

Pengembangan Aplikasi *Automated Essay Scoring* Berbasis Web Untuk Penilaian Jawaban Teks pada Tugas dan Ujian Online dengan Metode *Personal eXtreme Programming*

(Studi Kasus : SMP Negeri 10 Kotabumi)

TUGAS AKHIR

Diajukan sebagai syarat menyelesaikan jenjang strata Satu (S-1)
di Program Studi Teknik Informatika, Jurusan Teknologi,
Produksi dan Industri, Institut Teknologi Sumatera

Oleh:

Markus Togi Fedrian Rivaldi Sinaga

118140037



**PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNOLOGI, PRODUKSI, DAN INDUSTRI
INSTITUT TEKNOLOGI SUMATERA
LAMPUNG SELATAN
2023**

DAFTAR ISI

DAFTAR ISI.....	2
DAFTAR TABEL.....	5
DAFTAR GAMBAR.....	5
DAFTAR RUMUS	6
DAFTAR LAMPIRAN.....	10
BAB I PENDAHULUAN.....	12
1.1 Latar Belakang Masalah	12
1.2 Rumusan Masalah.....	15
1.3 Tujuan Penelitian	15
1.4 Batasan Masalah	15
1.5 Manfaat Penelitian	16
1.6 Sistematika Penulisan	16
1.6.1 Bab I Pendahuluan.....	17
1.6.2 Bab II Tinjauan Pustaka	17
1.6.3 Bab III Metode Penelitian	17
1.6.4 Bab IV Hasil Penelitian dan Pembahasan	17
1.6.5 Bab V Kesimpulan dan Saran.....	17
BAB II TINJAUAN PUSTAKA	18
2.1 Tinjauan Pustaka.....	18
2.2 Dasar Teori	20
2.2.1. Aplikasi.....	20
2.2.2. <i>Automated Essay Scoring</i>	21
2.2.2. <i>Agile Software Development Life Cycle Model</i>	22
2.2.3. <i>Personal eXtreme Programming</i>	22

2.2.4.	<i>Unified Modelling Language (UML)</i>	25
2.2.5.	<i>Class Diagram</i>	25
2.2.6.	<i>Use Case Diagram</i>	25
2.2.7.	<i>Activity Diagram</i>	25
2.2.8.	<i>Class Diagram</i>	26
2.2.9.	<i>MoSCoW</i>	26
2.2.10.	<i>Unit Testing</i>	26
2.2.11.	<i>Black Box Testing</i>	26
2.2.12.	<i>System Usability Scale</i>	27
	BAB III METODE PENELITIAN	29
3.1.	Alur Penelitian	29
3.2.	Penjabaran Langkah Penelitian.....	29
3.2.1.	Studi Literatur.....	30
3.2.2.	Metode PXP (1) : Analisis Kebutuhan	30
3.2.3.	Metode PXP (2) : Perencanaan.....	30
3.2.4.	Metode PXP (3) : Inisialisasi Iterasi.....	32
3.2.5.	Metode PXP (4) : Perancangan	32
3.2.6.	Metode PXP (5) : Implementasi	32
3.2.7.	Metode PXP (6) : Pengujian Sistem	33
3.2.8.	Metode PXP (7) : Restropektif	33
3.2.9.	Evaluasi Sistem	33
3.2.10.	Penyusunan Laporan Akhir	33
3.3.	Alat dan Bahan Tugas Akhir	33
3.3.1.	Alat	34
3.3.2.	Bahan	34
3.4.	Metode Tugas Akhir	34

3.4.1.	Analisis Kebutuhan	35
3.4.2.	Perencanaan.....	37
3.4.3.	Inisialisasi Iterasi	43
3.4.4.	Perancangan.....	59
3.4.5.	Implementasi	81
3.4.6.	Pengujian Sistem	82
3.4.7.	Retrospektif	84
3.5.	Evaluasi Akhir Aplikasi.....	84
BAB IV HASIL PENELITIAN DAN PEMBAHASAN.....		85
4.1.	Lingkungan Pengembangan Aplikasi	85
4.2.	Personal eXtreme Programming	85
4.2.1	Analisis Kebutuhan	85
4.2.2	Perencanaan.....	85
4.2.3	Inisialisasi Iterasi	86
4.2.4	Perancangan.....	86
4.2.5	Iterasi Pertama	86
4.2.6	Iterasi Kedua.....	108
4.2.7	Iterasi Ketiga	108
4.2.8	Iterasi Keempat.....	108
4.3.	Rangkuman Iterasi	108
DAFTAR PUSTAKA		109
LAMPIRAN.....		111

DAFTAR TABEL

Tabel 2.1 Penelitian Terdahulu	18
Tabel 2.2 Kriteria Metode SUS	27
Tabel 3.1 Deskripsi Aktor.....	36
Tabel 3.2 <i>User Stories</i>	36
Tabel 3.3 Estimasi <i>User Stories</i>	37
Tabel 3.4 Prioritas <i>User Stories</i>	39
Tabel 3.5 Tabel Iterasi	41
Tabel 3.6 Atribut pada tabel <i>users</i>	46
Tabel 3.7 Atribut pada tabel <i>classrooms</i>	46
Tabel 3.8 Atribut pada tabel <i>classroom_member</i>	47
Tabel 3.9 Atribut pada Tabel <i>materials</i>	50
Tabel 3.10 Atribut pada Tabel <i>tasks</i>	51
Tabel 3.11 Atribut pada Tabel <i>questions</i>	52
Tabel 3.12 Atribut pada Tabel <i>answer_keys</i>	52
Tabel 3.13 Atribut pada Tabel <i>student_answer</i>	53
Tabel 3.14 Atribut pada Tabel <i>student_score</i>	54
Tabel 3.15 Pengujian Sistem Iterasi 1	82
Tabel 3.16 Pengujian Sistem Iterasi 2	82
Tabel 3.17 Pengujian Sistem Iterasi 3	83
Tabel 3.18 Pengujian Sistem Iterasi 4	83
Tabel 4.1 <i>Unit Testing</i> Iterasi Pertama	93

DAFTAR GAMBAR

Gambar 2.1 Fase - fase dalam Metode PXP [6]	23
Gambar 3.1 Alur Penelitian	29
Gambar 3.2 Fase - fase dalam Metode PXP	35
Gambar 3.3 <i>Use Case Diagram</i> Iterasi 1	44
Gambar 3.4 <i>Activity Diagram</i> Iterasi 1	45
Gambar 3.5 <i>Class Diagram</i> Iterasi 1	45
Gambar 3.6 <i>Use Case Diagram</i> Iterasi 2	48
Gambar 3.7 <i>Activity Diagram</i> Iterasi 2	49
Gambar 3.8 <i>Class Diagram</i> Iterasi 2	50
Gambar 3.9 <i>Use Case Diagram</i> Iterasi 3	55
Gambar 3.10 <i>Activity Diagram</i> Iterasi 3	56
Gambar 3.11 <i>Class Diagram</i> Iterasi 3	57
Gambar 3.12 <i>Use Case Diagram</i> Iterasi 4	58
Gambar 3.13 <i>Activity Diagram</i> Iterasi 4	58
Gambar 3.14 <i>Class Diagram</i> Iterasi 4	59
Gambar 3.15 Halaman Login Seluruh User.....	60
Gambar 3.16 Halaman Awal Guru Setelah Login	60
Gambar 3.17 Tampilan Modal "Buat Kelas" untuk Guru	61
Gambar 3.18 Tampilan "Lihat Kelas" untuk Guru.....	61
Gambar 3.19 Tampilan "Edit Kelas" untuk Guru.....	62
Gambar 3.20 Tampilan Konfirmasi "Penghapusan Kelas" untuk Guru	62
Gambar 3.21 Halaman Awal Siswa Setelah Login.....	63
Gambar 3.22 Tampilan "Gabung Kelas" untuk Siswa.....	63
Gambar 3.23 Tampilan "Lihat Kelas" untuk Siswa.....	64
Gambar 3.24 Tampilan Konfirmasi "Keluar Kelas" untuk Siswa	64
Gambar 3.25 Tampilan Awal Kepala Sekolah setelah Login.....	65
Gambar 3.26 Tampilan "Lihat Kelas" untuk Kepala Sekolah	65
Gambar 3.27 Halaman Profil untuk Seluruh Guru dan Siswa.....	66
Gambar 3.28 Tampilan "Daftar Materi" untuk Guru	66
Gambar 3.29 Tampilan "Buat Materi" Baru untuk Guru	67

Gambar 3.30 Halaman “Detail Materi” untuk Guru.....	67
Gambar 3.31 Tampilan “Ubah Materi” untuk Guru	68
Gambar 3.32 Tampilan Konfirmasi “Penghapusan Materi” untuk Guru	68
Gambar 3.33 Tampilan “Daftar Tugas” untuk Guru	69
Gambar 3.34 Tampilan “Buat Tugas” Baru untuk Guru	69
Gambar 3.35 Halaman “Lihat Tugas” untuk Guru	70
Gambar 3.36 Tampilan “Ubah Tugas” untuk Guru	70
Gambar 3.37 Tampilan Konfirmasi "Penghapusan Tugas" untuk Guru.....	71
Gambar 3.38 Tampilan "Tambah Soal Essay"	71
Gambar 3.39 Tampilan "Tambah Soal Pilihan Ganda"	72
Gambar 3.40 Tampilan “Impor Soal” pada Tugas untuk Guru	72
Gambar 3.41 Tampilan "Ubah Soal" pada Tugas untuk Guru.....	73
Gambar 3.42 Tampilan Konfirmasi “Penghapusan Soal”	73
Gambar 3.43 Tampilan Lihat dan Unduh Hasil Penggerjaan Tugas	74
Gambar 3.44 Halaman "Daftar Materi" untuk Siswa	74
Gambar 3.45 Halaman “Lihat Materi” untuk Siswa.....	75
Gambar 3.46 Halaman "Daftar Tugas" untuk Siswa	75
Gambar 3.47 Halaman Pendahuluan Sebelum Mengerjakan Tugas.....	76
Gambar 3.48 Halaman Penggerjaan Tugas.....	76
Gambar 3.49 Halaman Pendahuluan Setelah Mengerjakan Tugas.....	77
Gambar 3.50 Halaman Detail Hasil Penggerjaan Tugas	77
Gambar 3. 51 Halaman "Kelola Akun" untuk Admin	78
Gambar 3.52 Halaman Lihat Data Kelas untuk Admin.....	78
Gambar 3.53 Tampilan “Tambah Akun” untuk Admin	79
Gambar 3.54 Tampilan Ubah Akun untuk Admin	79
Gambar 3.55 Tampilan Untuk Melihat dan Mengunduh Materi	80
Gambar 3.56 Halaman Informasi Tugas dan Ujian untuk Siswa	80
Gambar 3.57 Halaman Informasi Tugas dan Ujian untuk Guru	81
Gambar 3.58 Halaman Informasi Tugas dan Ujian untuk Kepala Sekolah.....	81
Gambar 4.1 <i>Migration Tabel users</i>	87
Gambar 4.2 <i>Migration Tabel classrooms</i>	87

Gambar 4.3 <i>Migration</i> Tabel Pivot <i>classroom_member</i>	88
Gambar 4.4 <i>Model User</i>	89
Gambar 4.5 <i>Model Classroom</i>	90
Gambar 4.6 Visualiasi Basis Data Iterasi Pertama	91
Gambar 4.7 Data <i>Dummy</i> Tabel <i>users</i>	91
Gambar 4.8 Data <i>Dummy</i> Tabel <i>classrooms</i>	92
Gambar 4.9 Data <i>Dummy</i> Tabel Pivot <i>classroom_member</i>	92
Gambar 4.10 Kode <i>Unit Testing</i> Iterasi Pertama (1)	94
Gambar 4.11 Luaran <i>Unit Testing</i> Iterasi Pertama (1).....	94
Gambar 4.12 Kode <i>Unit Testing</i> Iterasi Pertama (2)	94
Gambar 4.13 Luaran <i>Unit Testing</i> Iterasi Pertama (2).....	95
Gambar 4.14 Kode <i>Unit Testing</i> Iterasi Pertama (3)	95
Gambar 4.15 Luaran <i>Unit Testing</i> Iterasi Pertama (3).....	95
Gambar 4.16 Kode <i>Unit Testing</i> Iterasi Pertama (4)	96
Gambar 4.17 Luaran <i>Unit Testing</i> Iterasi Pertama (4).....	96
Gambar 4.18 Kode <i>Unit Testing</i> Iterasi Pertama (5)	97
Gambar 4.19 Luaran <i>Unit Testing</i> Iterasi Pertama (5).....	97
Gambar 4.20 Kode <i>Unit Testing</i> Iterasi Pertama (6)	98
Gambar 4.21 Luaran <i>Unit Testing</i> Iterasi Pertama (6).....	98
Gambar 4.22 Kode <i>Unit Testing</i> Iterasi Pertama (7)	99
Gambar 4.23 Luaran <i>Unit Testing</i> Iterasi Pertama (7).....	99
Gambar 4.24 Kode <i>Unit Testing</i> Iterasi Pertama (8)	100
Gambar 4.25 Luaran <i>Unit Testing</i> Iterasi Pertama (8).....	101
Gambar 4. 26 Tampilan Login Untuk Semua Aktor	101
Gambar 4.27 Halaman Awal Setelah Login untuk Guru.....	101

