana akan dimasukkan ke modul.

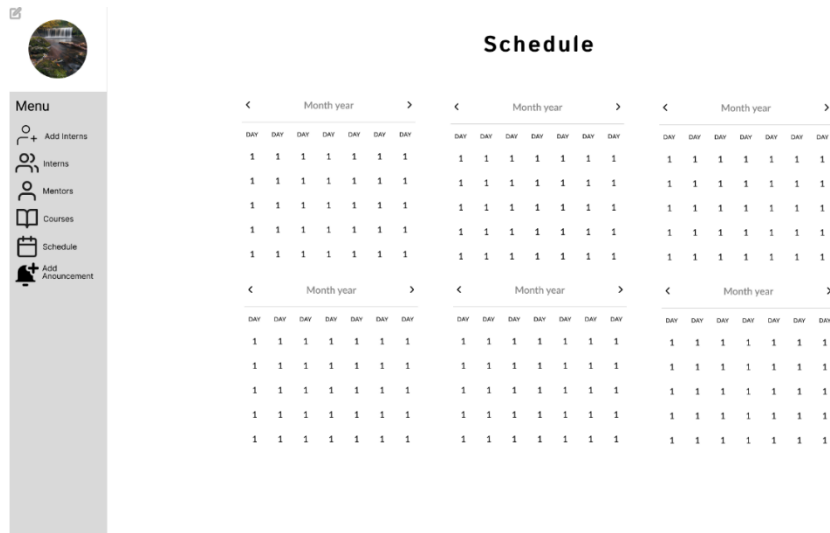


Gambar 3.23
Halaman Pengumuman Mentor

Pada gambar 3.23, mentor bisa membuat pengumuman untuk menginfokan anak magang pada kursus yang diajari oleh mentor tersebut dengan adanya tombol edit, delete dan tambah pengumuman .

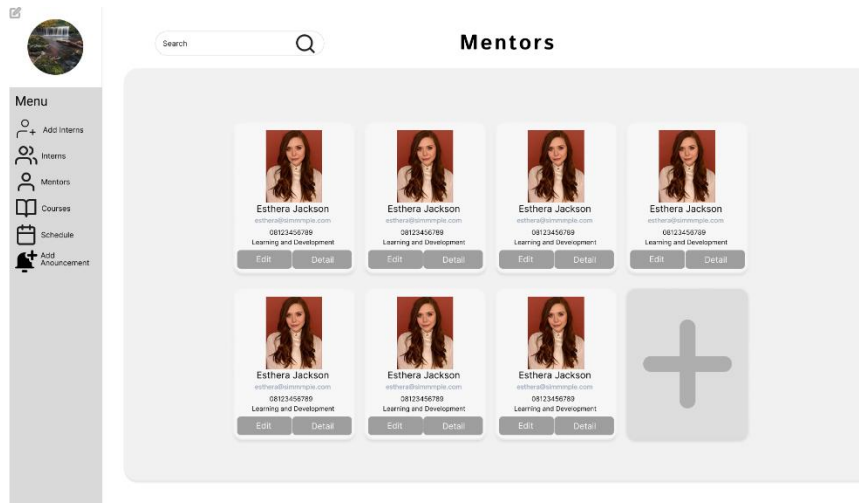
c. Halaman Admin

Pada halaman Admin, admin bisa mengatur data penting dari mentor dan anak magang , dimana admin bisa tambah edit mentor dan anak magang, lalu membuat jadwal pembelajaran , pengumuman penting dan menambahkan kursus baru untuk memantau aktivitas dalam website.



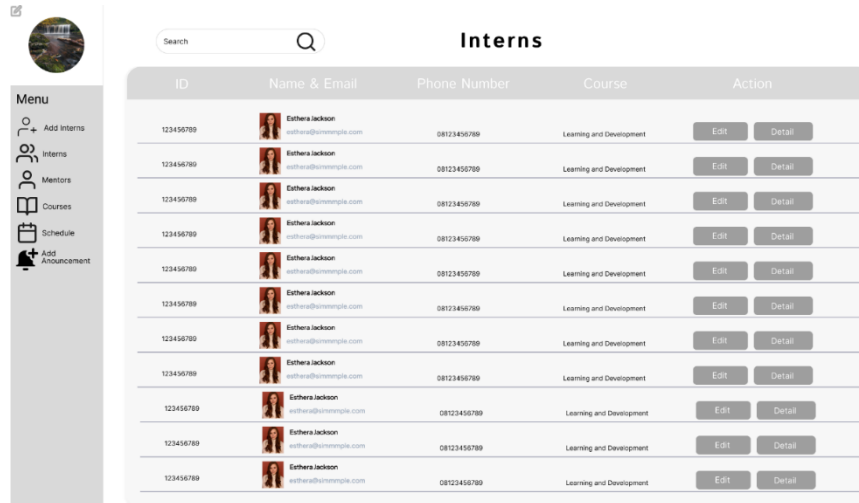
Gambar 3.24
Halaman Perjadwalan

Pada gambar 3.24, admin akan bisa membuat jadwal pembelajaran sesuai aktivitas pembelajaran yang berjalan. Dimana admin bisa upload calendar untuk dilihat anak magang dan mentor.



Gambar 3.25
Halaman List Mentor

Pada gambar 3.25, admin bisa melihat list mentor , untuk menambah, menghapus dan mengubah data mentor jika ada perubahan data pada mentor seperti kontak, cv dan data penting lainnya.



| ID | Name & Email | Phone Number | Course | Action |
|-----------|--|--------------|--------------------------|-------------|
| 123456789 | Esther Jackson esthera@innomingle.com | 08123456789 | Learning and Development | Edit Detail |
| 123456789 | Esther Jackson esthera@innomingle.com | 08123456789 | Learning and Development | Edit Detail |
| 123456789 | Esther Jackson esthera@innomingle.com | 08123456789 | Learning and Development | Edit Detail |
| 123456789 | Esther Jackson esthera@innomingle.com | 08123456789 | Learning and Development | Edit Detail |
| 123456789 | Esther Jackson esthera@innomingle.com | 08123456789 | Learning and Development | Edit Detail |
| 123456789 | Esther Jackson esthera@innomingle.com | 08123456789 | Learning and Development | Edit Detail |
| 123456789 | Esther Jackson esthera@innomingle.com | 08123456789 | Learning and Development | Edit Detail |
| 123456789 | Esther Jackson esthera@innomingle.com | 08123456789 | Learning and Development | Edit Detail |
| 123456789 | Esther Jackson esthera@innomingle.com | 08123456789 | Learning and Development | Edit Detail |
| 122456789 | Esther Jackson esthera@innomingle.com | 08123456789 | Learning and Development | Edit Detail |
| 123456789 | Esther Jackson esthera@innomingle.com | 08123456789 | Learning and Development | Edit Detail |
| 123456789 | Esther Jackson esthera@innomingle.com | 08123456789 | Learning and Development | Edit Detail |

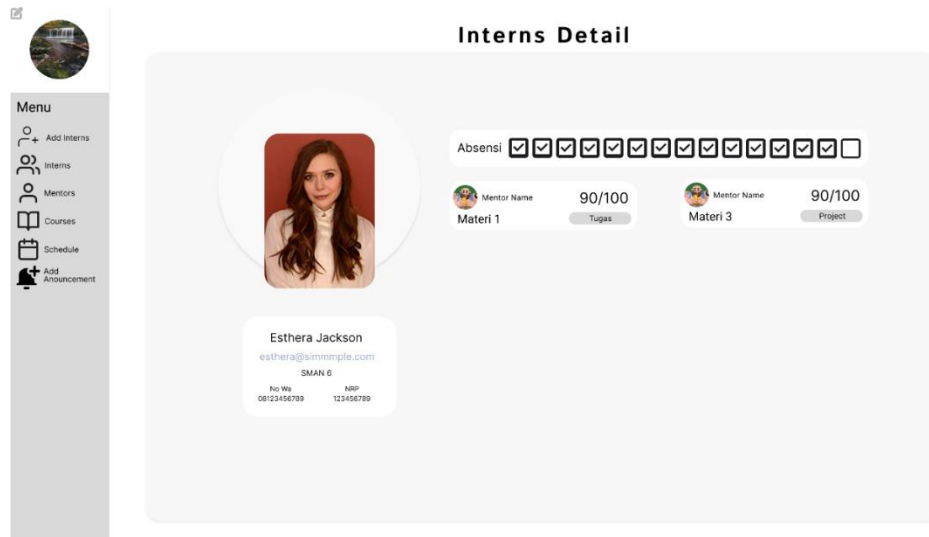
Gambar 3.26
Halaman List Anak Magang

Pada gambar 3.26, admin bisa melihat list anak Magang , untuk menambah, menghapus dan mengubah data anak Magang jika ada perubahan data pada anak magang seperti kontak, cv dan data penting lainnya. Seperti halaman mentor pada gambar 3.25



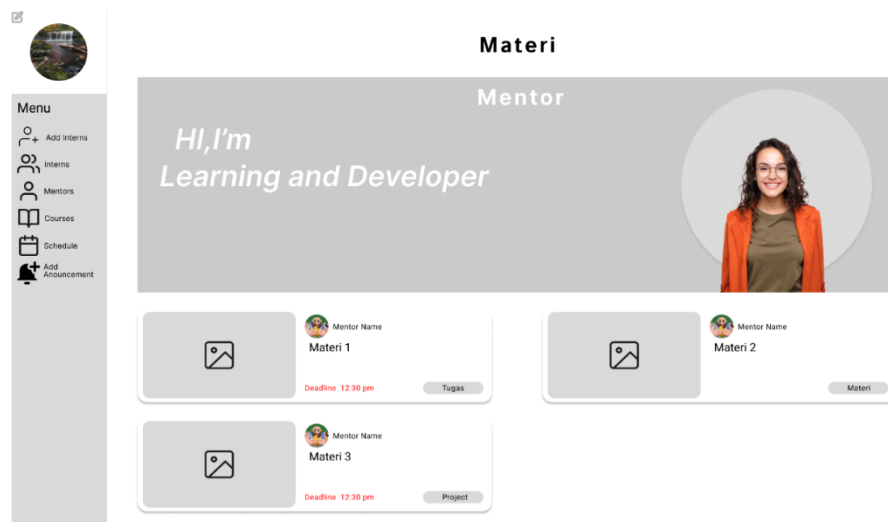
Gambar 3.27
Halaman Detail Mentor

Pada gambar 3.27, admin bisa melihat detail mentor seperti yang dibahas pada gambar 3.25 , detail mentor akan mencakupi biodata penting seperti sertifikat kerja dan dll.