DAFTAR RUMUS

DAFTAR LAMPIRAN

BAB I

PENDAHULUAN

1.1 Latar Belakang Masalah

Seiring pesatnya perkembangan teknologi informasi dan komunikasi, banyak bidang dalam kehidupan kita sehari-hari yang turut berkembang atau mengalami perubahan. Salah satunya adalah bidang pendidikan. Perubahan yang paling tampak pada bidang pendidikan tersebut adalah mulai ramainya praktik pembelajaran jarak jauh. Pembelajaran jarak jauh memanfaatkan teknologi informasi dan komunikasi seperti kelas virtual, konferensi video, dan forum diskusi *online* untuk memungkinkan terjadinya interaksi antara pengajar dan siswa meskipun mereka terpisah secara geografis. Seperti proses belajar mengajar pada umumnya, biasanya pada pembelajaran jarak jauh juga terdapat tugas dan ujian. Tugas dan ujian dilakukan sebagai salah satu cara untuk mengevaluasi pencapaian siswa dalam kegiatan belajar-mengajar [1]. Saat ini terdapat beragam platform ujian daring yang tersebar di internet seperti *Google Form*, *Quizziz*, *edBase*, *Testmoz*, dan sebagainya [2]. Namun seluruh platform yang disebutkan tadi, masih mengharuskan pengajar untuk melakukan penilaian jawaban satu persatu secara manual untuk soal tipe isian, yang mana hal ini tentu menyulitkan, dan tidak efisien dari segi waktu, di samping itu mengecek secara manual juga berpotensi mengakibatkan kesalahan dalam pengecekan.

SMP Negeri 10 Kotabumi yang berlokasi di Jl. Alamsyah Ratu Prawira Negara, Kelurahan Kelapa Tujuh, Kecamatan Kotabumi Selatan, Kabupaten Lampung Utara, Provinsi Lampung tersebut merupakan salah satu sekolah yang menghadapi situasi tersebut. Berdasarkan hasil diskusi dengan Wakil Kepala Bidang Kurikulum SMP Negeri 10 Kotabumi, yang sekaligus merangkap sebagai guru mata pelajaran Bahasa Inggris, Ibu Eny Ros, peneliti mendapati bahwa SMP Negeri 10 Kotabumi sudah tidak baru lagi terhadap teknologi berupa *platform* kelas virtual seperti *Google Classroom* maupun *platform* formulir *online* seperti *Google Form*. Ibu Eny mengaku bahwa adanya *platform* sejenis sangat memudahkan bagi guru khususnya yang mengajar pada mata pelajaran yang bersifat hafalan konsep seperti bahasa.

Ibu Eny juga menyampaikan ketika masa pandemi COVID lalu, SMP Negeri 10 cukup kewalahan dalam beradaptasi dengan proses belajar mengajar dalam jaringan,

sehingga media yang digunakan untuk menunjang kegiatan belajar mengajar saat itu hanyalah media sosial percakapan *WhatsApp*, yang dirasa paling mudah dijangkau dan mudah digunakan. Namun beliau juga menyampaikan bahwa salah satu kelas di SMP Negeri 10 Kotabumi, saat penelitian ini berlangsung, sedang dalam masa uji coba dimana untuk Penilaian Tengah Semester (PTS) dan Penilaian Akhir Semester (PAS) kelas tersebut akan dilaksanakan secara *online* dengan menggunakan layanan suatu penyedia jasa sistem informasi. Beliau juga menerangkan bahwa dalam beberapa waktu terakhir, terdapat beberapa penyedia jasa sistem informasi yang menawarkan layanan mereka, khususnya yang berbentuk *platform* ujian *online*. Namun dari apa yang dapat dirasakan pada praktiknya, menurut beliau beragam layanan tersebut masih belum cukup memudahkan, karena selain hanya mendukung penilaian otomatis pada soal bertipe pilihan ganda, yang mana untuk soal bertipe isian para guru masih harus melakukan pengecekan manual, namun juga tidak cukup memenuhi kebutuhan dari SMP Negeri 10, seperti fitur pengolahan dan visualisasi data hasil ujian siswa yang dirasa sangat dibutuhkan untuk membantu mengevaluasi proses belajar mengajar. Untuk menjawab situasi tersebut, perlu dikembangkan dan diteliti suatu aplikasi berbasis *web* yang dilengkapi dengan suatu model kecerdasan buatan berupa sistem penilaian otomatis untuk tugas dan ujian di SMP Negeri 10 Kotabumi.

Sistem penilaian otomatis (*Automatic Scoring System*) sendiri merupakan sistem yang mampu melakukan proses konversi dari performa dalam penyelesaian tugas (umumnya di bidang pendidikan dan penelitian) menjadi berbagai level atau karakteristik kualitas kemampuan [3]. Kemudian, karena yang menjadi fokus dari penelitian ini adalah penilaian jawaban teks, maka bidang terfokus yang paling sesuai adalah *Automated Essay Scoring* (AES), yang secara sederhana dapat dimaknai sebagai pemanfaatan kemampuan komputer untuk melakukan evaluasi dan penilaian pada kalimat yang diketik secara otomatis [4]. AES sendiri merupakan aplikasi dari Pengolahan Bahasa Alami (*Natural Language Processing*) yang merupakan salah satu fokus bidang dari Kecerdasan Buatan (*Artificial Intelligence*) [4]. Implementasi dari AES sendiri sudah cukup banyak diteliti dan dibahas. Pada penelitian ini sendiri, model AES yang digunakan dikembangkan oleh Pusat Riset dan Inovasi Kecerdasan Buatan Institut Teknologi Sumatera (PURINO Kecerdasan Buatan ITERA). Namun karena penelitian ini hanya berfokus pada pengembangan aplikasi berbasis web yang

akan menjadi wadah bagi model AES, peneliti akan lebih banyak membahas terkait metode SDLC (*Software Development Life Cycle*). Terdapat beberapa model SDLC, mulai dari model-model yang bersifat *plan-driven* yang paling awal dikenali seperti model *waterfall*, *prototyping*, dan *iterative*, hingga model yang lebih ringkas, ringan dan fleksibel seperti model *agile*.

Model *agile* sendiri memiliki beragam variasi metode, mulai dari *Scrum*, *eXtreme Programming* (XP), hingga *Personal eXtreme Programming* (PXP) [5]. Pada penelitian ini sendiri, untuk pengembangan aplikasi berbasis *web*-nya digunakan metode *agile* PXP, yang merupakan pengembangan dari metode *agile* terdahulunya yaitu *eXtreme Programming* (XP) yang berfokus pada empat hal, yaitu : keterlibatan klien, pengujian berkelanjutan, pemrograman dengan tim kecil yang terpadu, serta siklus iterasi yang singkat. Sementara untuk metode PXP, disamping empat hal yang menjadi fokus dalam metode XP, terdapat pula perhatian yang besar pada keotonomian pengembang individu [5]. PXP sebagai salah satu metode dalam model SDLC *agile* dipilih karena memberikan kenyamanan dan fleksibilitas tinggi pada pengembangan perangkat lunak berukuran kecil bila dibandingkan dengan model SDLC lainnya, serta sifat PXP yang menekankan pada keotonomian pengembang individu yang memungkinkannya untuk bekerja sesuai dengan cara dan kecepatannya sendiri. Selain itu, PXP membuat jalur komunikasi antara pengembang dan klien menjadi lebih singkat sehingga akan lebih mudah dan cepat untuk merumuskan berbagai kebutuhan dari perangkat lunak, serta lebih mudah melacak dan memprediksi perubahan yang terjadi [6]. Hal ini sesuai dengan kebutuhan pengembangan aplikasi penilaian teks otomatis berbasis web yang dapat dikatakan berukuran relatif kecil, dan berpotensi besar mengalami banyak penyesuaian selama proses pengembangan [7]–[10].

Di lain sisi, yang menjadi pertimbangan terbesar adalah batasan pada kemampuan peneliti yang sekaligus menjadi pengembang aplikasi dalam penelitian ini, dimana peneliti diharuskan untuk beradaptasi dan mempelajari berbagai teknologi yang paling tepat untuk digunakan dalam pengembangan aplikasi, dan lebih mampu bekerja dengan maksimal dengan keterlibatan klien yang memadai namun mudah dicapai. Hal ini sesuai dengan fleksibilitas, adaptabilitas, dan keotonomian yang tinggi bagi pengembang, yang bisa didapatkan dari metode *agile* model PXP.

Untuk mendukung metode SDLC yang dipilih, akan digunakan metode evaluasi *Unit Testing* pada tahap pengembangan, dan metode *Black Box Testing* serta *System Usability Scale (SUS)* pada tahap evaluasi akhir.

1.2 Rumusan Masalah

Berdasarkan latar belakang permasalahan yang telah diidentifikasi, berikut adalah rumusan masalah yang dapat saya susun :

1. Bagaimana proses pengembangan aplikasi *Automated Essay Scoring* berbasis *web* dengan menggunakan metode SDLC *agile* model PXP?
2. Bagaimana fungsionalitas dan kemudahan penggunaan aplikasi berbasis web yang dikembangkan dalam menggunakan kemampuan model AES yang telah dikembangkan berdasarkan hasil evaluasi menggunakan metode *Black Box Testing* dan metode *System Usability Scale*?

1.3 Tujuan Penelitian

Berikut adalah tujuan dari penelitian ini berdasarkan masalah-masalah yang telah dirumuskan :

1. Mengembangkan aplikasi *Automated Essay Scoring* berbasis *web* menggunakan model SDLC *agile* metode PXP.
2. Menganalisis fungsionalitas dan kemudahan penggunaan aplikasi berbasis web saat menggunakan kemampuan model *Automated Essay Scoring* dalam melakukan penilaian jawaban teks singkat secara otomatis melalui hasil evaluasi menggunakan metode *Black Box Testing* dan *System Usability Scale*.

1.4 Batasan Masalah

Batasan masalah dalam penelitian ini perlu ditetapkan untuk kespesifikasi tujuan dari penelitian, dan tidak membebani berbagai pihak yang terkait dengan penelitian ini, adapun rumusan masalahnya sebagai berikut :

1. Metode SDLC yang digunakan dalam pengembangan aplikasi adalah model *agile* metode PXP.
2. Penelitian hanya bertujuan untuk mengembangkan aplikasi berbasis web yang akan menjadi wadah untuk model AES yang telah dikembangkan sebelumnya,

sehingga dokumen penelitian ini tidak akan membahas secara mendalam terkait model kecerdasan buatan yang digunakan.

3. Aplikasi yang dikembangkan hanya akan dapat digunakan oleh admin, tim manajemen pengembangan, serta pengguna yang telah didaftarkan oleh admin.
4. Tim manajemen pengembangan, pengguna, dan admin yang dimaksud pada poin sebelumnya bisa berasal dari lingkup Program Studi Teknik Informatika ITERA (khususnya pembimbing dan penguji penelitian ini), lingkup PURINO Kecerdasan Buatan ITERA sebagai pengembang model AES pada penelitian ini, lingkup SMP Negeri 10 Kotabumi sebagai studi kasus penelitian ini, dan peneliti sendiri.
5. Aplikasi hanya menyesuaikan dengan karakteristik masukan dan luaran yang dibutuhkan oleh model kecerdasan buatan yang telah dikembangkan sebelumnya.

1.5 Manfaat Penelitian

Manfaat yang diharapkan dari penelitian ini adalah sebagai berikut :

1. Memperkenalkan secara singkat metode-metode yang digunakan dalam penelitian ini kepada pembaca.
2. Membantu memudahkan pengembangan dan penelitian lebih lanjut di kemudian hari.
3. Bagi SMP Negeri 10 Kotabumi, luaran penelitian ini berupa aplikasi penilaian otomatis berbasis web sebagai media yang dapat menunjang kemudahan kegiatan belajar mengajar.
4. Bagi peneliti, sebagai sarana untuk memperdalam pengetahuan dan kemampuan di bidang teknologi informasi, khususnya pengolahan bahasa alami, dan pengembangan web.
5. Memenuhi tanggung jawab menyelesaikan tugas akhir sebagai prasyarat kelulusan peneliti.

1.6 Sistematika Penulisan

Sistematika penulisan dokumen penelitian ini terdiri dari lima bab utama, yaitu sebagai berikut :

1.6.1 Bab I Pendahuluan

Berisi gambaran umum terkait isi dari dokumen penelitian ini, antara lain, latar belakang, rumusan masalah, tujuan penelitian, batasan masalah penelitian, manfaat penelitian, dan sistematika penulisan dan penyusunan dokumen penelitian.

1.6.2 Bab II Tinjauan Pustaka

Berisi informasi singkat terkait berbagai karya ilmiah yang dijadikan sumber acuan dalam penyusunan dokumen penelitian ini.

1.6.3 Bab III Metode Penelitian

Berisi deskripsi rinci terkait berbagai metode yang digunakan dalam melakukan penelitian ini.

1.6.4 Bab IV Hasil Penelitian dan Pembahasan

Berisi deskripsi rinci serta pembahasan menyeluruh terkait hasil dari penelitian yang telah dilakukan.

1.6.5 Bab V Kesimpulan dan Saran

Berisi kesimpulan yang merangkum hasil analisis dari pembahasan hasil penelitian yang telah dilakukan

BAB II

TINJAUAN PUSTAKA

2.1 Tinjauan Pustaka

Penelitian ini tentunya tidak terlepas dari bantuan dan ilham dari berbagai penelitian serupa maupun terkait terdahulu. Selain itu, berbagai penelitian terdahulu juga digunakan sebagai pembanding, dan acuan untuk meningkatkan hasil yang diharapkan. Berikut ini adalah berbagai penelitian terdahulu yang menjadi acuan bagi penelitian ini :

Tabel 2.1 Penelitian Terdahulu

No	Judul	Peneliti	Tahun	Metode	Hasil	Perbedaan
1	Implementasi Metode <i>Personal Extreme Programming</i> dalam Pengembangan Sistem Manajemen Transaksi Perusahaan (Studi Kasus: CV. Todjoe Sinar Group)	Muhammad Ulfi, Gita Indah Marthasari, Ilyas Nuryasin	2020	Metode SDLC : Personal eXtreme Programming	Sistem Informasi yang dapat memudahkan manajemen transaksi perusahaan	Implementasi metode SDLC akan dilakukan pada aplikasi yang memiliki kegunaan dan bidang studi kasus berbeda.
2	<i>Personal Extreme Programming with MoSCoW Prioritization for Developing Library Information System</i>	Gita Indah Marthasari, Wildan Suharso, Frendy Ardiansyah	2018	Metode SDLC : Personal eXtreme Programming	Sistem Informasi yang dapat memudahkan manajemen perpustakaan	Implementasi metode SDLC akan dilakukan pada aplikasi yang memiliki kegunaan dan bidang studi kasus berbeda
3	Sistem Informasi Program Keluarga Harapan (PKH) Menggunakan Metode <i>Personal Extreme Programming</i> dengan Metode	Abdullah Faqih Septiyanto, Wildan Suharso, Ilyas Nuryasin	2020	Metode SDLC : Personal eXtreme Programming	Sistem informasi yang dapat menunjukkan data sederhana dan mengunggah file	Implementasi metode SDLC akan dilakukan pada aplikasi yang memiliki kegunaan dan bidang studi kasus berbeda

No	Judul	Peneliti	Tahun	Metode	Hasil	Perbedaan
	Prioritas Ranking					
4	Muster: Virtual Classroom For Students Using D-Jango	Mahesh Kancharla, Govinda Sai Kamisetty, Suraj Hussain Dudekula	2022	Model SDLC : Iterative	Sistem Ruang Kelas Virtual untuk media pembelajaran daring	Pembahasan yang menjadi fokus penelitian

Penelitian pertama dilakukan oleh Muhammad Ulfie, Gita Indah Marthasari, dan Ilyas Nuryasin pada tahun 2020 dengan judul “Implementasi Metode *Personal Extreme Programming* dalam Pengembangan Sistem Manajemen Transaksi Perusahaan (Studi Kasus: CV. Toedjoe Sinar Group)” [11], pada penelitian ini menggunakan metode PXP. Berbeda dengan penelitian yang dibahas pada laporan ini, luaran dari penelitian Ulfie dan rekan tidak menyinggung terkait *Automated Essay Scoring*, melainkan berfokus dalam pengembangan sistem informasi yang dapat memudahkan manajemen transaksi perusahaan. Namun salah satu masalah yang terjadi adalah ketidaksesuaian waktu pengerjaan dengan waktu yang sebelumnya telah diestimasikan, akibat dari ketidak-familiar-an pengembang dengan salah satu permintaan (*requirement*) dari pengguna.

Pada penelitian kedua dilakukan oleh Gita Indah Marthasari, Wildan Suharso, dan Frendy Ardiansyah pada tahun 2018 dengan judul “*Personal Extreme Programming with MoSCoW Prioritization for Developing Library Information System*” [12], pada penelitian ini digunakan metode PXP. Berbeda dengan penelitian yang dibahas pada laporan ini, luaran dari penelitian Marthasari dan rekan tidak menyinggung terkait *Automated Essay Scoring*, melainkan berfokus dalam pengembangan sistem informasi yang dapat memudahkan manajemen perpustakaan. Kekurangan dari penelitian ini adalah meski telah menggunakan pendekatan MoSCoW untuk penentuan prioritas kebutuhan, namun tim pengembang tetap tidak dapat mencegah terjadinya waktu tunggu kala transisi antar iterasi.

Pada penelitian ketiga dilakukan oleh Abdullah Faqih Septiyanto, Wildan Suharso, dan Ilyas Nuryasin pada tahun 2020 dengan judul “Sistem Informasi Program Keluarga Harapan (PKH) Menggunakan Metode *Personal Extreme Programming* dengan Metode Prioritas Ranking” [13]. Pada penelitian ini digunakan metode SDLC

model Personal Extreme Programming dengan Metode Prioritas Ranking. Berbeda dengan penelitian yang dibahas pada laporan ini, luaran dari penelitian Septiyanto dan rekan tidak menyinggung terkait *Automated Essay Scoring*, melainkan berfokus dalam pengembangan sistem informasi yang dapat menunjukkan data dengan proses sederhana dan mengunggah file. Selain itu Septiyanto dan rekan tidak menggunakan metode *MoSCoW*, melainkan menggunakan Metode Prioritas Ranking dalam menentukan prioritas kebutuhan sistem.

Pada penelitian keempat dilakukan oleh Mahesh Kancharla, Govinda Sai Kamisetty, dan Suraj Hussain Dudekula pada tahun 2022 dengan judul penelitian “Muster: *Virtual Classroom For Students Using D-Jango*” [14], pada penelitian ini dilakukan pengembangan kelas virtual. Berbeda dengan penelitian yang dibahas pada laporan ini, luaran dari penelitian Marthasari dan rekan tidak menyinggung terkait *Automated Essay Scoring*, melainkan berfokus dalam pengembangan Sistem Ruang Kelas Virtual untuk media pembelajaran daring. Namun, kekurangan dari penelitian ini adalah kurangnya pembahasan terkait metode SDLC yang digunakan.

Berdasarkan penelitian sebelumnya, penulis mengajukan penelitian mengenai Pengembangan Aplikasi dengan kemampuan AES untuk penilaian jawaban teks pada tugas dan ujian *online* di SMP Negeri 10 Kotabumi. Perbedaan penelitian yang dilakukan peneliti dengan penelitian terkait adalah pada bidang studi kasus yang digunakan peneliti. Metode yang digunakan peneliti adalah *Personal Extreme Programming* (PXP) dan fitur yang menjadi pembeda dalam penelitian ini adalah fitur penilaian otomatis menggunakan bantuan model kecerdasan buatan.

2.2 Dasar Teori

2.2.1. Aplikasi

Dalam ilmu komputer, aplikasi (aplikasi perangkat lunak) merujuk pada program komputer yang dibangun untuk melakukan tugas spesifik tertentu. Aplikasi dapat berupa perangkat lunak yang dapat diinstal di perangkat seperti komputer, telepon seluler, atau tablet. Aplikasi perangkat lunak yang baik dibangun untuk menyelesaikan tugas-tugas tertentu dengan lingkup yang spesifik serta berfokus pada peningkatan efisiensi dan produktivitas.

2.2.2. Automated Essay Scoring

Sistem penilaian otomatis (*Automatic Scoring System*, biasa disingkat *Automatic Scoring*) merupakan sistem yang mampu melakukan proses konversi dari performa dalam penyelesaian tugas (umumnya di bidang pendidikan dan penelitian) menjadi berbagai level atau karakteristik kualitas kemampuan [3]. Dalam penelitian ini, yang menjadi fokus dari bidang penilaian otomatis adalah penilaian teks jawaban singkat, maka bidang terfokus yang paling sesuai adalah *Automated Essay Scoring* (AES), yang secara sederhana dapat dimaknai sebagai pemanfaatan kemampuan komputer untuk secara otomatis melakukan evaluasi dan penilaian pada kalimat yang diketik [4]. AES sendiri merupakan aplikasi dari Pengolahan Bahasa Alami (*Natural Language Processing*) yang merupakan salah satu fokus bidang dari Kecerdasan Buatan (*Artificial Intelligence*), dimana AES melakukan penilaian secara otomatis dengan cara melakukan pembobotan pada istilah atau kata, dan melakukan perbandingan untuk mendapatkan nilai kemiripan antara dua atau lebih dokumen yang berkaitan [4].

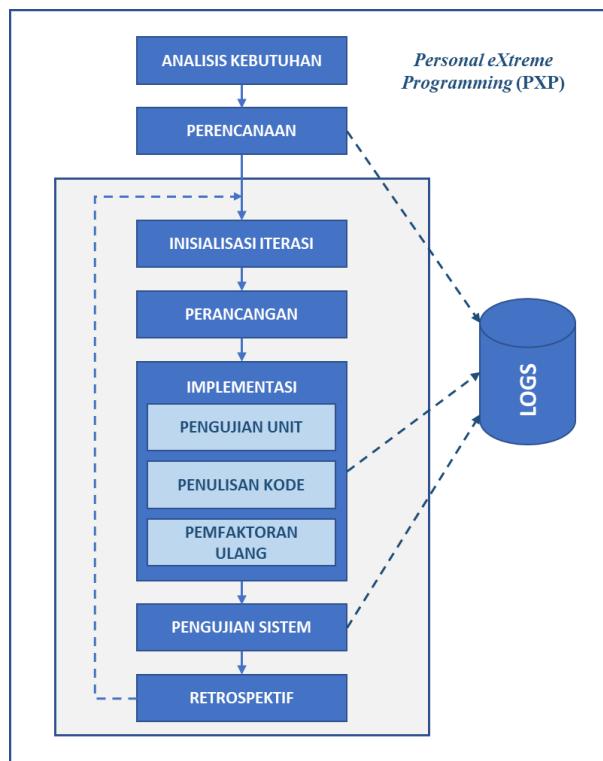
Pada penelitian ini sendiri, model AES yang digunakan telah dikembangkan sebelumnya oleh pihak PURINO Kecerdasan Buatan ITERA, menggunakan kombinasi metode pembobotan TF-IDF dengan metode pengecekan kemiripan *Cosine Similarity*. TF-IDF merupakan singkatan dari *Term Frequency – Inverse Document Frequency*. *Term Frequency* merujuk pada frekuensi kemunculan suatu istilah (*term*, umumnya dilambangkan sebagai t) dalam suatu dokumen (*document*, umumnya dilambangkan sebagai d) teks. Sehingga frekuensi (*frequency*, dilambangkan sebagai f) suatu istilah dalam teks, biasa dilambangkan sebagai tf . Sementara untuk *Inverse Document Frequency*, memiliki tujuan yang sama seperti tf yaitu pembobotan kata/istilah. Karena dalam praktiknya di dunia nyata pembobotan kata tidak dihadapkan pada satu dokumen saja, maka idf digunakan sebagai pendekatan yang cukup adil untuk menurunkan bobot dari istilah yang kemunculannya berlebih dalam suatu kumpulan dokumen berjumlah masif. [11]

2.2.2. Agile Software Development Life Cycle Model

Agile merupakan model daur hidup pengembangan perangkat lunak yang bersifat *incremental*. Model *agile* memberikan kemudahan dalam pengembangan perangkat lunak berskala kecil atau menengah ke bawah. Setiap tahapan *incremental*-nya sendiri berfokus untuk mengembangkan perangkat lunak secara cepat, bertahap, dan melibatkan pengguna secara langsung untuk menghasilkan luaran berkualitas tinggi. Metode *agile* memiliki beberapa variasi metode diantaranya adalah *Scrum*, *eXtreme Programming*, *Personal eXtreme Programming*, *Adaptive Software Development*, *Dynamic Systems Development Method*, dan *Agile Modeling* [15].