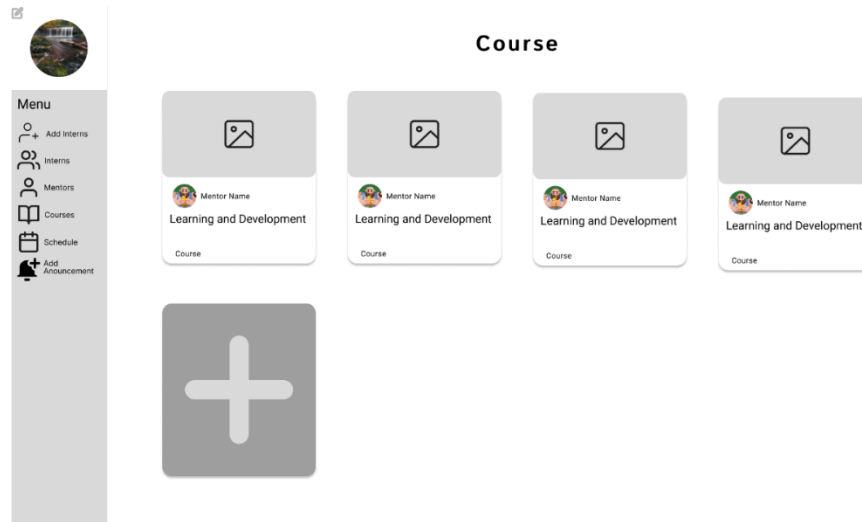
Gambar 3.28
Halaman Detail Anak Magang

Pada gambar 3.28, admin bisa melihat detail anak magang seperti yang dibahas pada gambar 3.26 , detail anak magang akan mencakupi biodata penting seperti portfolio , kontak darurat dan dll.



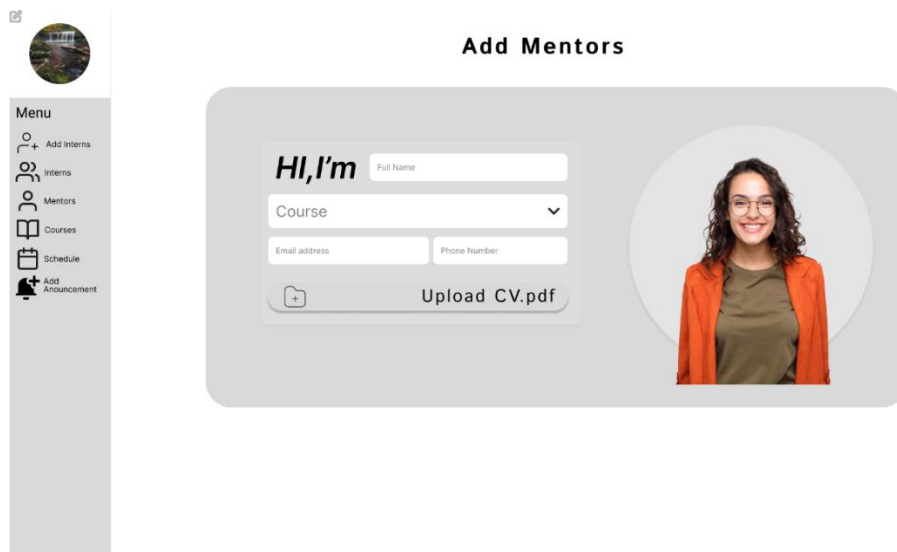
Gambar 3.29
Halaman Detail Kursus

Pada gambar 3.29, admin bisa melihat detail kursus , dimana kursus akan ada detail pengajar dan daftar kelas.



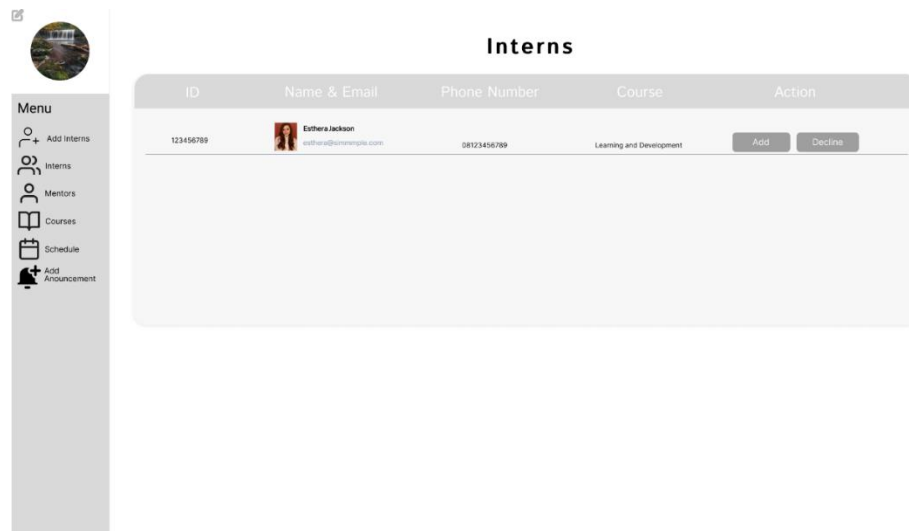
Gambar 3.30
Halaman List Kursus

Pada gambar 3.30, admin bisa melihat list kursus , untuk menambah atau edit, data kursus supaya mengkategorikan pelajaran apa yang diajarkan kepada anak Magang.



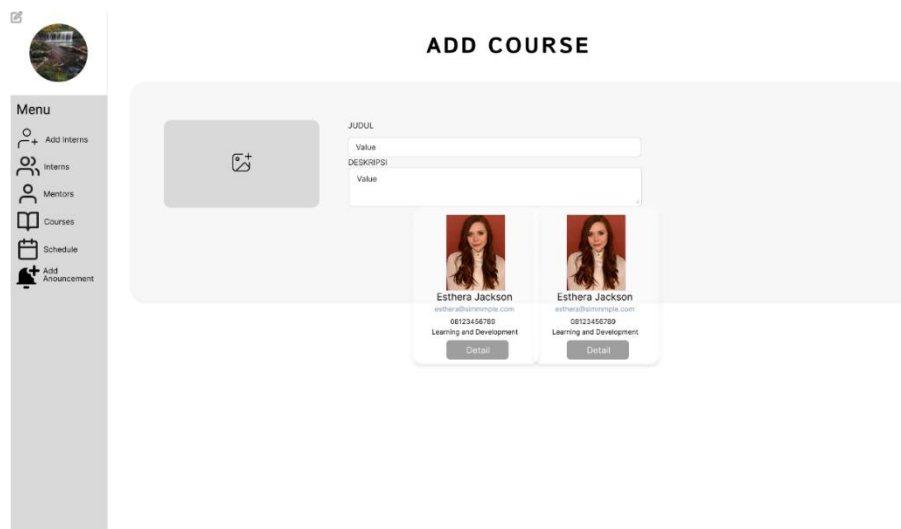
Gambar 3.31
Halaman Tambah Mentor

Pada gambar 3.31, admin bisa tambah mentor , sebagai opsi jika mentor belum register pada website tapi sudah merupakan mentor pada perusahaan nya. Akan ada input field, nama, kursus, email, nomor telpon dan sertifikat cv.



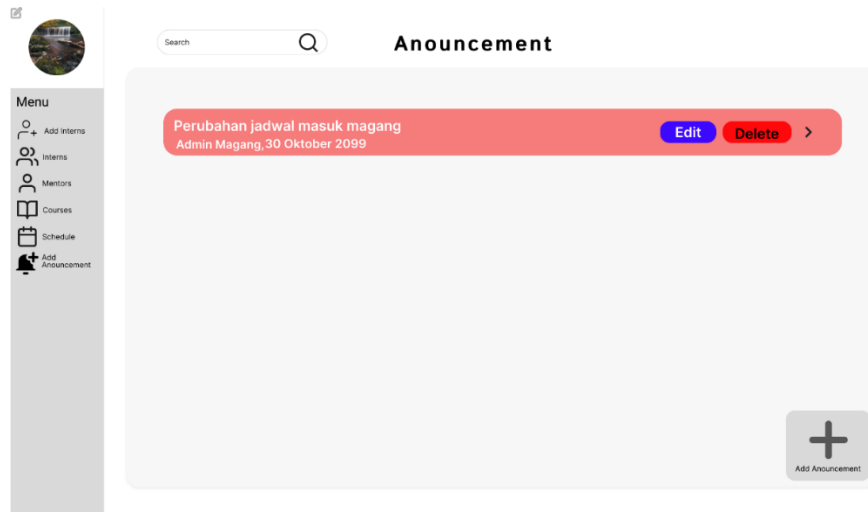
Gambar 3.32
Halaman Tambah Anak Magang

Pada gambar 3.32, admin bisa tambah anak Magang, sebagai opsi jika anak magang belum register pada website tapi sudah magang di perusahaan tersebut.



Gambar 3.33
Halaman Tambah Kursus

Pada gambar 3.33, admin bisa tambah kursus, yang harus dilakukan karena hanya admin yang bisa menambah kursus, dimana admin bisa juga menambahkan mentor dan anak magang masuk ke dalam kursus yang dibuat.



Gambar 3.34
Halaman Pengumuman Admin

Pada gambar 3.34, admin bisa buat pengumuman , seperti mentor pada gambar 3.23 halaman pengumuman mentor , dimana admin bisa mengumumkan mungkin nya ada event penting yang tidak ada pada kalender yang dibuat.

BAB IV

IMPLEMENTASI

Bab Implementasi akan memperjelaskan bagaimana proyek ini bekerja secara detail. Proyek ini akan dijelaskan secara detail dengan sumber kode ditunjukkan pada segmen subbab. Dari situ fitur-fitur utama proyek bisa dijelaskan.

4.1 Konfigurasi

Pada Segmen Program 4.1 menunjukkan bagaimana suatu aplikasi berbasis Nest.js yang merupakan sebuah framework dari Node.js memiliki konfigurasi Port nya. Berikut adalah Langkah-langkahnya:

Aplikasi dimulai dengan inisialisasi Express (`const app = express()`), yang merupakan framework untuk membangun server. Kemudian, sebuah server HTTP dibuat menggunakan `http.createServer(app)`, yang memungkinkan aplikasi untuk menangani permintaan HTTP.

Port server didefinisikan sebagai 3000, yang digunakan untuk mendengarkan koneksi dari klien. Selanjutnya, konfigurasi CORS (Cross-Origin Resource Sharing) ditentukan melalui objek `corsOptions`. Konfigurasi ini menetapkan asal yang diizinkan (`http://localhost:5173`), metode HTTP yang dapat digunakan (GET, POST, PUT, DELETE), dan header yang diizinkan (seperti Content-Type dan Authorization). Middleware CORS diterapkan menggunakan `app.use(cors(corsOptions))`, sehingga permintaan dari frontend dapat diterima.

Socket.IO dikonfigurasi dengan opsi CORS yang sama agar komunikasi real-time antara server dan klien dapat berjalan tanpa masalah lintas domain. Socket.IO kemudian dihubungkan ke aplikasi Express dengan metode `app.set('io', io)`.

Middleware tambahan diterapkan untuk menangani parsing body permintaan. `express.json()` digunakan untuk memproses data JSON, sementara `express.urlencoded()` digunakan untuk menangani data yang dikodekan dalam URL.

Rute aplikasi didefinisikan menggunakan `app.use('/api', api)`, yang menunjukkan bahwa semua rute API akan diakses melalui path `/api`.

Untuk integrasi Socket.IO, server mendengarkan koneksi baru melalui event `connection`. Ketika pengguna terhubung, pesan log dicetak. Server juga menangani event `newChat` untuk menerima pesan baru dari klien dan memancarkannya ke semua klien yang terhubung. Saat pengguna terputus, pesan log lainnya dicetak untuk menunjukkan bahwa koneksi telah ditutup.

Secara keseluruhan, kode ini memungkinkan server untuk menangani permintaan HTTP, komunikasi real-time dengan klien melalui Socket.IO, dan memastikan kompatibilitas lintas domain melalui CORS.

Segmen Program 4.1 Konfigurasi Code Untuk setup backend

```
01: Initialize Express application:
02: const app = express();
03: Create an HTTP server using the Express app:
04: const server = http.createServer(app);
05: Define the server port:
06: const port = 3000;
07: Configure CORS options:
08: const corsOptions = {
09:   origin: 'http://localhost:5173',
10:   methods: ['GET', 'POST', 'PUT', 'DELETE'],
11:   allowedHeaders: ['Content-Type', 'Authorization']
12: };
13: Apply CORS middleware with the defined options:
14: app.use(cors(corsOptions));
15: Configure Socket.IO with the same CORS options:
16: const io = socketIo(server, { cors: corsOptions });
17: Attach Socket.IO instance to the Express app:
18: app.set('io', io);
19: Apply middleware to parse incoming request bodies:
20: app.use(express.json()); // Parse JSON request bodies
```

Segmen Program 4.1 (lanjutan)

```
21: app.use(express.urlencoded({ extended: true
22: Define application routes:
23: app.use('/api', api); // Mount the API routes at '/api'
24: Handle Socket.IO events:
25: io.on('connection', (socket) => {
26:   console.log('A user connected'); // Log when a user connects
27:   socket.on('newChat', (data) => { // Listen for 'newChat' events
28:     console.log('New message:', data); // Log the received message
29:     io.emit('newChat', data);
30:   });
31:   socket.on('disconnect', () => { // Handle user disconnection
32:     console.log('A user disconnected'); // Log the disconnection
33:   });
34: });
```

Segmen program 4.2 Konfigurasi backend untuk sambung database dimulai dengan menghubungkan server ke database MongoDB menggunakan Mongoose. Perintah `mongoose.connect()` digunakan dengan URL koneksi database (`mongodb://localhost:27017/projectFPW`). Jika koneksi berhasil, pesan "Database connected" akan dicetak ke konsol. Namun, jika terjadi kesalahan selama koneksi, error tersebut akan ditangkap dan ditampilkan di konsol menggunakan `console.error`.

Setelah koneksi database berhasil, server HTTP diaktifkan dengan memanggil `server.listen(port)`, di mana `port` adalah variabel yang telah didefinisikan sebelumnya (misalnya 3000). Ketika server berhasil berjalan, pesan "Server is running on `http://localhost:port`" akan dicetak untuk memberikan informasi bahwa server siap menerima permintaan.

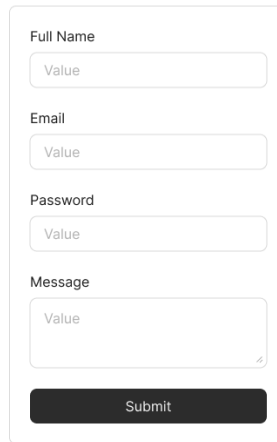
Untuk memastikan server tetap stabil meskipun terjadi error yang tidak terduga, dua mekanisme penanganan error diterapkan. Pertama, event `unhandledRejection` digunakan untuk menangani promise yang tidak tertangani. Ketika error seperti ini terjadi, server mencetak pesan error ke konsol. Kedua, event `uncaughtException`

menangkap error yang tidak terduga pada proses runtime. Pesan error dicetak, dan server secara eksplisit menghentikan proses dengan `process.exit(1)` untuk mencegah perilaku yang tidak diinginkan.

Segmen Program 4.2 Konfigurasi backend untuk sambung database

```
01: Connect to the MongoDB database:
02: mongoose.connect('mongodb://localhost:27017/projectFPW', {})
03:   .then(() => {
04:     console.log('Database connected');
05:   })
06:   .catch((e) => {
07:     console.error('Error connecting to the database:', e); //
Handle connection errors
08:   });
09: Start the HTTP server:
10: server.listen(port, () => {
11:   console.log(`Server is running on http://localhost:${port}`);
// Log server start message
12: });

13: Handle unhandled promise rejections:
14: process.on('unhandledRejection', (error) => {
15:   console.error('Unhandled Rejection:', error); // Log unhandled
promise rejection
16: });
17: Handle uncaught exceptions:
18: process.on('uncaughtException', (error) => {
19:   console.error('Uncaught Exception:', error);
20:   process.exit(1);
21: });
```

A user registration form with a light gray border. It contains four input fields, each with a label above it and a placeholder text 'Value'. The labels are 'Full Name', 'Email', 'Password', and 'Message'. The 'Message' field is a text area with a small cursor icon at the bottom right. Below the input fields is a dark gray 'Submit' button.

Gambar 4.1
Form Registrasi User

Segmen program 4.3 registrasi user baru dimulai dengan mendefinisikan schema validasi menggunakan Joi untuk memastikan data yang dikirim oleh klien sesuai dengan aturan yang ditentukan. Schema validasi registerSchema mencakup beberapa field: namaUser yang wajib diisi sebagai string, Profile_Picture yang dapat berupa string kosong atau null, roleType yang harus berupa angka 0, 1, atau 2, noTelpon yang wajib diisi sebagai string, email yang harus berupa format email yang valid, dan password yang harus memiliki panjang minimal 6 karakter. Selain itu, terdapat course yang bersifat opsional, dan asalSekolah yang hanya wajib diisi jika roleType bernilai 2 (menunjukkan bahwa pengguna adalah seorang "Anak Magang"). Penggunaan Joi.when memungkinkan validasi kondisional untuk field asalSekolah.

Pada route POST /register, server akan menangani permintaan pendaftaran. Pertama, data yang diterima dari req.body akan divalidasi menggunakan schema yang telah didefinisikan. Jika ada kesalahan validasi, server akan mengembalikan respon dengan status 400 beserta pesan error yang sesuai. Jika validasi berhasil, sistem akan memeriksa apakah email yang diajukan sudah terdaftar di database dengan melakukan

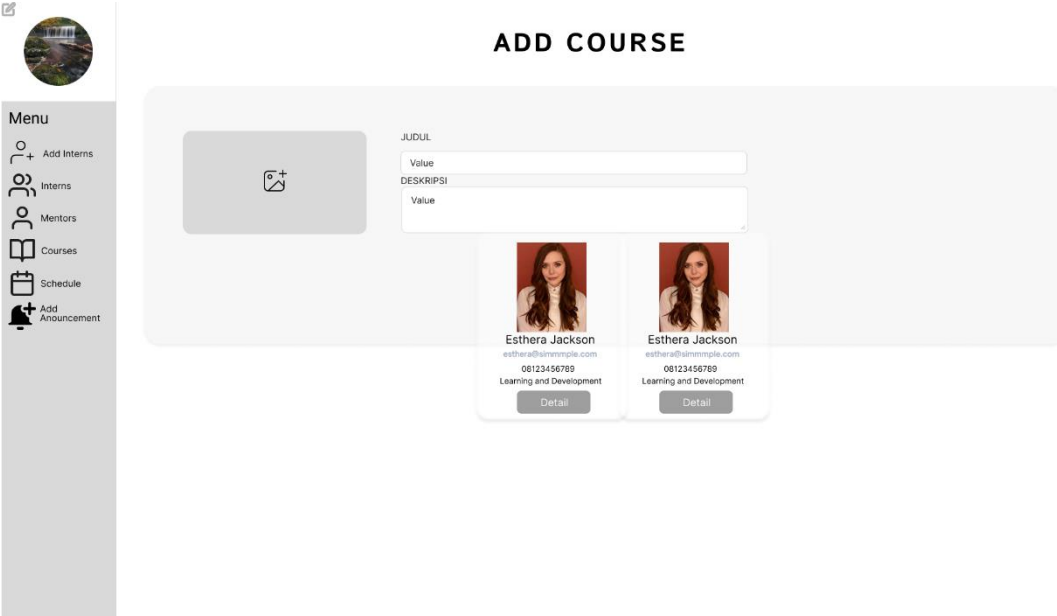
query `UserData.findOne({ email })`). Jika email sudah terdaftar, server akan mengembalikan pesan kesalahan. Selanjutnya, kode ini memeriksa jika pengguna memberikan nilai untuk course. Jika ada nilai, maka sistem akan memeriksa keberadaan course di database melalui query `Course.findOne({ namaCourse: course })`. Jika course yang dimaksud tidak ditemukan, server akan mengembalikan pesan kesalahan dengan status 400. Pada frontend gambar 4.1 berinteraksi dengan segmen program 4.3