2.2.3. Personal eXtreme Programming

Personal eXtreme Programming (PXP) merupakan salah satu metode dari model SDLC *agile*, yang merupakan pengembangan dari metode terdahulunya yaitu *eXtreme Programming*, yang menitikberatkan pada sinergi sesama anggota tim pengembang berlingkup kecil. Untuk meningkatkan performa khususnya pada pengembang individu, PXP lebih menguntungkan karena pengembang dapat menentukan cara dan waktu bekerjanya sendiri, sehingga pengembang lebih mudah dalam melacak serta memprediksi perubahan yang akan terjadi [7]. Metode ini dipilih karena laju prosesnya yang relatif cepat, cocok untuk pengembangan perangkat lunak skala menengah ke bawah, dan fleksibilitasnya yang tinggi, serta pengembang tidak diharuskan untuk melakukan dokumentasi perencanaan matang secara menyeluruh terkait kebutuhan perangkat lunak yang bisa saja tidak dapat langsung diidentifikasi saat mengawali pengembangan.



Gambar 2.1 Fase - fase dalam Metode PXP [6]

PXP menuntut pengembang untuk bertanggung jawab pada setiap tugas dan perubahan yang terjadi. PXP terdiri dari beberapa fase dalam proses pengembangan perangkat lunak. Tahapan-tahapan tersebut dapat dilihat pada Gambar 2.1., dengan rincian dari tiap fasenya adalah sebagai berikut [6]:

1. Analisis Kebutuhan

Analisis Kebutuhan merupakan fase dimana pengembang mengumpulkan kebutuhan perangkat lunak melalui diskusi atau wawancara dengan *client*. Kebutuhan-kebutuhan yang diperoleh akan dituliskan dalam bentuk *user stories*.

2. Perencanaan

Pada fase perencanaan, pengembang menyusun dan membuat sekumpulan *task* yang akan dilaksanakan pada setiap iterasi berdasarkan kebutuhan yang disusun dari *user stories* yang telah didapatkan. Pembagian *task* dilakukan berdasarkan prioritas dari *user stories* dan estimasi waktu pengerjaan. Penyusunan tugas-tugas yang dilakukan pengembang disebut dengan *practice planning game* [16][37Fikri].

3. Inisialisasi Iterasi

Fase ini adalah fase awal yang dilaksanakan untuk memulai *task* yang akan dikerjakan. Pada fase ini dilakukan pemilihan tugas yang akan dijadikan fokus utama dari iterasi yang sedang berlangsung, dan pemodelan sistem dalam bentuk *Unified Modeling Language* (UML).

4. Perancangan

Fase perancangan sendiri merupakan fase dimana pengembang memodelkan modul dan kelas yang nantinya akan diimplementasikan di dalam sistem selama proses iterasi yang berlangsung. Rancangan yang dibuat pengembang berupa desain antarmuka pengguna, hanya mengacu kepada kebutuhan *client* yang diperoleh pada tahap *requirement*, tanpa memikirkan akan perubahan di masa mendatang. Pengembang diberi kebebasan untuk memilih metode perancangan apa yang paling tepat, namun pada PXP sangat disarankan untuk membuat proses ini sesederhana mungkin.

5. Implementasi

Fase ini merupakan fase pengimplementasian setiap objek pada tahap perancangan menjadi fitur / unit program. Fase ini memiliki tiga sub-fase yaitu: **Pengujian Unit, Penulisan Kode, dan Pemfaktoran Ulang Kode**. Untuk dapat melanjutkan ke fase berikutnya dari fase implementasi ini, kode program yang dibuat harus dapat dijalankan tanpa error dan lolos sub-fase pengujian unit.

6. Pengujian Sistem

Pada fase ini dilakukan pengujian fungsionalitas semua fitur yang ada dalam sistem. Pada penelitian ini hasil pengujian disajikan melalui *Black Box Testing*. Pada fase ini, seluruh kesalahan pada sistem dicatat dan diperbaiki.

7. Retrospektif

Fase ini merupakan fase terakhir untuk setiap iterasi. Pada fase ini pengembang melakukan analisis terhadap estimasi waktu pengembangan yang dibuat ketika fase perencanaan dengan waktu pengembangan sesungguhnya pada iterasi yang sedang berlangsung, dan menganalisis potensi hambatan untuk menekan kemungkinan terjadinya kesalahan dalam memperkirakan waktu penggerjaan iterasi pada proyek berikutnya. Pada fase retrospektif ini pula, penyebab

kesalahan atau gangguan dalam suatu iterasi dicatat dan dianalisis untuk mencegah hal serupa terulang di iterasi selanjutnya.

Di sepanjang seluruh proses, pengembang mengelola dan memelihara sejumlah catatan (*log*) yang berisi informasi terkait perencanaan tiap *task* dan duarasi sesungguhnya ketika pelaksanaan *task* tersebut, jumlah dan detail kesalahan yang terjadi, serta saran peningkatan untuk iterasi maupun proyek berikutnya [6].

2.2.4. *Unified Modelling Language (UML)*

UML merupakan bahasa standar yang digunakan untuk menulis / menotasikan cetak biru perangkat lunak. UML dapat digunakan untuk tujuan visualisasi, penentuan, pembangunan, dan dokumentasi hal-hal yang berhubungan erat dengan sistem perangkat lunak. UML umumnya dinotasikan dalam bentuk diagram seperti : *Class Diagram*, *Use Case Diagram*, *Activity Diagram*, dan sebagainya [17].

2.2.5. *Class Diagram*

Class Diagram merupakan salah diagram yang paling banyak digunakan dalam pemodelan sistem berbasis orientasi-kepada-objek. Diagram ini memvisualisasikan sekumpulan kelas dan antarmukanya, serta hubungan antara kelas-kelas tersebut [17].

2.2.6. *Use Case Diagram*

Use Case Diagram merupakan notasi dalam UML yang fokusnya berpusat pada pemodelan perilaku suatu sistem, subsistem, atau kelas, yang masing-masing menunjukkan sekumpulan kasus penggunaan, aktor, dan hubungan satu sama lain antara mereka [17].

2.2.7. *Activity Diagram*

Activity Diagram, merupakan notasi lainnya dari UML, yang digunakan untuk memodelkan aspek dinamis dari suatu sistem, seperti langkah berurut dalam proses komputasi, maupun alur yang dilalui suatu objek dalam sistem, dari suatu kondisi ke kondisi lainnya dalam aliran kontrol sistem tersebut [17].

2.2.8. Class Diagram

Class diagram adalah diagram yang menggambarkan struktur dan deskripsi class, package dan obyek beserta hubungan satu sama lain seperti *containment*, pewarisan, asosiasi, dan lainlain [42].

2.2.9. MoSCoW

MoSCoW merupakan akronim dari *Must have*, *Should Have*, *Could Have*, dan *Won't Have*. MoSCoW merupakan aturan yang digunakan dalam menentukan prioritas pada kebutuhan dalam pengembangan perangkat lunak. *Must have* adalah bagian paling dasar dari sistem yang dibangun. *Should have* adalah bagian penting dari sistem yang dibangun namun ada jangka waktu tertentu yang digunakan untuk pengerjaannya. *Could have* adalah bagian sistem yang dapat dikeluarkan dari rencana apabila waktu yang dimiliki tidak ada lagi. *Won't have* adalah prioritas yang tidak terlalu dibutuhkan untuk sistem yang akan dibangun.

2.2.10. Unit Testing

Dengan *Unit Testing*, pengembang melakukan pengujian terhadap metode atau class spesifik secara terpisah satu per satu. Dengan melakukan *unit testing*, pengembang diharuskan untuk memahami bagaimana metode dalam program ditulis, bagaimana bentuk data yang diolah dalam metode tersebut, apa keunikan dan tugas khusus dari metode tersebut, dan seperti apa tipe dan nilai luaran yang diharapkan dari metode tersebut [5].

Unit test biasanya ditulis dalam bahasa pemrograman yang sama dengan perangkat lunak dan menggunakan kerangka atau pustaka pengujian yang menyediakan alat yang diperlukan untuk membuat dan menjalankan pengujian secara otomatis dan terus menerus sebagai bagian dari proses pembangunan perangkat lunak.

2.2.11. Black Box Testing

Black box testing adalah jenis pengujian perangkat lunak di mana fungsionalitas perangkat lunak tidak diketahui. Pengujian dilakukan tanpa pengetahuan internal terkait produk yang diuji, artinya pengujian dilakukan dari sudut pandang pengguna luar. Tujuan dari pengujian ini adalah untuk memverifikasi apakah perangkat lunak dapat berfungsi dengan benar dan memenuhi persyaratan fungsional yang telah ditentukan tanpa perlu mengetahui bagaimana kode atau sistem di dalamnya bekerja.

Metode pengujian ini bertujuan untuk menemukan cacat dalam fungsionalitas, tampilan, dan kinerja perangkat lunak. Tes ini dilakukan dengan menguji masukan dan luaran dari perangkat lunak dan memeriksa apakah masukan yang dihasilkan sesuai dengan yang diharapkan. Dalam praktiknya, *black box testing* sering dilakukan oleh tim pengujian yang terpisah dari tim pengembangan perangkat lunak untuk memastikan bahwa pengujian dilakukan secara independen dan obyektif [5].

2.2.12. System Usability Scale

Metode *System Usability Scale* (SUS) merupakan metode yang menggunakan kuisioner untuk mengukur persepsi kegunaan perangkat lunak. Pengujian metode ini dilakukan setelah sistem telah selesai dibangun dan dikembangkan [18]. Metode SUS berisi 10 pertanyaan yang diberikan skala 1 sampai dengan skala 5. Pengertian skala yang dimaksud adalah 1 artinya sangat tidak setuju dan juga 5 yang mengartikan sangat setuju [19]. Metode SUS memiliki kriteria yang digunakan untuk mengelompokkan hasil kuisioner yang diperoleh dari responden. Adapun kriteria metode SUS adalah sebagai berikut [18]:

Tabel 2.2 Kriteria Metode SUS

SUS	Tingkatan	Kriteria
$x > 80,3$	A	Sangat Baik
$68 < x \leq 80,3$	B	Baik
$x = 68$	C	Cukup
$51 \leq x < 68$	D	Kurang
$X < 51$	E	Sangat Kurang

Pada tabel 2.1 telah dijelaskan bahwa hasil SUS diatas $> 80,3$ maka kriterianya sangat baik. Nilai terendah lebih kecil < 51 maka kriterianya sangat kurang.

Pertanyaan yang diberikan kepada responden akan dihitung skornya dengan menggunakan rumus metode SUS. Adapun rumus yang digunakan untuk menghitung skor dalam metode SUS dapat dilihat pada rumus 2.1

$$\tilde{x} = \frac{\Sigma x}{n} \quad (2.2)$$

Keterangan rumus 2.1:

\tilde{x} = Skor rata-rata

Σx = Jumlah Skor SUS

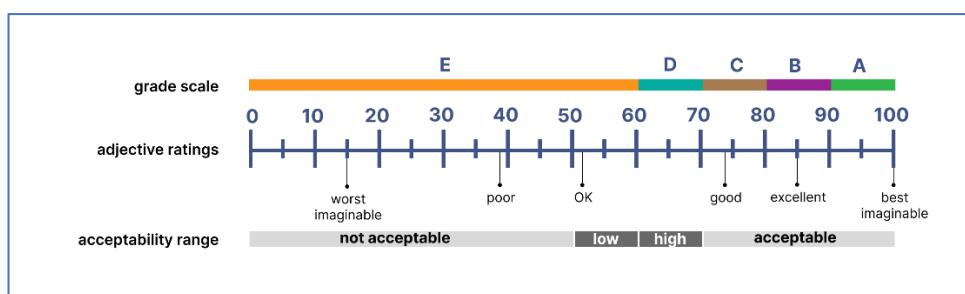
n = Jumlah responden

Perhitungan Skor dalam metode SUS menggunakan beberapa tahapan, diantaranya adalah sebagai berikut [20]:

Dari setiap soal yang bernomor ganjil, maka skor dikurangi 1 (X-1).

1. Pernyataan bernomor genap, nilainya akan dikurangi dengan 5 (5-X).
2. Hasil berkisar dari 0 hingga 4 atau bisa disimpulkan bahwa 4 adalah nilai paling positif.
3. Jumlahkan skor nilai untuk jawaban bernomor genap dan bernomor ganjil lalu kalikan jumlah proporsinya dengan 2.5.
4. Menghitung rata-rata jawaban instrument.