Segmen Program 4.3 registrasi user baru

```
01: Define registration validation schema with Joi:
02: const registerSchema = Joi.object({
03:   namaUser: Joi.string().required(),
04:   Profile_Picture: Joi.string().allow(null, ''),
05:   roleType: Joi.number().valid(0, 1, 2).required(),
06:   noTelpon: Joi.string().required(),
07:   email: Joi.string().email().required(),
08:   password: Joi.string().min(6).required(),
09:   course: Joi.string().optional(),
10:   asalSekolah: Joi.string().when('roleType', {
11:     is: 2, // Only required when 'roleType' is 2 (Apprentice)
12:     then: Joi.required(),
13:     otherwise: Joi.optional(),
14:   }),
15: });
16: router.post('/register', async (req, res) => {
17:   try {
18:     const { namaUser, Profile_Picture, roleType, noTelpon, email,
password, course, asalSekolah } = req.body;
19:     console.log(req.body);
20:     const { error } = registerSchema.validate(req.body);
21:     if (error) {
22:       console.error('Validation error:', error.details[0].message);
23:       return res.status(400).json({ message: error.details[0].message });
24:       const existingUser = await UserData.findOne({ email });
```

Segmen Program 4.3 (lanjutan)

```
25:     if (existingUser) {
26:         console.warn('Email already registered:', email);
27:         return res.status(400).json({ message: 'Emailregistered.' });
28:     }
29:     let courseExist;
30:     if (course && course !== '') {
31:         courseExist = await Course.findOne({ namaCourse: course });
32:         if (!courseExist) {
33:             return res.status(400).json({ message: 'Course not found' });
34:         }
35:     }
36: } catch (error) {
37:     return res.status(500).json({ message: 'Server error' });
38: }
39: });
```



Gambar 4.2

Form Kursus Baru

Segmen program 4.4 buat pengumuman berinteraksi dengan Gambar 4.2 form kursus baru mendefinisikan route POST pada endpoint /Course untuk membuat kursus

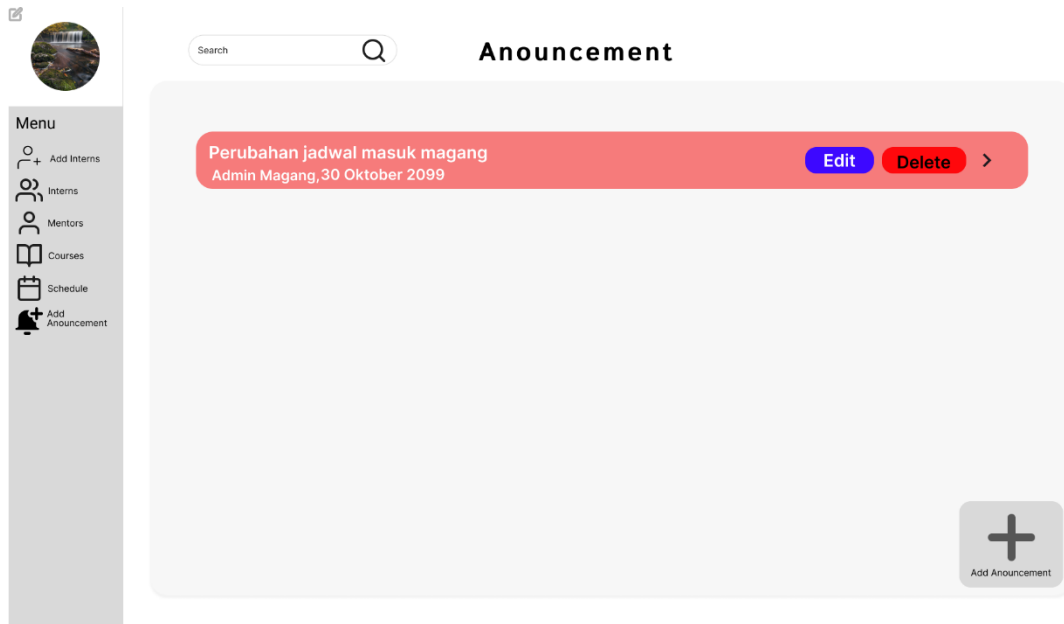
baru dalam sistem. Di dalam route handler, pertama-tama, data seperti namaCourse, Deskripsi, mentorID, dan daftarKelas diambil dari body permintaan (req.body). Data ini digunakan untuk membuat objek baru dari model Course.

Objek newCourse yang baru dibuat berisi informasi yang diterima dari body permintaan, di mana mentorID dan daftarKelas memiliki nilai default berupa array kosong jika tidak ada data yang diberikan dalam permintaan. Setelah itu, objek newCourse disimpan dalam database menggunakan await newCourse.save().

Jika kursus berhasil disimpan, respons dengan status HTTP 201 (created) dikirim kembali bersama dengan data kursus yang baru disimpan dalam format JSON. Jika terjadi kesalahan saat pembuatan kursus (misalnya masalah dalam menyimpan ke database).

Segmen Program 4.4 buat kursus baru

```
01: Define POST '/Course' route handler:
02: router.post("/Course", async (req, res) => {
03:   try {
04:     const { namaCourse, Deskripsi, mentorID, daftarKelas } =
req.body;
05:     const newCourse = new Course({
06:       namaCourse,
07:       Deskripsi,
08:       mentorID: mentorID || [],
09:       daftarKelas: daftarKelas || []
10:     });
11:     const savedCourse = await newCourse.save();
12:     res.status(201).json(savedCourse);
13:   } catch (error) {
14:     res.status(500).json({ message: "Error creating course", error
});
```



Gambar 4.3

Halaman Add Announcement

Segmen program 4.5 buat pengumuman dengan gambar 4.3 ini mendefinisikan route POST pada endpoint `/announcement` untuk membuat pengumuman baru yang dapat mencakup lampiran file. Pertama, route ini dilindungi oleh middleware `verifyToken([0])`, yang memastikan bahwa hanya pengguna dengan role tertentu yang dapat mengaksesnya (misalnya, administrator). Middleware `upload.fields([{ name: 'attachments', maxCount: 5 }])` digunakan untuk menangani file yang diunggah, memungkinkan pengguna mengunggah hingga lima file sebagai lampiran untuk pengumuman.

Di dalam handler route, data seperti title, description, dan `createdBy` diambil dari body permintaan. Jika ada file yang diunggah, file tersebut diambil dari `req.files.attachments`, dan jika tidak ada file, akan diinisialisasi sebagai array kosong. Setiap file lampiran diproses untuk membuat objek yang mencakup informasi seperti nama file (`originalname`), path file di server (`path`), tipe file (`mimetype`), ukuran file (`size`), dan tanggal unggah.

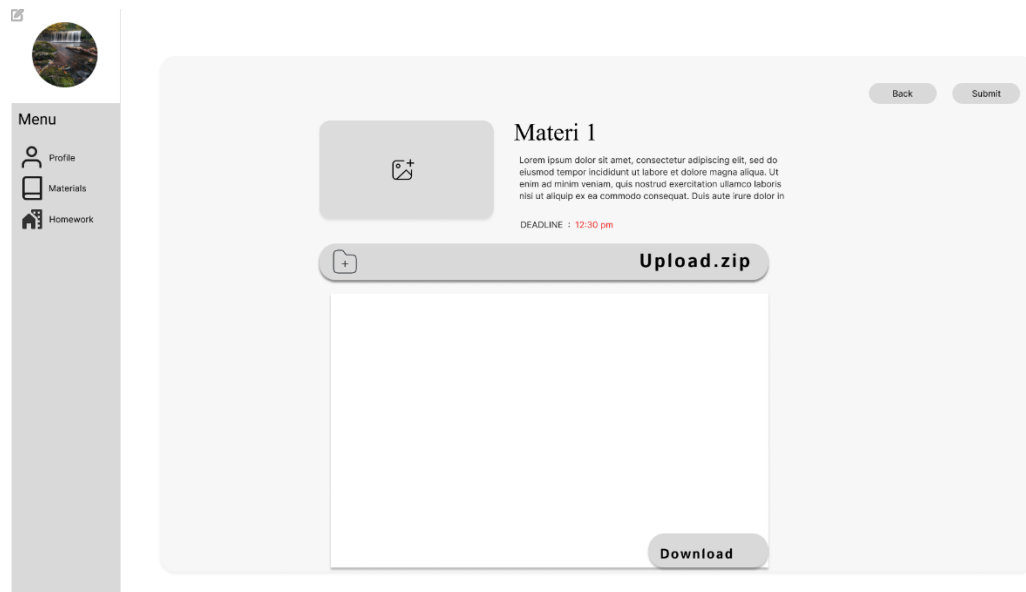
Setelah itu, objek pengumuman baru dibuat menggunakan data yang diambil dari body permintaan dan informasi file yang diunggah. Pengumuman baru ini kemudian disimpan dalam database dengan await newAnnouncement.save(). Jika pengumuman berhasil disimpan, respons dengan status 201 (created) dikirim bersama pesan sukses dan data pengumuman yang baru dibuat. Jika terjadi kesalahan dalam proses pembuatan pengumuman, kesalahan dicatat di log dan respons dengan status 500 (internal server error) dikirim, menyertakan pesan kesalahan.