Penentuan hasil perhitungan SUS merujuk pada kepada tiga aspek utama, yaitu akseptabilitas (*acceptability range*), skala nilai (*adjective ratings*), dan tingkat kriteria sistem (*grade scale*). Akseptabilitas merupakan aspek yang menentukan penerimaan suatu sistem dalam kondisi seperti tidak dapat diterima (*not acceptable*), marginal (*low or high*), dan dapat diterima (*acceptable*). Skala nilai digunakan untuk menentukan tingkat kualitas aplikasi yang terdiri dari tingkat A,B,C,D, dan E. Sementara tingkat kriteria sistem adalah yang menentukan kebergunaan sistem yang meliputi 6 tingkatan yaitu terburuk (*worst imaginable*), buruk (*poor*), oke, baik (*good*), sangat baik (*excellent*), dan istimewa (*best imaginable*) [21]. Pedoman umum interpretasi SUS score dapat dilihat pada gambar 2.2



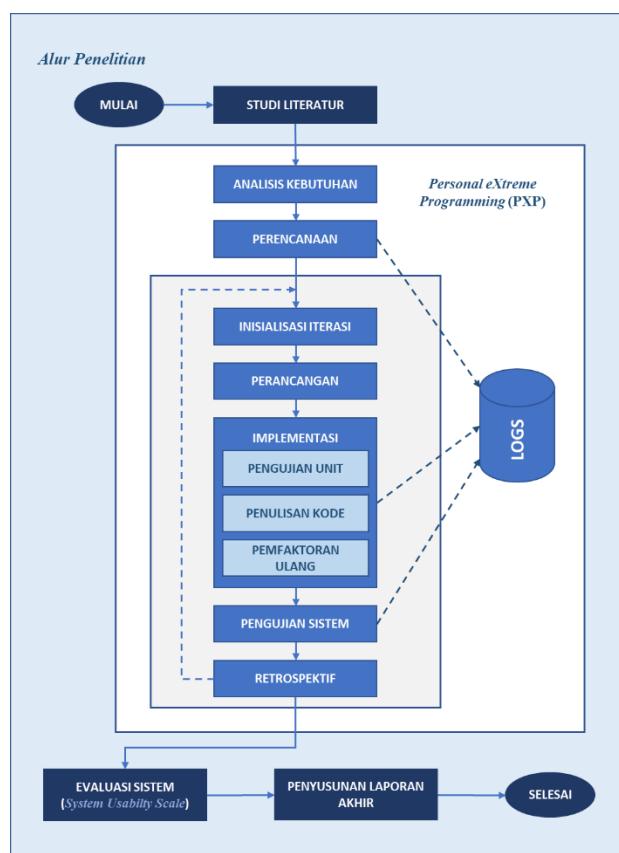
Gambar 2.2 Pedoman umum interpretasi SUS Score [22]

BAB III

METODE PENELITIAN

3.1. Alur Penelitian

Alur penelitian merupakan tahapan pelaksanaan yang disusun untuk membantu mempermudah jalannya penelitian. Alur penelitian ini dituliskan dalam bentuk flowchart atau diagram alir yang menggambarkan semua tahapan dari awal hingga akhir. Diagram alir dapat dilihat dibawah ini :



Gambar 3.1 Alur Penelitian

Alur penelitian ini seperti yang ditunjukkan pada diagram alir di atas dimulai dari studi literatur lalu masuk ke tahapan PXP yaitu analisis kebutuhan, perencanaan, inisialisasi iterasi, perancangan, implementasi, pengujian sistem, retrospektif, evaluasi sistem, penyusunan laporan akhir dan selesai.

3.2. Penjabaran Langkah Penelitian

Penelitian ini dilakukan dengan langkah-langkah yang sudah digambarkan pada gambar 3.1. Berikut merupakan uraian dari setiap langkah dalam alur penelitian :

3.2.1. Studi Literatur

Pengembangan aplikasi AES berbasis web ini membutuhkan pemahaman teoritis mengenai bidang-bidang yang terkait dalam penelitian ini. Studi literatur dilakukan dengan mempelajari berbagai referensi baik dari jurnal, buku, dan situs-situs terpercaya. Pemahaman pengembang terhadap teoritis yang lebih baik diharapkan dapat membuat pelaksanaan penelitian yang dilakukan menjadi tepat guna, sehingga berbagai kesulitan dalam proses pengembangan dapat diatasi.

Jurnal yang dijadikan referensi dalam penelitian ini, beberapa diantaranya adalah; “Implementasi Metode Personal Extreme Programming dalam Pengembangan Sistem Manajemen Transaksi Perusahaan (Studi Kasus: CV. Todjoe Sinar Group)” oleh Muhammad Ulfi dan rekan, dan “Personal Extreme Programming with MoSCoW Prioritization for Developing Library Information System” oleh Gita Indah Marthasari dan rekan. Sementara beberapa buku yang dijadikan acuan utama dalam penelitian ini adalah buku yang berjudul “*Software Development, Design and Coding*”, yang ditulis oleh J. F. Dooley. Selain itu penelitian ini juga menggunakan salah satu dokumen tesis berjudul “The anatomy of the modern window manager”, oleh M. v. Deurzen.

3.2.2. Metode PXP (1) : Analisis Kebutuhan

Analisis kebutuhan merupakan tahap pertama dalam metode PXP. Tahapan ini dijadikan pengembang untuk mengumpulkan kebutuhan-kebutuhan yang akan dituangkan kedalam sistem. Pengumpulan kebutuhan ini dilakukan dengan wawancara dan diskusi bersama pihak SMPN 10 Kotabumi. Kebutuhan-kebutuhan yang diperoleh dari hasil wawancara dituliskan dalam bentuk *user stories*.

3.2.3. Metode PXP (2) : Perencanaan

Tahap perancanaan merupakan tahap dimana pengembang menentukan prioritas dari setiap *user story*, dan menyusunnya menjadi tugas-tugas (*tasks*) yang akan dikerjakan dalam setiap iterasi [37Fikri]. Tahap ini terbagi menjadi tiga fase, yaitu :

1. Mengestimasikan waktu pengerjaan masing-masing *user story*.

Pada fase ini, pengembang mengestimasikan upaya yang dibutuhkan untuk mengimplementasikan setiap *user story*, unit yang digunakan untuk menggambarkan besar kecilnya upaya itu disebut dengan *story point*. Nilai *story point* sebuah *user story* dipengaruhi oleh kompleksitas, ukuran pekerjaan, dan

ketidakpastian resiko dari *user story* tersebut. Umumnya pengembang menggunakan deret *Fibonacci* sebagai rentang nilai *story point* untuk suatu *user story*, dimulai dari 1, 2, 3, 5, 8, 13, 21. Pada praktiknya pengembang biasanya membagi suatu *story* menjadi tugas-tugas kecil, ketika *story* tersebut dirasa cukup sulit hingga memiliki nilai *story point* 20. Namun untuk *story* yang diberikan estimasi *story point* kurang dari 20 tidak akan dilakukan pemecahan kepadanya. Besaran nilai pada *story point* dapat berupa jam atau hari, pada umumnya menggunakan besaran 1 *story point* sama dengan 2 hari waktu kerja ideal [38][36]. Nilai *story point* ditentukan berdasarkan estimasi pengembang yang menilai tingkat kesulitan setiap *user story* [39].

2. Mengestimasikan prioritas masing-masing *user story*.

Metode yang digunakan peneliti untuk mengestimasikan prioritas *user story* menggunakan pendekatan metode *MoSCoW* yang dibahas pada poin 2.2.9.

3. Mengestimasikan *velocity* dan penyusunan iterasi.

Pada fase ini pengembang menentukan nilai *velocity*. Nilai *velocity* ini merupakan nilai estimasi banyaknya *story point* yang dapat diselesaikan pengembang dalam satu kali *sprint* (aktivitas penggerjaan dalam rentang waktu kerja tertentu). Pada penelitian ini, pengembang memiliki *velocity* sebesar 5 untuk setiap iterasi dengan *sprint-time* selama 10 hari kerja.

Setelah mengestimasikan nilai *story point* dan nilai *velocity*, pengembang kemudian akan menghitung nilai Total Iterasi menggunakan rumus 3.1.

$$\boxed{\mathbf{Total\ Iterasi} = \frac{\mathbf{Total\ stories\ point}}{\mathbf{Velocity}}} \quad (3.1)$$

Keterangan rumus 3.1 :

<i>Total Iterasi</i>	= Jumlah iterasi yang terbentuk untuk semua <i>user stories</i> .
<i>Total stories point</i>	= Jumlah total dari estimasi waktu penggerjaan <i>user stories</i> .
<i>Velocity</i>	= Waktu yang dibutuhkan untuk mengerjakan setiap iterasi.

Pada tahap ini juga berbagai keputusan penting akan diambil seperti penentuan bahasa pemrograman, *framework* pengembangan, serta model aplikasi yang akan dikembangkan.

3.2.4. Metode PXP (3) : Inisialisasi Iterasi

Inisiasi iterasi merupakan tahapan awal sebelum sebuah iterasi dimulai. Iterasi dimulai dengan membentuk pemodelan sistem berdasarkan kebutuhan pada setiap iterasinya yang diperoleh dari hasil perencanaan pada tahap sebelumnya yaitu tahapan perencanaan. Pemodelan sistem dituangkan dalam bentuk *Unified Modeling Language* (UML) berupa:

1. *Use Case Diagram*

Use Case Diagram adalah diagram yang akan merepresentasikan keterkaitan antara salah satu atau lebih aktor dengan sistem. Pembentukan use case diagram diharapkan mampu menggambarkan fungsionalitas sistem yang akan dikembangkan [42].

2. *Activity Diagram*

Activity Diagram adalah diagram yang menggambarkan siklus kerja suatu *use case* [43].

3. *Class Diagram*

Class diagram adalah diagram yang menggambarkan struktur dan deskripsi *class*, dan *object* beserta hubungan satu sama lain seperti *containment*, pewarisan, asosiasi, dan lainnya [42].

3.2.5. Metode PXP (4) : Perancangan

Pada tahapan perancangan, pengembang membuat model rancangan yang akan diimplementasikan selama proses iterasi. Desain yang dirancang hanya memenuhi kebutuhan pengguna yang diperoleh pada tahap analisis kebutuhan. Desain ini berupa antarmuka pengguna dalam bentuk *low fidelity prototype*. Desain dibuat sesederhana mungkin, sehingga antarmuka yang dibuat hanya berdasarkan UML yang sudah dimodelkan pada tahap inisialisasi iterasi.

3.2.6. Metode PXP (5) : Implementasi

Implementasi merupakan tahapan mengeksekusi desain yang dibuat pada tahap perancangan kedalam kode program sehingga dapat dipergunakan menjadi sistem pengadaan di Dinas Pertanian Toba. Tahapan implementasi memiliki tiga tahap yaitu *Unit Testing*, *Code Generation*, dan *Code Refactoring*. *Unit testing* merupakan

pengujian fungsionalitas *code program* dimana sebagian *code program* dituliskan oleh pengembang di awal tahap pengembangan lalu dilakukan pengujian. *Unit testing* melakukan pengujian otomatis menggunakan *library PHP unit*. *Code generation* adalah *code program* setiap fitur yang telah lulus *unit testing* lalu dilanjutkan dengan melengkapi *code program* hingga selesai. Tahap terakhir adalah *refactoring* atau optimasi *code program* [16].

3.2.7. Metode PXP (6) : Pengujian Sistem

Pengujian sistem merupakan pengujian fungsionalitas yang dilakukan terhadap fitur-fitur yang telah diimplementasikan dari setiap iterasi. Pengujian sistem dalam penelitian ini dilakukan dengan menggunakan metode *black box testing* yang telah dijelaskan pada poin 2.2.11.

3.2.8. Metode PXP (7) : Restropektif

Retrospektif adalah tahapan terakhir dari proses iterasi. Pengembang melakukan analisis terhadap pengembangan sistem baik dari kesesuaian estimasi waktu pengerjaan, kendala yang menyebabkan keterlambatan, dan lain sebagainya. Analisis dilakukan untuk mencegah hal yang tersebut terulang kembali pada iterasi selanjutnya.

3.2.9. Evaluasi Sistem

Evaluasi sistem merupakan pengujian fungsionalitas yang dilakukan terhadap fitur-fitur yang telah diimplementasikan setelah seluruh iterasi selesai dilakukan. Evaluasi dalam penelitian ini dilakukan dengan menggunakan metode *System Usability Scale* yang telah dijelaskan pada poin 2.2.12.

3.2.10. Penyusunan Laporan Akhir

Penulisan laporan akhir adalah tahapan menuangkan hasil penelitian kedalam laporan. Laporan akhir ini akan dijadikan salah satu bukti dan syarat bahwa peneliti telah selesai melakukan penelitian terkait pengembangan aplikasi *Automated Essay Scoring* berbasis web.

3.3. Alat dan Bahan Tugas Akhir

Perancangan dan pembangunan aplikasi yang dilakukan dalam penelitian ini membutuhkan alat dan bahan yang digunakan peneliti untuk menunjang penelitian.