Segmen Program 4.5 buat pengumuman

```
01: Define POST '/announcement' route handler:
02: router.post("/announcement", verifyToken([0]), upload.fields([
03:   { name: 'attachments', maxCount: 5 }
04: ]), async (req, res) => {
05:   try {
06:     const { title, description, createdBy } = req.body;
07:     const attachments = req.files.attachments || [];
08:     const newAnnouncement = new Announcement({
09:       title,
10:       description,
11:       createdBy,
12:       attachments: attachments.map(file => ({
13:         fileName: file.originalname,
14:         filePath: file.path,
15:         fileType: file.mimetype,
16:         fileSize: file.size,
17:         uploadDate: new Date(),
18:       })),
19:       date: new Date(),
20:     });
21:     await newAnnouncement.save();
22:     res.status(201).json({
23:       message: 'Announcement created successfully.',
```

Segmen Program 4.5 (lanjutan)

```
24:     announcement: newAnnouncement,  
25:   });  
26: } catch (err) {  
27:   console.error('Error creating announcement:', err);  
28:   res.status(500).json({  
29:     message: 'Failed to create announcement.',  
30:     error: err.message,  
31:   });  
32: }  
33: });
```



Gambar 4.3

Halaman UploadJawaban

Segmen program 4.6 jawab soal/uploadjawaban bersangkutan dengan gambar 4.3 mendefinisikan route POST pada endpoint /Jawaban untuk menangani pengiriman jawaban dari anak magang terhadap soal modul. Pertama, data yang diterima melalui body permintaan diambil, termasuk anakMagangID, courseID, soalModulID, jawaban, dan jawabanType. Jika ada file yang diunggah, file tersebut diproses dan disimpan dalam objek uploadJawaban, yang berisi informasi seperti nama file, path, tipe file, dan

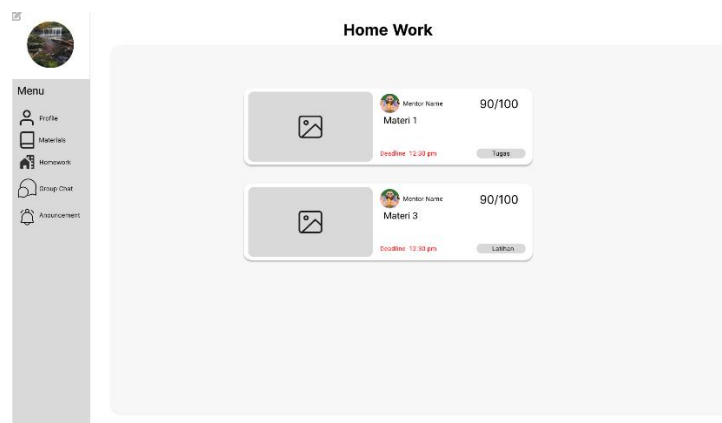
tanggal unggah. Setelah itu, dilakukan validasi menggunakan schema Joi, yang memastikan bahwa ID untuk course, anak magang, dan soal modul valid, serta bahwa jawabanType adalah salah satu dari dua opsi yang valid: "essay" atau "file".

Segmen Program 4.6 jawab soal/upload jawaban

```
01: Define POST '/Jawaban' route handler:
02: router.post("/Jawaban", verifyToken([2]), upload.single("uploadJawaban"), async (req, res) => {
03:   const { anakMagangID, courseID, soalModulID, jawaban, jawabanType
    } = req.body;
04:   const uploadJawaban = req.file ? {
05:     fileName: req.file.filename,
06:     filePath: req.file.path,
07:     fileType: req.file.mimetype,
08:     uploadDate: new Date(),
09:   } : null;
10:
11:   Define Joi validation schema for 'Jawaban' data:
12:   schema = Joi.object({
13:     courseID: Joi.string().custom(checkIdValid).required(),
14:     anakMagangID: Joi.string().custom(checkIdValid).required(),
15:     soalModulID: Joi.string().custom(checkIdValid).required(),
16:     jawabanType: Joi.string().required().valid("essay", "file"),
17:   })
18:   .when(Joi.object({ jawabanType: "essay" }).unknown(), {
19:     then: Joi.object({
20:       jawaban: Joi.string().required(),
21:     }),
22:   })
23:   .when(Joi.object({ jawabanType: "file" }).unknown(), {
24:     then: Joi.object({
25:       uploadJawaban: Joi.object({
26:         fileName: Joi.string().required(),
```

Segmen Program 4.6 (lanjutan)

```
27:         filePath: Joi.string().required(),
28:         fileType: Joi.string().required(),
29:         uploadDate: Joi.date().iso().required(),
30:     }).required(),
31:   }},
32: });
33:
34:   Validate the request body against the schema:
35:   const { error } = schema.validate({ ...req.body, uploadJawaban
36:   });
37:   if (error) {
38:     return res.status(400).json({ message:
39:     error.details[0].message });
40:   }
41:   Try to process the request:
42:   const findCourse = await checkIdExist(Course, courseID);
43:   const findAnakMagang = await checkIdExist(AnakMagang,
44:   anakMagangID);
45:   const findSoal = await checkIdExist(SoalModul, soalModulID);
46:   if (!findCourse && !findAnakMagang && !findSoal) {
```



Gambar 4.4
Halaman homework

Segmen program 4.7 melanjutkan 4.6 dan bersangkutan dengan gambar 4.4 dimana Validasi lebih lanjut memastikan bahwa jika jenis jawaban adalah "essay", maka field jawaban harus ada, dan jika jenis jawaban adalah "file", maka file yang diunggah harus disertakan. Jika terjadi kesalahan dalam validasi, server mengirimkan respons error dengan status 400.

Setelah validasi berhasil, dilakukan pencarian untuk memastikan bahwa courseID, anakMagangID, dan soalModulID ada dalam database. Jika salah satu dari ID tersebut tidak ditemukan, server mengirimkan respons error dengan status 404. Jika semua ID valid, objek jawaban baru (JawabanModul) dibuat dengan data yang diterima, dan jawaban disimpan ke dalam database. Terakhir, respons sukses dengan status 201 dikirimkan bersama dengan jawaban yang baru disimpan. Jika terjadi kesalahan dalam proses, pesan kesalahan dicatat dan respons dengan status 500 dikirimkan.

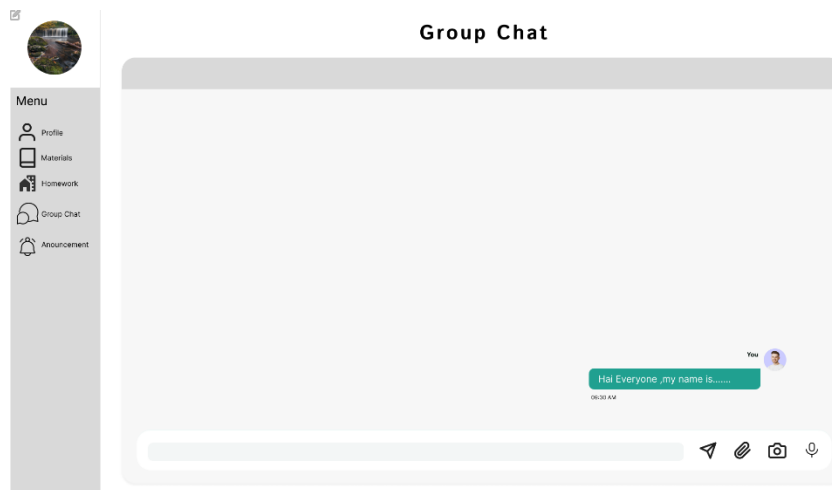
Segmen Program 4.7 jawab soal/upload bagian ke 2

```
45:     return res.status(404).json({ message: 'id from soal anakMagang
or course not found' });
46:   }
47:
48:   Create new JawabanModul:
49:   const newJawaban = new JawabanModul({
50:     courseID,
51:     anakMagangID,
52:     soalModulID,
53:     jawabanType,
54:     jawaban: jawabanType === "essay" ? jawaban : null,
55:     uploadJawaban: jawabanType === "file" ? uploadJawaban : null,
56:   });
57:
58:   Save the JawabanModul:
59:   const savedJawaban = await newJawaban.save();
60:
61:   Send success response:
```

```

62:   res.status(201).json(savedJawaban);
63: } catch (err) {
64:   Log error and send failure response:
65:   console.error("Error creating jawaban:", err);
66:   res.status(500).json({ message: "Internal server error" });
67: }
68: });

```



Gambar 4.5

Halaman chat

Segmen program 4.8 chat mendefinisikan route POST pada endpoint / untuk menangani pengiriman pesan chat. Pertama, data dari body permintaan diambil, yaitu senderID, courseID, content, dan attachments. Kemudian, dilakukan validasi terhadap senderID dan courseID untuk memastikan bahwa keduanya adalah ObjectId yang valid. Jika ada masalah dengan ID yang diberikan, respons error dengan status 400 dan pesan kesalahan akan dikirimkan. Jika validasi berhasil, objek pesan chat baru dibuat menggunakan data yang diterima, dan tanggal/waktu chat ditambahkan secara otomatis. Setelah itu, objek chat ini disimpan ke dalam database. Selanjutnya, instance WebSocket (io) diambil dari aplikasi dan dicek apakah tersedia. Jika io tersedia, pesan chat baru akan dipancarkan (broadcast) ke semua klien yang terhubung melalui WebSocket. Terakhir, jika semua langkah berhasil, respons sukses dengan status 201

dan pesan chat yang baru dibuat dikirimkan. Jika terjadi kesalahan di sepanjang proses, error dicatat dan respons kegagalan dengan status 500 dikirimkan kepada klien.

Segmen Program 4.8 chat

```
01: Define POST '/' route handler:
02: router.post('/', async (req, res) => {
03:   Try to process the request:
04:   const { senderID, courseID, content, attachments } = req.body;
05:   Log the received request body:
06:   console.log('Received request body:', req.body);
07:
08:   Validate senderID and courseID as valid ObjectIds:
09:   if (!isValidObjectId(senderID) || !isValidObjectId(courseID)) {
10:     return res.status(400).json({ error: 'Invalid senderID or
courseID' });
11:   }
12:
13:   Create a new Chat message object:
14:   const newChat = new Chat({
15:     senderID,
16:     courseID,
17:     content,
18:     attachments,
19:     chatDate: new Date(), // Set the current date and time
20:   });
21:   Log the new chat message:
22:   console.log('New chat message:', newChat);
23:
24:   Save the new chat message to the database:
25:   await newChat.save();
26:
27:   Access WebSocket io instance from the app:
28:   const io = req.app.get('io');
29:   Log the WebSocket io instance:
30:   console.log('WebSocket io instance:', io);
```

Segmen Program 4.8 (lanjutan)

```
31:
32:   Check if io is available:
33:   if (!io) {
34:       throw new Error('WebSocket io instance is not available');
35:   }
36:
37:   Emit the new chat message to all connected clients via WebSocket:
38:   io.emit('newChat', newChat); // Broadcast the new chat message
to all clients
39:
40:   Send success response with the new chat message:
41:   res.status(201).json({ message: 'Chat created successfully',
newChat });
42: } catch (error) {
43:   Log the error and send failure response:
44:   console.error('Error sending chat message:', error);
45:   res.status(500).json({ error: 'Error sending chat message' });
46: }
```


Gambar 4.6

Halaman buat modul

Segmen program 4.9 buat modul dan bersangkutan dengan gambar 4.6, mendefinisikan route POST pada endpoint /Modul untuk menangani pembuatan modul baru. Pertama, middleware `verifyToken([1])` digunakan untuk memastikan bahwa hanya pengguna dengan peran tertentu yang dapat mengakses endpoint ini. Data dari body permintaan seperti nama modul (`name`), deskripsi modul (`desc`), ID kursus (`courseID`), ID soal (`soalID`), dan batas waktu (`deadline`) diambil. Selanjutnya, schema validasi menggunakan Joi memastikan bahwa data yang diterima sesuai dengan format yang diinginkan. Jika array `soalID` diberikan, maka validasi dilakukan pada ID soal tersebut. Setelah validasi berhasil, server mencari kursus berdasarkan `courseID`, dan jika kursus ditemukan, sebuah objek Modul baru dibuat dengan data yang diterima. Modul ini disimpan di database, dan respons sukses dengan modul yang baru disimpan dikirimkan ke klien. Jika terjadi kesalahan dalam proses, server akan mengembalikan respons error dengan pesan yang sesuai.