3.3.1. Alat

Alat yang digunakan dalam penelitian pengembangan aplikasi ini adalah sebagai berikut:

1. *Software*
 - a. Microsoft word
 - b. Sistem Operasi Windows 10
 - c. Visual Studio Code
 - d. MySQL
 - e. Chrome Web Browser
2. *Hardware*
 - a. Laptop dengan prosesor intel *celeron* dengan RAM 4GB
 - b. Printer
 - c. Flashdisk
 - d. Smartphone
3. *Programming Language*
 - a. PHP
 - b. JavaScript

3.3.2. Bahan

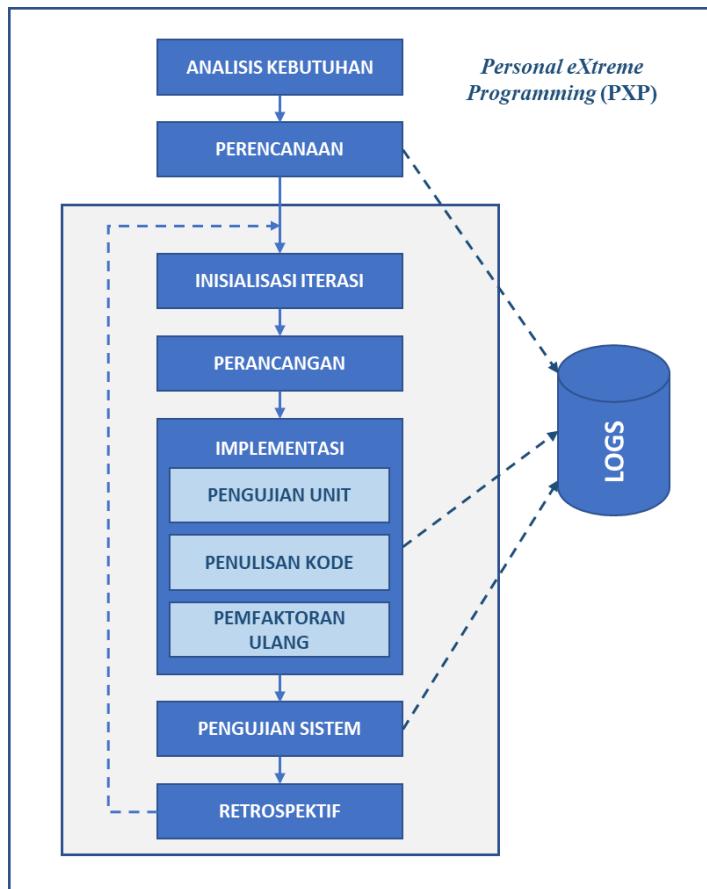
Bahan penelitian yang digunakan peneliti adalah hasil dari wawancara dan observasi yang dilakukan. Bahan-bahan penelitian yang digunakan adalah sebagai berikut:

- a. Data guru dan siswa di SMP Negeri 10 Kotabumi
- b. Data mata pelajaran yang akan dimasukkan ke dalam aplikasi yang dikembangkan.
- c. Data tugas dan ujian dengan soal beserta jawabannya yang akan dimasukkan ke dalam aplikasi yang dikembangkan.

3.4. Metode Tugas Akhir

Metode penelitian yang dilakukan dalam pengembangan aplikasi *Automated Essay Scoring* berbasis web ini adalah *Personal Extreme Programming* (PXP). Metode PXP memiliki tahapan analisis kebutuhan, perencanaan, inisiasi iterasi, perancangan,

implementasi, pengujian sistem dan retrospektif. Metode ini dapat dilihat pada Gambar 3.2.



Gambar 3.2 Fase - fase dalam Metode PXP

Tahapan metode PXP pada Gambar 3.2 akan dijabarkan oleh pengembang sesuai dengan setiap proses yang dilakukan dalam pengembangan aplikasi *Automated Essay Scoring* berbasis web. Berikut adalah penjabaran tahapan metode PXP :

3.4.1. Analisis Kebutuhan

Tahap analisis kebutuhan ini diperoleh dari *client* melalui wawancara dan diskusi bersama ibu Eny Ros selaku Wakil Kepala Bidang Kurikulum di SMP Negeri 10 Kotabumi, dan menyesuaikan kebutuhan aplikasi dengan model AES yang telah dikembangkan oleh PURINO Kecerdasan Buatan ITERA. Hasil yang didapatkan oleh pengembang dituliskan dalam bentuk user stories dengan format “Sebagai <jenis pengguna>, < saya ingin melakukan tindakan sesuatu>, sehingga <mendapatkan manfaat dari tindakan tersebut>”. Deskripsi dari pengguna atau aktor dapat dilihat pada tabel 3.1.

Tabel 3.1 Deskripsi Aktor

No	Aktor	Deskripsi
1	Kepala Sekolah	Seseorang yang menetapkan rencana, merumuskan kebijakan, dan memimpin penyelenggaraan program di SMP Negeri 10 Kotabumi.
2	Guru	Seseorang yang bertugas mengajar mata pelajaran tertentu dan mengevaluasi progres belajar siswa melalui tugas dan ujian.
3	Siswa	Seseorang yang menerima pengajaran dari guru dan bertanggung jawab untuk menyelesaikan tugas dan ujian yang diberikan oleh guru.

Berdasarkan hasil analisis kebutuhan untuk pengembangan aplikasi *Automatic Scoring* berbasis web untuk jawaban teks pada tugas dan ujian di SMP Negeri 10 Kotabumi menggunakan metode *Personal Extreme Programming* (PXP) dituliskan *user stories* yang dapat dilihat pada Tabel 3.2.

Tabel 3.2 User Stories

Kode User Story	User Story	Acceptance Test
Story-01	Sebagai kepala sekolah saya ingin dapat melihat data statistik dari capaian proses belajar mengajar melalui nilai rata-rata tugas dan ujian siswa, sehingga dapat membantu saya dalam menentukan kebijakan untuk menunjang penyelenggaraan proses belajar mengajar	Kepala Sekolah dapat melihat data statistik sederhana terkait capaian proses belajar mengajar dari seluruh siswa di setiap kelas.
Story-02	Sebagai kepala sekolah, saya ingin setiap pengguna sistem ini memiliki akun dengan <i>role</i> -nya masing-masing, sehingga tidak terjadi kesalahan maupun kecurangan dalam proses belajar mengajar.	Seluruh pengguna dapat login ke dalam aplikasi dengan akun dan <i>role</i> -nya masing-masing.
Story-03	Sebagai kepala sekolah saya ingin terdapat role admin yang dapat mengelola data secara keseluruhan, khususnya data akun / pengguna dalam sistem, sehingga seluruh urusan terkait akun dapat ditangani secara terpusat.	Terdapat <i>role</i> admin yang dapat mengelola seluruh data akun.
Story-04	Sebagai guru, saya ingin dapat melakukan pengelolaan kelas, sehingga saya bisa membagikan materi maupun tugas dan ujian kepada siswa.	Guru dapat melakukan pengelolaan kelas.
Story-05	Sebagai guru, saya ingin dapat melakukan pengelolaan tugas dan ujian bagi siswa, beserta dengan pengelolaan pertanyaan yang mendukung	Guru dapat melakukan pengelolaan ujian di dalam sebuah kelas.

Kode User Story	User Story	Acceptance Test
	berbagai tipe soal, dan aplikasi dapat melakukan penilaian otomatis terhadap jawaban siswa, sehingga dapat memudahkan pekerjaan saya sebagai guru.	
Story-06	Sebagai guru, saya ingin dapat melihat dan mengunduh hasil pengerjaan tugas dan ujian oleh siswa, sehingga dapat menjadi bahan evaluasi bagi saya dalam melaksanakan proses belajar mengajar.	Guru dapat melihat dan mengunduh daftar nilai tugas dan ujian siswa.
Story-07	Sebagai guru, saya ingin dapat melihat progres belajar seorang siswa, maupun seluruh siswa dari satu kelas berdasarkan hasil pengerjaan tugas dan ujian siswa bersangkutan, sehingga dapat membantu saya untuk mengevaluasi keaktifan dan kemampuan belajar siswa.	Kepala Sekolah dapat melihat data statistik sederhana terkait capaian proses belajar mengajar dari seluruh siswa di setiap kelas yang diajarnya.
Story-08	Sebagai siswa, saya ingin dapat melihat seluruh hasil dari pengerjaan tugas maupun ujian yang pernah saya lakukan, sehingga saya dapat mengevaluasi aktivitas dan kemampuan belajar saya.	Siswa dapat melihat hasil pengerjaan tugas maupun ujian yang telah dikirimkannya.
Story-09	Sebagai siswa, saya ingin dapat mengunduh materi yang dibagikan oleh guru, sehingga saya dapat mengulangi pelajaran kapanpun saya butuhkan.	Siswa dapat mengunduh materi pelajaran dari seluruh kelas yang ia telah bergabung ke dalamnya.

Berdasarkan Tabel 3.2, didapatkan sebanyak total 9 *user stories* yang merepresentasikan setiap kebutuhan dari aktor / pengguna sistem. Persebaran *user stories* tersebut adalah sebagai berikut :

1. Sebanyak 3 *user stories* untuk aktor Kepala Sekolah
2. Sebanyak 4 *user stories* untuk aktor Guru
3. Sebanyak 2 *user stories* untuk aktor Siswa

3.4.2. Perencanaan

1. Estimasi *User Stories*

Estimasi pengerjaan yang telah ditentukan pengembang berdasarkan tingkat kesulitan pada setiap *user stories* dapat dilihat pada tabel 3.5.

Tabel 3.3 Estimasi *User Stories*

Kode Story	User Story	Story Point	Estimasi Waktu (Hari)
Story-01	Sebagai kepala sekolah saya ingin	3	6

Kode Story	User Story	Story Point	Estimasi Waktu (Hari)
	dapat melihat data statistik dari capaian proses belajar mengajar melalui nilai rata-rata tugas dan ujian siswa, sehingga dapat membantu saya dalam menentukan kebijakan untuk menunjang penyelenggaraan proses belajar mengajar		
Story-02	Sebagai kepala sekolah, saya ingin setiap pengguna sistem ini memiliki akun dengan <i>role</i> -nya masing-masing, sehingga tidak terjadi kesalahan maupun kecurangan dalam proses belajar mengajar.	2	4
Story-03	Sebagai kepala sekolah saya ingin terdapat role admin yang dapat mengelola data secara keseluruhan, khususnya data akun / pengguna dalam sistem, sehingga seluruh urusan terkait akun dapat ditangani secara terpusat.	1	2
Story-04	Sebagai guru, saya ingin dapat melakukan pengelolaan kelas, sehingga saya bisa membagikan materi maupun tugas dan ujian kepada siswa.	3	6
Story-05	Sebagai guru, saya ingin dapat melakukan pengelolaan tugas dan ujian bagi siswa, beserta dengan pengelolaan pertanyaan yang mendukung berbagai tipe soal, dan aplikasi dapat melakukan penilaian otomatis terhadap jawaban siswa, sehingga dapat memudahkan pekerjaan saya sebagai guru.	3	6
Story-06	Sebagai guru, saya ingin dapat melihat dan mengunduh hasil pengerjaan tugas dan ujian oleh siswa, sehingga dapat menjadi bahan evaluasi bagi saya dalam melaksanakan proses belajar mengajar.	2	4
Story-07	Sebagai guru, saya ingin dapat melihat progres belajar seorang siswa, maupun seluruh siswa dari	2	4

Kode Story	User Story	Story Point	Estimasi Waktu (Hari)
	satu kelas berdasarkan hasil penggerjaan tugas dan ujian siswa bersangkutan, sehingga dapat membantu saya untuk mengevaluasi keaktifan dan kemampuan belajar siswa.		
Story-08	Sebagai siswa, saya ingin dapat melihat seluruh hasil dari penggerjaan tugas maupun ujian yang pernah saya lakukan, sehingga saya dapat mengevaluasi aktivitas dan kemampuan belajar saya.	2	4
Story-09	Sebagai siswa, saya ingin dapat mengunduh materi yang dibagikan oleh guru, sehingga saya dapat mengulangi pelajaran kapanpun saya butuhkan.	2	4
Total		20	40

2. Prioritas User Stories

Untuk prioritas dari setiap *user story* sendiri, berikut adalah estimasi pengembang menggunakan aturan MoSCoW seperti ditampilkan pada Tabel .