Segmen Program 4.9 buat modul

```
01: Define POST '/Modul' route handler:
02: router.post("/Modul", verifyToken([1]), async (req, res) => {
03:   Extract data from request body:
04:   const { name, desc, courseID, soalID, deadline } = req.body;
05:
06:   Define Joi validation schema:
07:   const schema = Joi.object({
08:     name: Joi.string().required(),
09:     desc: Joi.string().optional().allow(""),
10:     courseID: Joi.string().custom(checkIdValid, "ObjectId
validation").required(),
11:     soalID: Joi.array().items(Joi.string()).optional(),
12:     deadline: Joi.date().iso().min(new Date(Date.now() + 60 * 60
* 1000)).required()
13:   });
14:
15:   Validate request body with schema:
16:   const { error } = schema.validate({ ...req.body });
17:   if (error) {
18:     Return error response with validation message:

19: return res.status(400).json({ message: error.details[0].message});
20:   }
21:   Validate soalID array if provided:
22:   let soalIDs;
23:   if (soalID && soalID !== []) {
24:     soalIDs = validateArrayOfIDs(Modul, soalID, "Modul");
25:   }
26:   Find course by courseID:
27:   const findCourse = await Course.findById(courseID);
28:   if (!findCourse) {
29:     Return error response if course not found:
30:     return res.status(404).json({ message: "course not found" });
31:   }
32:   Create a new Modul object:
33:   const newModul = new Modul({
34:     namaModul: name,
35:     Deskripsi: desc,
36:     courseID: findCourse,
37:     soalID: soalIDs || [],
38:     Deadline: deadline,
39:     Dinilai: false,
40:   });
41:
42:   Save the new Modul to the database:
43:   const savedModul = await newModul.save();
44:
45:   Return success response with saved Modul:
46:   res.status(201).json(savedModul);
```

Segmen Program 4.9 (lanjutan)

```
47: } catch (err) {  
48:   Log error and return failure response:  
49:   console.error("Error creating course:", err);  
50:   res.status(500).json({ message: "Internal server error" });
```

Gambar 4.7

Halaman Buat Soal

Segmen program 4.10 buat soal dan bersangkutan dengan gambar 4.7, mendefinisikan mendefinisikan route POST pada endpoint /Soal untuk menangani pembuatan entri soal baru. Pertama, middleware `verifyToken([1])` digunakan untuk memverifikasi token dan memastikan hanya pengguna dengan role tertentu yang dapat mengaksesnya. Selain itu, file soal dapat diunggah menggunakan middleware `upload.single("uploadSoal")`. Data soal seperti nama soal (`name`), deskripsi (`desc`), tipe soal (`SoalType`), pilihan jawaban (`Pilihan`), dan kunci jawaban (`kunciJawaban`) diambil dari body permintaan.

Segmen Program 4.10 buat soal

```
01: Define POST '/Soal' route handler:  
02:   router.post("/Soal", verifyToken([1]),  
upload.single("uploadSoal"), async (req, res) => {  
03:     Extract data from request body:  
04:     const { name, desc, SoalType, Pilihan, kunciJawaban } = req.body;  
05:  
06:     Process file upload if present:  
07:     const uploadSoal = req.file
```

Segmen Program 4.10 (lanjutan)

```
08:   ? {
09:     fileName: req.file.filename,
10:     filePath: req.file.path,
11:     fileType: req.file.mimetype,
12:     uploadDate: new Date(),
13:   }
14:   : null;
15:   Define Joi validation schema:
16:   const schema = Joi.object({
17:     name: Joi.string().required(),
18:     desc: Joi.string().optional().allow(""),
19:     SoalType: Joi.number().required().valid(0, 1, 2),
20:     Pilihan: Joi.array().items(Joi.string().required()).length(4).optional(),
21:     kunciJawaban: Joi.number().optional().min(0).max(3),
22:     uploadSoal: Joi.object({
23:       fileName: Joi.string().required(),
24:       filePath: Joi.string().required(),
25:       fileType: Joi.string().required(),
26:       uploadDate: Joi.date().iso().required(),
27:     }).optional(),
28:   })
29:   .when(Joi.object({ SoalType: 0 }).unknown(), {
30:     then: Joi.object({
31:                                                                 Pilihan:
Joi.array().items(Joi.string().required()).length(4).required(),
32:       kunciJawaban: Joi.number().required().min(0).max(3),
33:     }),
34:   })
35:   .when(Joi.object({ SoalType: Joi.valid(1, 2) }).unknown(), {
36:     then: Joi.object({
37:       uploadSoal: Joi.object({
38:         fileName: Joi.string().required(),
39:         filePath: Joi.string().required(),
40:         fileType: Joi.string().required(),
41:         uploadDate: Joi.date().iso().required(),
42:       }).required(),
43:     }),
44:   });
45:   Validate request body and file:
46:   const { error } = schema.validate({ ...req.body, uploadSoal });
47:   if (error) {
48:     Return error response with validation message:
49:     return res.status(400).json({ message:
error.details[0].message });
50:   }
```

Segmen program 4.10 melanjutkan 4.11, yang dilanjutkan pada skenario Jika ada file yang diunggah, informasi terkait file tersebut (nama, path, tipe, dan tanggal

unggah) disimpan dalam objek `uploadSoal`. Selanjutnya, schema validasi menggunakan Joi memastikan bahwa data yang diterima sesuai format yang diinginkan, dengan syarat khusus untuk setiap tipe soal. Setelah data berhasil divalidasi, objek `SoalModul` baru dibuat dan disimpan ke database. Jika berhasil, server akan mengembalikan respons dengan data soal yang baru disimpan. Jika terjadi kesalahan, server akan mengembalikan respons error dengan pesan yang sesuai.

Segmen Program 4.11 buat soal bagian ke 2

```
51:   Try to process the request:
52:   Create a new SoalModul document:
53:   const newSoal = new SoalModul({
54:     namaSoal: name,
55:     Deskripsi: desc,
56:     Gambar: uploadSoal, // File data should be passed here as an
object
57:     SoalType: SoalType,
58:     Pilihan: Pilihan,
59:     kunciJawaban: kunciJawaban,
60:   });
61:   Save the new SoalModul to the database:
62:   const savedSoal = await newSoal.save();
63:   Return success response with saved SoalModul:
64:   res.status(201).json(savedSoal);
65: } catch (err) {
66:   Log error and return failure response:
67:   console.error("Error creating soal:", err);
68:   res.status(500).json({ message: "Internal server error" });
69: }
```

| ID | Name & Email | Phone Number | Course | Action |
|-----------|--|--------------|--------------------------|--------|
| 123456789 | Karthika Jackson karthika@bismillah.com | 08123456789 | Learning and Development | ✓ |
| 123456789 | Karthika Jackson karthika@bismillah.com | 08123456789 | Learning and Development | ✓ |
| 123456789 | Karthika Jackson karthika@bismillah.com | 08123456789 | Learning and Development | ✓ |
| 123456789 | Karthika Jackson karthika@bismillah.com | 08123456789 | Learning and Development | ✓ |
| 123456789 | Karthika Jackson karthika@bismillah.com | 08123456789 | Learning and Development | ✓ |
| 123456789 | Karthika Jackson karthika@bismillah.com | 08123456789 | Learning and Development | ✓ |
| 123456789 | Karthika Jackson karthika@bismillah.com | 08123456789 | Learning and Development | ✓ |
| 123456789 | Karthika Jackson karthika@bismillah.com | 08123456789 | Learning and Development | ✓ |

Gambar 4.7
Halaman Absensi

Segmen program 4.12 absensi dengan gambar 4.7 halaman absensi dimulai dengan mengambil data (courseID, mentorID, anakMagangID) dari body permintaan. Kemudian, sebuah schema validasi Joi digunakan untuk memastikan bahwa ID yang diberikan memiliki format yang benar dan tidak kosong. Jika validasi gagal, error 400 dikembalikan dengan pesan yang sesuai.

Segmen Program 4.12 absensi

```

01: Define POST '/absensi' route handler:
02: router.post('/absensi', verifyToken([1]), async (req, res) => {
03:   Extract data from request body:
04:   const { courseID, mentorID, anakMagangID } = req.body;
05:   Define Joi validation schema:
06:   const schema = Joi.object({
07:     courseID: Joi.string().required().custom(checkIdValid, "object
validation").messages({
08:       'string.base': 'course ID must be a string.',
09:       'string.empty': 'course ID cannot be empty.',
10:       'any.required': 'course ID is required.',
11:     }),
12:     mentorID: Joi.string().required().custom(checkIdValid, "object
validation").messages({
13:       'string.base': 'Mentor ID must be a string.',
14:       'string.empty': 'Mentor ID cannot be empty.',
15:       'any.required': 'Mentor ID is required.',

```

Segmen Program 4.12 (lanjutan)

```
16:     }),
17:                                     anakMagangID:
Joi.array().items(Joi.string().required().messages({
18:     'string.base': 'Each Anak Magang ID must be a string.',
19:     'string.empty': 'An Anak Magang ID cannot be empty.',
20:     'any.required': 'An Anak Magang ID is required.',
21:   })).required().messages({
22:     'array.base': 'Anak Magang IDs must be an array.',
```

Segmen Program 4.12 (lanjutan)

```
23:     'array.includesRequiredUnknowns': 'Anak Magang IDs must
contain valid strings.',
24:     'any.required': 'Anak Magang IDs are required.',
25:   })),
26: });
27: Validate request body using Joi:
28: const { error, value } = schema.validate(req.body);
29: if (error) {
30:   Return error response with validation message:
31:   return res.status(400).json({ message:
error.details[0].message });
32: }
33: Try to process the request:
34: Check if the course exists:
```

Segmen program 4.13 melanjutkan 4.12, yang terjadi setelah validasi, kode memeriksa apakah kursus dan mentor ada di database menggunakan `findById` dan mengembalikan error jika tidak ditemukan. Kode juga memvalidasi bahwa kursus tersebut terkait dengan mentor, dengan mengiterasi `courseID` milik mentor untuk memeriksa apakah `courseID` yang diberikan valid. Jika tidak ada kecocokan, error 404 dikembalikan.