Tabel 3.4 Prioritas *User Stories*

Kode Story	User Story	Story Point	Prioritas
Story-01	Sebagai kepala sekolah saya ingin dapat melihat data statistik dari capaian proses belajar mengajar melalui nilai rata-rata tugas dan ujian siswa, sehingga dapat membantu saya dalam menentukan kebijakan untuk menunjang penyelenggaraan proses belajar mengajar	3	<i>Should have</i>
Story-02	Sebagai kepala sekolah, saya ingin setiap pengguna sistem ini memiliki akun dengan <i>role</i> -nya masing-masing, sehingga tidak terjadi kesalahan maupun kecurangan dalam proses belajar mengajar.	2	<i>Must have</i>
Story-03	Sebagai kepala sekolah saya ingin	1	<i>Must have</i>

Kode Story	User Story	Story Point	Prioritas
	terdapat role admin yang dapat mengelola data secara keseluruhan, khususnya data akun / pengguna dalam sistem, sehingga seluruh urusan terkait akun dapat ditangani secara terpusat.		
Story-04	Sebagai guru, saya ingin dapat melakukan pengelolaan kelas, sehingga saya bisa membagikan materi maupun tugas dan ujian kepada siswa.	3	<i>Must have</i>
Story-05	Sebagai guru, saya ingin dapat melakukan pengelolaan tugas dan ujian bagi siswa, beserta dengan pengelolaan pertanyaan yang mendukung berbagai tipe soal, dan aplikasi dapat melakukan penilaian otomatis terhadap jawaban siswa, sehingga dapat memudahkan pekerjaan saya sebagai guru.	3	<i>Must have</i>
Story-06	Sebagai guru, saya ingin dapat melihat dan mengunduh hasil pengerjaan tugas dan ujian oleh siswa, sehingga dapat menjadi bahan evaluasi bagi saya dalam melaksanakan proses belajar mengajar.	2	<i>Must have</i>
Story-07	Sebagai guru, saya ingin dapat melihat progres belajar seorang siswa, maupun seluruh siswa dari satu kelas berdasarkan hasil pengerjaan tugas dan ujian siswa bersangkutan, sehingga dapat membantu saya untuk mengevaluasi keaktifan dan kemampuan belajar siswa.	2	<i>Should have</i>
Story-08	Sebagai siswa, saya ingin dapat melihat seluruh hasil dari pengerjaan tugas maupun ujian yang pernah saya lakukan, sehingga saya dapat mengevaluasi aktivitas dan kemampuan belajar saya.	2	<i>Should have</i>
Story-09	Sebagai siswa, saya ingin dapat mengunduh materi yang dibagikan oleh guru, sehingga saya dapat mengulangi pelajaran	2	<i>Should have</i>

Kode Story	User Story	Story Point	Prioritas
	kapanpun saya butuhkan.		

3. Perancanaan Iterasi

Pada tahap ini ditentukan banyaknya iterasi yang akan dilakukan dengan perhitungan menggunakan rumus 3.1 berdasarkan nilai *velocity* pengembang, dan nilai total *story points*.

$$\text{Total Iterasi} = \frac{20}{5}$$

$$\text{Total Iterasi} = 4$$

Berdasarkan nilai *story point*, prioritas dan jumlah total iterasi yang akan dilakukan, didapatkan pembagian iterasi seperti yang ditampilkan pada tabel 3.6 berikut :

Tabel 3.5 Tabel Iterasi

Iterasi 1				
Kode Story	User Story	Priority	Story point	Estimasi waktu (Hari)
Story-02	Sebagai kepala sekolah, saya ingin setiap pengguna sistem ini memiliki akun dengan <i>role</i> -nya masing-masing, sehingga tidak terjadi kesalahan maupun kecurangan dalam proses belajar mengajar.	Must have	2	4
Story-04	Sebagai guru, saya ingin dapat melakukan pengelolaan kelas, sehingga saya bisa membagikan materi maupun tugas dan ujian kepada siswa.	Must have	3	6
<i>Total</i>			5	10
Iterasi 2				
Kode Story	User Story	Priority	Stories point	Estimasi waktu (Hari)
Story-05	Sebagai guru, saya ingin dapat melakukan pengelolaan materi, tugas dan ujian bagi siswa, beserta dengan pengelolaan pertanyaan yang mendukung berbagai tipe soal, dan aplikasi dapat melakukan penilaian otomatis terhadap jawaban siswa,	Must have	3	6

	sehingga dapat memudahkan pekerjaan saya sebagai guru.			
Story-06	Sebagai guru, saya ingin dapat melihat dan mengunduh hasil pengerjaan tugas dan ujian oleh siswa, sehingga dapat menjadi bahan evaluasi bagi saya dalam melaksanakan proses belajar mengajar.	Must have	2	4
<i>Total</i>			5	10
Iterasi 3				
Kode Story	User Story	Priority	Stories point	Estimasi waktu (Hari)
Story-03	Sebagai kepala sekolah saya ingin terdapat role admin yang dapat mengelola data secara keseluruhan, khususnya data akun / pengguna dalam sistem, sehingga seluruh urusan terkait akun dapat ditangani secara terpusat.	Must have	1	2
Story-08	Sebagai siswa, saya ingin dapat melihat seluruh hasil dari pengerjaan tugas maupun ujian yang pernah saya lakukan, sehingga saya dapat mengevaluasi aktivitas dan kemampuan belajar saya.	Should have	2	4
Story-09	Sebagai siswa, saya ingin dapat mengunduh materi yang dibagikan oleh guru, sehingga saya dapat mengulangi pelajaran kapanpun saya butuhkan.	Should have	2	4
<i>Total</i>			5	10
Iterasi 4				
Kode Story	User Story	Priority	Stories point	Estimasi waktu (Hari)
Story-01	Sebagai kepala sekolah saya ingin dapat melihat data statistik dari capaian proses belajar mengajar melalui nilai rata-rata tugas dan ujian siswa, sehingga dapat membantu saya dalam menentukan kebijakan untuk menunjang penyelenggaraan proses belajar mengajar	Should have	3	6
Story-07	Sebagai guru, saya ingin dapat melihat progres belajar seorang siswa, maupun seluruh siswa dari satu kelas berdasarkan hasil pengerjaan tugas dan ujian siswa bersangkutan, sehingga dapat membantu saya untuk	Should have	2	4

	menevaluasi keaktifan dan kemampuan belajar siswa.			
	<i>Total</i>	5	10	

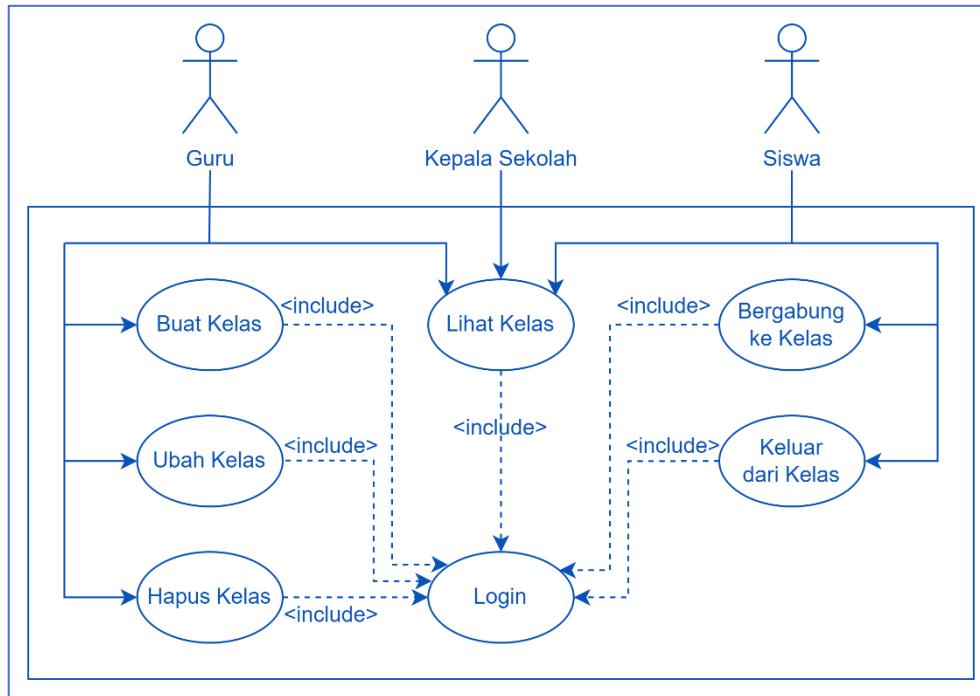
3.4.3. Inisialisasi Iterasi

Pemodelan sistem menggunakan metode *Unified Model Language* (UML) terbagi pada setiap iterasi yang sudah disiapkan. Setiap iterasinya akan di bentuk *use case diagram*, *activity diagram*, dan *class diagram*.

1. Iterasi Pertama

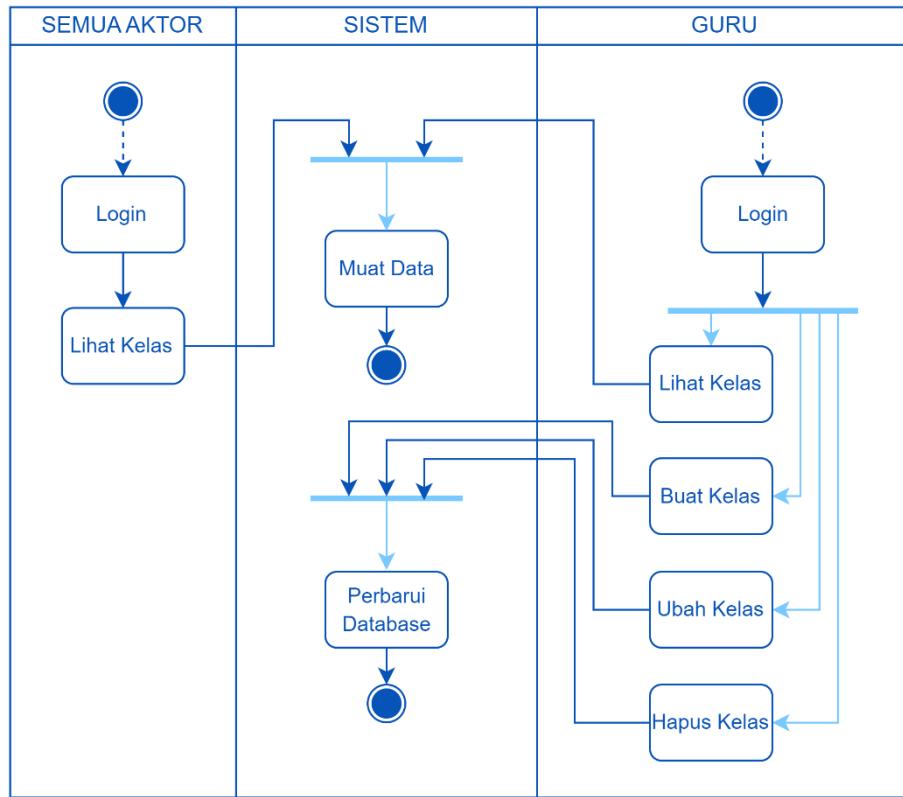
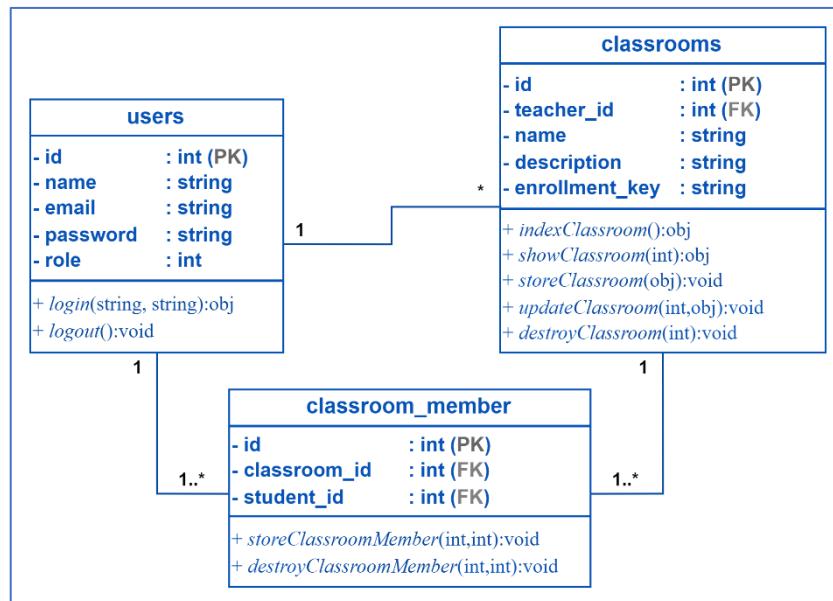
Pada iterasi pertama terdapat *user story* yang menyangkut kebutuhan semua aktor (*Story-02*), yaitu setiap pengguna memiliki akun dengan *role*-nya masing-masing. Umumnya, hal ini akan menjadi prioritas utama dalam pembangunan suatu aplikasi maupun sistem, karena fitur *authentication* dan *authorization* menyangkut setiap user yang berinteraksi dengan aplikasi. Terdapat setidaknya tiga jenis akun atau *role* yang harus tersedia dalam aplikasi ini, antara lain Kepala Sekolah, Guru, dan Siswa.