Selanjutnya, kode memvalidasi array `anakMagangID` dan mencatat ID yang sudah divalidasi. Tanggal saat ini disimpan dan digunakan untuk memperbarui setiap dokumen `AnakMagang` yang sesuai dengan menambahkan tanggal saat ini ke field `absensi` (`absensiKelas`). Setiap dokumen `AnakMagang` yang diperbarui kemudian disimpan.

Terakhir, sebuah dokumen `Absensi` baru dibuat untuk mencatat absensi berdasarkan kursus yang diberikan dan array `anakMagangID` yang sudah divalidasi. Dokumen absensi baru ini disimpan ke database. Setelah berhasil, respons 200 dengan

pesan sukses dan data absensi baru dikirimkan kembali. Jika terjadi error selama proses, error tersebut akan tertangkap, dicatat, dan respons error 500 dikembalikan yang menunjukkan adanya kesalahan di server.

Segmen Program 4.13 absensi bagian ke 2

```
35:   const coursefind = await Course.findById(courseID);
36:   if (!coursefind) {
37:     Return error if course is not found:
38:     return res.status(403).json({ message: 'course not found in
the database' });
39:   }
40:   Check if the mentor exists:
41:   const mentorUser = await Mentor.findById(mentorID);
42:   if (!mentorUser) {
43:     Return error if mentor is not found:
44:     return res.status(403).json({ message: 'Mentor not found in
the database' });
45:   }
46:   Validate mentor's course association:
47:   const checkCourseMentor = mentorUser.courseID;
48:   Initialize flag `find` as false:
49:   for (const id of checkCourseMentor) {
50:     If courseID matches mentor's courseID, set `find` to true:
51:     if (id == coursefind.id) {
52:       find = true;
53:     }
54:   }
55:   If no match, return error:
56:   if (!find) {
57:     return res.status(404).json({ message: 'courseId not found in
mentor' });
58:   }
59:   Validate anakMagangID array:

60:   const validatedIds = await validateArrayOfIDs (AnakMagang,
anakMagangID, "anakMagang");
61:   Log validated IDs:
62:   console.log(validatedIds);
```


Segmen Program 4.13 (lanjutan)

```
63:   Get today's date:
64:   const today = new Date();
65:   Update attendance for each AnakMagang:
66:   for (const id of validatedIds) {
67:     Find each AnakMagang document:
68:     const anakMagangData = await AnakMagang.findById(id);
69:     Add today's date to absensiKelas:
70:     anakMagangData.absensiKelas.push(today);
71:     Save updated document:
72:     await anakMagangData.save();
73:   }
74:   Create new Absensi document:
75:   const absenToday = new Absensi({
76:     courseID: coursefind.id,
77:     tanggalAbsensi: today,
78:     absensiKelas: validatedIds,
79:   });
80:   Save new Absensi document:
81:   const newAbsenRes = await absenToday.save();
82:   Return success response with attendance data:
83:   res.status(200).json({
84:     message: 'Attendance successfully recorded',
85:     newAbsenRes,
86:   });
87: } catch (error) {
88:   Log error and return failure response:
89:   console.error('Error processing attendance:', error);
90:   res.status(500).json({ message: 'An error occurred on the
server.' });
```

BAB V

TESTING

Bab Testing akan menampilkan hasil outputnya yang diekspektasikan dan hasil sebenarnya. Testing akan dilakukan untuk mencegah adanya kecacatan program dengan diidentifikasi dulu dengan testing agar bisa segera dibenarkan, dengan testing program yang ditemukan bugnya bisa dibenarkan agar Ketika selesai program tidak ada error Ketika dipakai secara umum.

5.1 List Test Case

Pada subbab ini akan ada beberapa list yang berisi test case sebagai percobaan hasil program pada proyek ini. Yaitu user data, admin, anak magang, mentor, chat, announcement dan modul.

1. user data

Pada test case ini, akan berhubungan dengan user data, di mana sistem akan memverifikasi email dan password pengguna untuk memastikan bahwa hanya pengguna yang terdaftar dan memiliki kredensial yang valid dapat login untuk mengakses website ini. Proses validasi ini juga mencakup pemeriksaan status akun, seperti apakah akun telah diverifikasi oleh admin atau masih dalam status tidak aktif.

| Id test | Deskripsi Testcase | Langkah uji | Hasil ekspektasi | Hasil asli | status |
|----------------|---------------------------|---|---|--|---------------|
| 1 | Login Berhasil | 1. Pada halaman login, Inputkan email dan password 2. Tekan tombol Login | User akan berhasil login dan dipindahkan ke halaman home user | User akan berhasil login dan dipindahkan ke halaman home user sesuai dengan role | Pass |

| Id test | Deskripsi Testcase | Langkah uji | Hasil ekspektasi | Hasil asli | status |
|----------------|--|---|--|--|---------------|
| | | | sesuai dengan role | | |
| 2 | Login gagal dari input kosong | 1. Pada halaman login, Input kosong 2. Tekan tombol Login | login Error karena input kosong | login Error karena input kosong | Pass |
| 3 | Login gagal email invalid dan password invalid | 1. Pada halaman login, Input email invalid 2. Tekan tombol Login | login error email or password invalid | login error email or password invalid | Pass |
| 4 | Login gagal belum diVerify oleh admin | 1. Pada halaman login, Inputkan email dan password 2. Tekan tombol Login | login error user belum ke verify | login error user belum ke verify | pass |
| 5 | register berhasil | 1. pada halaman login tekan tombol register 2. input field nama , password , nomor telpon email. 3. tekan tombol register | register berhasil user masuk ke dalam database | register berhasil user masuk ke dalam database | pass |
| 6 | register gagal invalid email, password | 1. pada halaman login tekan tombol register 2. input field email dan password invalid | register error input email invalid/ password minimal 6 | register error input email invalid/ password minimal 6 | pass |

| Id test | Deskripsi Testcase | Langkah uji | Hasil ekspektasi | Hasil asli | status |
|----------------|---------------------------|--------------------------|-------------------------|-------------------|---------------|
| | | 3. tekan tombol register | | | |

Tabel 5.1
Test case UserData

2. admin

Pada test case ini, akan berhubungan dengan peran admin, di mana admin memiliki akses penuh untuk mengelola berbagai fungsi penting dalam sistem. Admin dapat membuat kursus baru, menambahkan pengumuman, dan mengedit detail pengguna, termasuk anak magang dan mentor. Peran ini memungkinkan admin untuk memastikan semua informasi dalam sistem tetap terorganisir dan diperbarui. Selain itu, admin juga bertanggung jawab dalam memonitor aktivitas pengguna serta memastikan bahwa semua modul dan fitur dapat berjalan sesuai kebutuhan perusahaan. Berikut adalah detail test case yang melibatkan peran admin:

| Id test | Deskripsi Testcase | Langkah uji | Hasil ekspektasi | Hasil asli | status |
|----------------|---------------------------|--|--------------------------|--------------------------|---------------|
| 1 | buat kursus baru | 1. Pada halaman home admin, tekan tombol courses 2. Tekan tombol plus 3. input nama, desc 4. tekan tombol add | course baru akan dibuat | course baru akan dibuat | Pass |
| 2 | buat pengumuman | 1. Pada halaman home admin, tekan tombol add announcement 2. input nama dan desc,lalu file 3. tekan tombol add | announcement baru dibuat | announcement baru dibuat | Pass |
| 3 | edit detail anak magang | 1. Pada halaman home admin, | anak magang teredit | anak magang teredit | Pass |

| Id test | Deskripsi Testcase | Langkah uji | Hasil ekspektasi | Hasil asli | status |
|----------------|-------------------------------------|---|-----------------------------------|---|---------------|
| | | 2. tekan tombol interns lalu edit interns 3. input field yang ingin diubah 4. tekan tombol edit | | | |
| 4 | buat kursus baru gagal input kosong | 1. Pada halaman home admin, tekan tombol courses 2. Tekan tombol plus input nama, desc kosong 3. tekan tombol add | course error input kosong | course error input kosong | Pass |
| 5 | buat pengumuman gagal input kosong | 1. Pada halaman home admin, tekan tombol add announcement 2. input nama dan desc,lalu file kosong 3. tekan tombol add | announcement error tidak ada file | announcement error tidak ada attachment | Pass |

Tabel 5.2
Test case Admin

3. mentor

Pada test case ini, akan berhubungan dengan userdata, di mana sistem akan memverifikasi email dan password pengguna untuk memastikan bahwa hanya pengguna yang terdaftar dan memiliki kredensial yang valid dapat login untuk mengakses website ini. Proses validasi ini juga mencakup pemeriksaan status akun, seperti apakah akun telah diverifikasi oleh admin atau masih dalam status tidak aktif.

| Id tes t | Deskripsi Testcase | Langkah uji | Hasil ekspektasi | Hasil asli | statu s |
|-------------------------|-------------------------------|--|-----------------------------|--------------------------|--------------------|
| 1 | buat kursus baru | 1. Pada halaman home admin, tekan tombol courses 2. Tekan tombol plus 3. input nama, desc 4. tekan tombol add | course baru akan dibuat | course baru akan dibuat | Pass |
| 2 | buat pengumuman | 1. Pada halaman home admin, tekan tombol add announcement 2. input nama dan desc,lalu file 3. tekan tombol add | announcement baru dibuat | announcement baru dibuat | Pass |
| 3 | edit detail anak magang | 1. Pada halaman home admin, 2. tekan tombol interns lalu edit interns 3. input field yang ingin diubah 4. tekan tombol edit | anak magang teredit | anak magang teredit | Pass |
| 4 | add mentor | 1. Pada halaman home admin, | mentor bertambah | mentor bertambah | Pass |

| Id tes t | Deskripsi Testcase | Langkah uji | Hasil ekspektasi | Hasil asli | statu s |
|-------------------------|----------------------------------|--|--|--|--------------------|
| | | 2. tekan tombol interns lalu add mentor 3. input field dengan lengkap 4. tekan tombol add | | | |
| 5 | bikin modul dan soal | 1. ke halaman materi 2. lalu add 3. isi nama modul, desc , lalu ke halaman soal 4. input field yang sesuai 5. tekan tombol submit | modul dan soal berhasil tambah | modul dan soal berhasil tambah | Pass |
| 6 | bikin materi dan attendance | 1. ke halaman materi 2. lalu add 3. isi nama modul, desc , lalu ke halaman attendance input field 4. tekan siapa saja yang diabsen 5. submit | materi dan attendance berhasil di buat | materi dan attendance berhasil di buat | Pass |
| 7 | kursus gagal karena input kosong | 1. Pada halaman home admin, tekan tombol courses | error input kosong | error input kosong | pass |

| Id tes t | Deskripsi Testcase | Langkah uji | Hasil ekspektasi | Hasil asli | statu s |
|-------------------------|---|--|--|--|--------------------|
| | | 2. Tekan tombol plus 3. input kosong 4. tekan tombol add | | | |
| 8 | buat pengumuman gagal karena file dan kosong | 1. Pada halaman home admin, tekan tombol add announceme nt 2. input file nama dan desc tidak diinput 3. tekan tombol add | error attachment file kosong dan input kosong | error attachment file kosong dan input kosong | Pass |
| 9 | edit detail anak magang invalid password(dibawa h 6 karakter dan butuh kapital) | 1. Pada halaman home admin, 2. tekan tombol interns lalu edit interns 3. input password tidak valid 4. tekan tombol edit | error password minimal 6 huruf dan butuh kapital | error password minimal 6 huruf dan butuh kapital | Pass |
| 10 | add mentor gagal karena email / input kosong | 1. Pada halaman home admin, 2. tekan tombol interns lalu add mentor | error email / input kosong | error email / input kosong | Pass |

| Id tes t | Deskripsi Testcase | Langkah uji | Hasil ekspektasi | Hasil asli | statu s |
|-------------------------|--|---|--|--|--------------------|
| | | 3. input field email invalid / input koson 4. tekan tombol add | | | |
| 11 | bikin modul dan soal gagal input kosong | 1. ke halaman materi 2. lalu add 3. isi nama modul, desc , lalu ke halaman soal 4. input field yang sesuai 5. tekan tombol submit | modul dan soal error input kosong | modul dan soal error input kosong | Pass |
| 12 | bikin materi dan attendance gagal karena karena input kosong | 1. ke halaman materi 2. lalu add 3. isi nama modul, desc , lalu ke halaman attendance input field 4. tekan siapa saja yang diabsen 5. submit | materi dan attendance error input kosong | materi dan attendance error input kosong | Pass |

Tabel 5.3
Test case Mentor

4. anakmagang

Pada test case ini, akan berhubungan dengan anak magang. dimana pada halaman anak magang mereka bisa buat soal, liat announcement, liat detail announcement, chat,liat absensi ,kerjakan soal lalu upload jawaban.

| Id test | Deskripsi Testcase | Langkah uji | Hasil ekspektasi | Hasil asli | status |
|----------------|---------------------------|---|--|--|---------------|
| 1 | tampilkan announcement | 1. pada halaman home tekan tombol announcement | bisa melihat announcement dalam bentuk list | bisa melihat announcement dalam bentuk list | Pass |
| 2 | liat detail announcement | 1. pada halaman home tekan tombol announcement 2. tekan detail | bisa melihat announcement detail | bisa melihat announcement detail | Pass |
| 3 | baca group chat | 1. pada halaman home tekan tombol chat | bisa melihat chat orang lain | bisa melihat chat orang lain | Pass |
| 4 | mengerjakan soal | 1. pada halaman home tekan materi 2. tekan tombol tugas 3. kerjakan tugas | bisa menjawab soal dan melihat jawaban salah dan benar(pilgan) | bisa menjawab soal dan melihat jawaban salah dan benar(pilgan) | Pass |
| 5 | chatting | 1. pada halaman home tekan tombol chat 2. tulis pesan 3. tekan send | chat terkirim dengan baik | chat terkirim dengan baik | Pass |
| 6 | tampilkan absensi | 1. pada halaman home tekan tombol materi | bisa lihat absensi | bisa lihat absensi | Pass |
| 7 | chatting input kosong | 1. pada halaman home tekan tombol chat 2. tulis pesan 3. tekan send | pesan tidak ke send | pesan tidak ke send | pass |
| 8 | upload jawaban | 1. pada halaman home tekan materi 2. tekan tombol upload tugas | jawaban berhasil diupload | jawaban berhasil diupload | Pass |

| Id test | Deskripsi Testcase | Langkah uji | Hasil ekspektasi | Hasil asli | status |
|----------------|-----------------------------|---|---------------------------|-----------------------------|---------------|
| | | 3. upload file lalu submit | | | |
| 9 | upload jawaban tipe invalid | 1. pada halaman home tekan materi 2. tekan tombol upload tugas 3. upload file invalid lalu submit | muncul error file invalid | muncul error di console.log | failed |

Tabel 5.4
Test case Anak Magang

BAB VI

PENUTUPAN

Projek PT Orela Shipyard demikian selesai dengan tujuan utama untuk membuat sebuah website yang bisa dipakai untuk pembelajaran. Selama durasi proyek, tim telah bekerja secara kolaboratif untuk memenuhi semua kriteria , meskipun jadwal lebih telat daripada biasanya.

6.1 Kesimpulan

Projek pengembangan sistem pelatihan untuk mentor di PT Orela Shipyard berhasil diselesaikan dengan menyediakan platform pembelajaran yang mendukung materi pelatihan berupa modul, video, atau dokumen lain yang dapat diunggah oleh mentor atau admin perusahaan. Materi ini disampaikan melalui platform yang dapat diintegrasikan, seperti Google Classroom atau sistem internal perusahaan yang sesuai.

Sistem juga memungkinkan pembuatan soal ujian dengan berbagai tipe, seperti pilihan ganda, esai, dan praktik. Penilaian untuk soal pilihan ganda dapat dilakukan secara otomatis oleh sistem, memberikan efisiensi dalam evaluasi. Namun, penilaian untuk soal esai dan praktik memerlukan evaluasi manual oleh mentor, memastikan kualitas penilaian yang mendalam dan akurat.

Meskipun ada tantangan dalam hal keterlambatan jadwal, tim projek tetap berhasil menyelesaikan sistem dengan memenuhi semua kriteria yang ditetapkan. Sistem ini diharapkan menjadi fondasi yang kuat untuk proses pembelajaran dan evaluasi di PT Orela Shipyard.

6.2 Saran

Untuk pengembangan sistem pelatihan di PT Orela Shipyard ke depannya, disarankan agar fitur evaluasi ditingkatkan, misalnya dengan menambahkan teknologi pengenalan teks untuk mendukung penilaian otomatis pada soal esai sederhana. Hal ini dapat membantu mengurangi beban kerja mentor dalam proses evaluasi. Selain itu, integrasi sistem dengan berbagai platform pembelajaran seperti Learning Management System (LMS) lainnya juga dapat dipertimbangkan agar sistem lebih fleksibel dan sesuai dengan kebutuhan perusahaan. Fitur analitik pembelajaran juga sangat penting untuk ditambahkan, sehingga kinerja peserta dan mentor dapat dipantau secara komprehensif melalui analisis data hasil ujian dan tingkat partisipasi dalam pelatihan. Tidak hanya itu, otomatisasi pembuatan laporan hasil pelatihan dapat mempercepat dan mempermudah proses dokumentasi untuk admin perusahaan. Agar sistem dapat dimanfaatkan secara maksimal, pelatihan penggunaan sistem untuk mentor dan admin perusahaan juga perlu dilakukan. Dengan implementasi berbagai saran ini, sistem pelatihan diharapkan dapat terus berkembang dan memberikan manfaat yang lebih besar bagi PT Orela Shipyard.

LAMPIRAN

Lampiran: Pertanyaan dan Jawaban Terkait Sistem Pelatihan Anak Magang di PT. Orela Shipyard

1. Jenis Soal untuk Mentor:

- Jenis soal: Pilihan ganda atau esai (kedua jenis soal dapat digunakan)
- Jumlah pilihan untuk soal pilihan ganda: 4 opsi
- Format soal lain: Dapat menggunakan gambar

2. Materi yang Diupload:

- Pengunggah materi: Admin yang akan mengunggah materi
- Materi disediakan oleh: PT. Orela Shipyard (admin yang akan menyediakan materi)

3. Penilaian:

- Sistem penilaian: Menggunakan Google Form untuk penilaian otomatis pada soal pilihan ganda
- Tanggung jawab penilaian: Mentor yang bertanggung jawab dalam menilai hasil ujian
- Skema penilaian: Dirancang oleh mentor

4. Materi dan Modul Pengajaran:

- Modul pengajaran: Mentor dapat meminta atau menyediakan modul pengajaran khusus
- Distribusi modul: Modul diunggah melalui sistem
- Metode pengajaran: Belajar mandiri dengan mempelajari modul dan online meeting 2x dalam seminggu
- Media pengajaran: Menggunakan Microsoft Teams

5. Durasi Pelatihan dan Penilaian:

- Durasi pelatihan: 4 jam per minggu
- Penilaian: Dilakukan mingguan

6. Fitur Sistem:

- Fitur yang dibutuhkan: Manajemen kelas, pengelolaan jadwal, materi, laporan perkembangan, serta fitur penilaian otomatis untuk soal pilihan ganda

- Integrasi dengan platform lain: Sistem dapat diintegrasikan dengan aplikasi eksternal (misalnya Google Classroom) jika memungkinkan

7. Soal Ujian dan Penilaian:

- Penyusunan soal ujian: Dibuat oleh mentor
- Tanggung jawab penilaian: Mentor yang bertanggung jawab
- Penyusunan materi pembelajaran: Dilakukan oleh mentor

8. Tes Praktik:

- Terdapat tes praktik yang diharuskan bagi peserta magang

9. Proses Pelatihan Magang:

- Metode pelatihan: Online
- Pembagian mentor: Setiap divisi memiliki mentor tersendiri, dan terdapat pembagian yang lebih spesifik di setiap divisi

10. Penggunaan LMS:

- Penggunaan LMS: Belum menggunakan LMS, namun sistem dapat mengintegrasikan materi pelatihan yang tersedia untuk peserta magang

Gambar