User story lainnya yang akan dikerjakan pada iterasi pertama adalah *Story-04*. Dengan fitur kelas, pengelolaan tugas dan ujian, bahkan hingga materi oleh guru akan lebih efisien, karena akan menghindarkan siswa dari mengakses materi, tugas, atau ujian yang bukan dimaksud untuknya, maupun menghindarkan guru dari memberikan materi, tugas, atau ujian kepada kelas yang tidak diampunya.



Gambar 3.3 Use Case Diagram Iterasi 1

Setiap aktor dapat menggunakan fitur login untuk mengakses akunnya masing-masing. Fitur ini bertujuan untuk memberikan *authentication* dan *authorization* berdasarkan *role* yang dimiliki pada aplikasi. Untuk pengelolaan kelas, Guru dapat membuat, melihat, mengubah, dan menghapus kelas setelah login ke dalam aplikasi. Sementara untuk fitur melihat kelas, tak hanya Guru, fitur ini dapat dilakukan oleh semua aktor, selama sudah login ke dalam aplikasi. Use case pada Gambar 3.3 ditampilkan dalam bentuk *activity diagram* pada Gambar 3.4.

Gambar 3.4 *Activity Diagram Iterasi 1*Gambar 3.5 *Class Diagram Iterasi 1*

Class diagram pada iterasi pertama seperti ditampilkan pada Gambar 3.5 terdiri dari tiga tabel, yaitu *users*, *classrooms*, dan *classroom_member*.

- 1) Tabel basis data *users*, mewakili entitas pengguna / aktor yang akan menggunakan aplikasi, terdiri dari lima atribut, seperti yang dideskripsikan pada Tabel 3.6 :

Tabel 3.6 Atribut pada tabel *users*

No	Atribut	Tipe data	Keterangan
1	id	bigint	<i>Primary Key</i>
2	name	varchar	Nama pengguna
3	email	varchar	BerNilai unik, berupa alamat email pengguna
4	password	varchar	Kata sandi akun pengguna
5	role	int	Level <i>authorization</i> pengguna, pada iterasi pertama terdiri menjadi tiga pilihan nilai, yaitu : 0 untuk Kepala Sekolah, 1 untuk Guru dan, 2 untuk Siswa.

- 2) Tabel basis data *classrooms*, mewakili entitas kelas yang dapat dikelola oleh Guru dan dapat dilihat oleh Kepala Sekolah, dan Siswa yang telah bergabung, memiliki lima atribut seperti dideskripsikan pada Tabel 3.7 :

Tabel 3.7 Atribut pada tabel *classrooms*

No	Atribut	Tipe data	Keterangan
1	id	bigint	<i>Primary Key</i>
2	teacher_id	bigint	<i>Foreign Key</i> , yang merujuk pada entitas <i>user</i> / pengguna, pada implementasinya, ini menandakan bahwa sebuah kelas dimiliki dan dikelola oleh seorang Guru.
3	name	varchar	Nama kelas
4	deskripsi	varchar	Deskripsi kelas
5	enrollment_key	varchar	Kode pendaftaran yang dapat digunakan siswa untuk bergabung ke kelas

- 3) Tabel *classroom_member*, merupakan *pivot table* untuk menyederhanakan hubungan *many-to-many* antara entitas *user* dan entitas *classroom*, karena seorang Siswa dapat menjadi bagian dari banyak kelas yang berisi banyak

Siswa lainnya. Tabel basis data ini hanya memiliki tiga atribut seperti yang ditampilkan pada Tabel 3.8 :

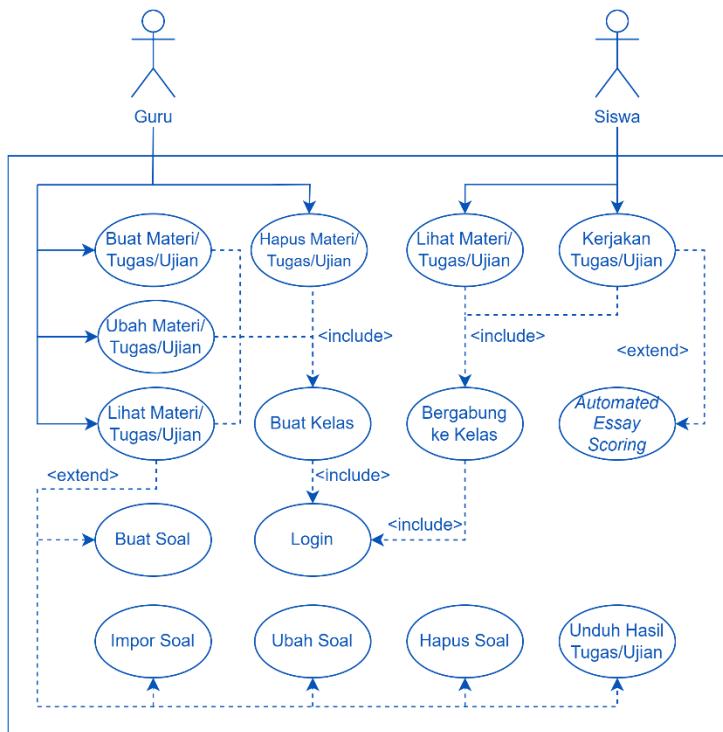
Tabel 3.8 Atribut pada tabel *classroom_member*

No	Atribut	Tipe data	Keterangan
1	id	bigint	<i>Primary Key</i>
2	classroom_id	bigint	<i>Foreign Key</i> , yang merujuk pada entitas <i>classroom</i> .
3	student_id	bigint	<i>Foreign Key</i> , yang merujuk pada entitas <i>user</i> / pengguna.

2. Iterasi Kedua

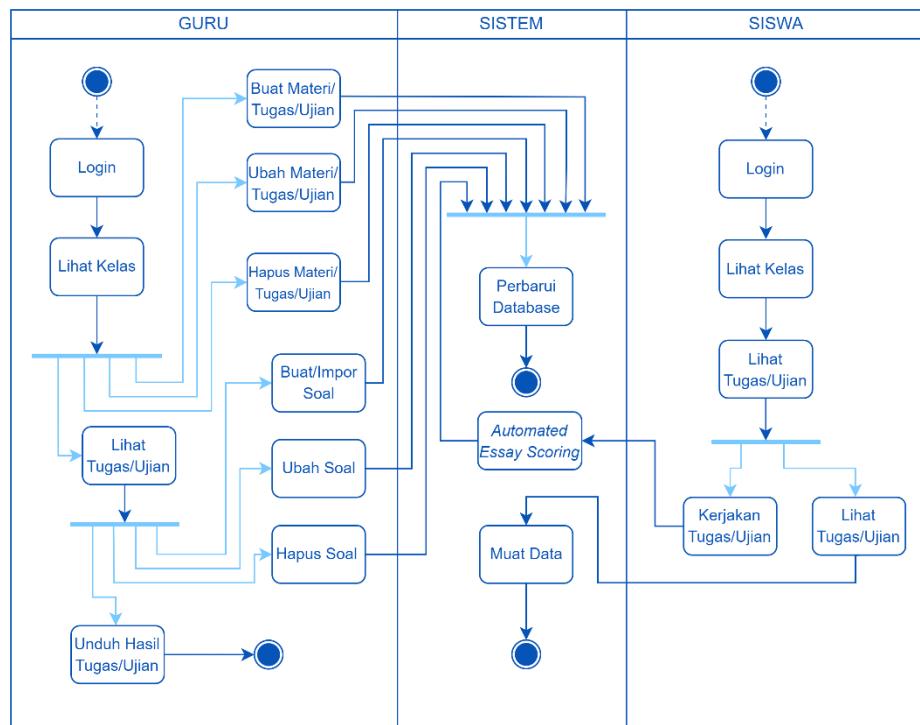
Pada iterasi kedua, akan dikerjakan task berdasarkan *Story-05* yaitu fitur pengelolaan tugas dan ujian yang dapat dilakukan oleh Guru. Dalam mengelola tugas atau ujian, Guru dapat menambahkan pertanyaan yang mendukung berbagai tipe soal, yang dalam penelitian ini dibatasi berupa soal isian yang sesuai dengan topik utama penelitian ini, dan soal pilihan berganda. *Story* ini sekaligus mencakup fitur penggerjaan ujian yang dapat dilakukan oleh siswa, dan mengimplementasikan AES yang menjadi fokus utama dari penelitian ini.

Pada iterasi kedua ini terdapat juga *Story-06*, yang mencakup fitur yang memungkinkan guru untuk melihat dan mengunduh hasil penggerjaan tugas dan ujian oleh siswa. Pada implementasinya, fitur ujian ini akan bisa diakses melalui kelas, seperti yang dijelaskan pada iterasi pertama. Hal ini dilakukan agar seluruh data tugas dan ujian bisa diakses oleh Siswa dan Guru yang bersangkutan saja melalui Kelas yang bersangkutan pula, sehingga aplikasi lebih mudah digunakan, dan data yang ditampilkan kepada pengguna lebih testruktur.

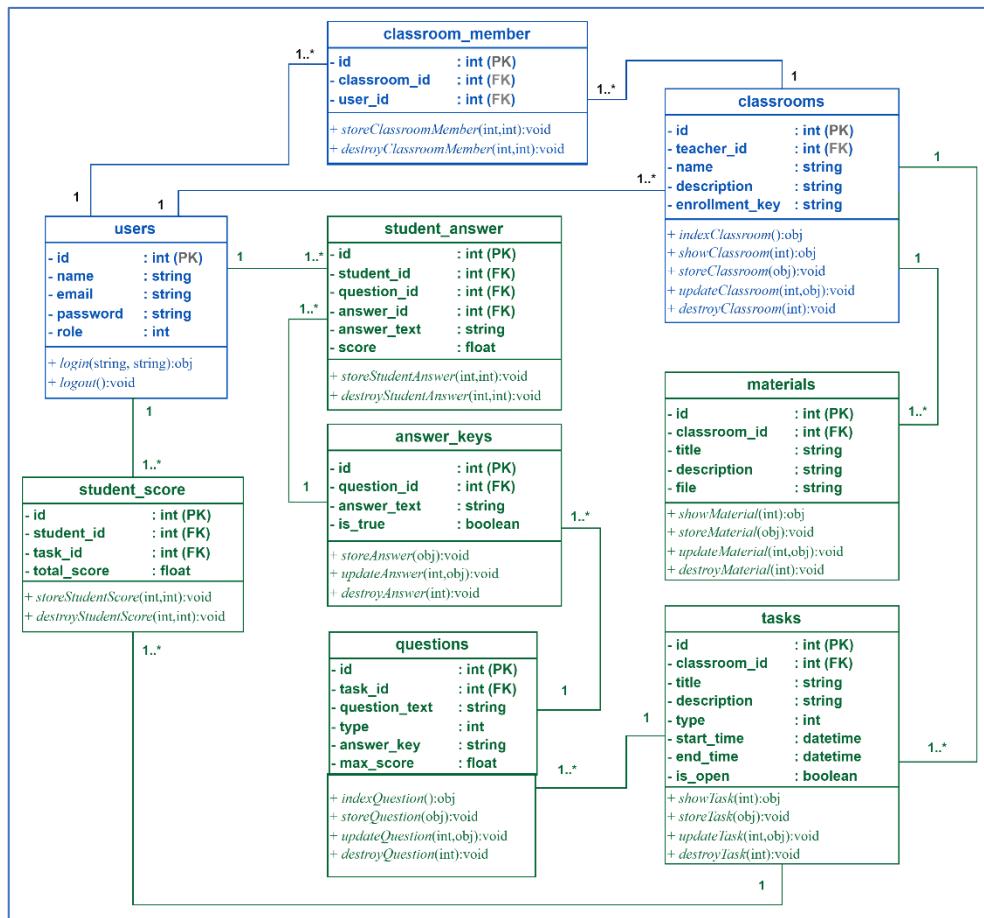


Gambar 3.6 Use Case Diagram Iterasi 2

Setelah Guru membuat kelas, Guru bisa membuat tugas atau ujian, dan melihat, mengubah, maupun menghapus tugas atau ujian yang telah dibuat. Lebih lanjut lagi di dalam tugas/ujian, Guru dapat membuat soal dan kunci jawabannya. Di lain sisi, setelah Siswa bergabung ke dalam kelas, Siswa tersebut bisa melihat maupun mengerjakan tugas atau ujian di dalamnya yang sudah dibuat oleh Guru. Hasil dari penggerjaan ini akan bisa dilihat maupun diunduh oleh guru. *Use case* pada gambar 3.6 ditampilkan dalam bentuk *activity diagram* pada gambar 3.7.



Gambar 3.7 Activity Diagram Iterasi 2



Gambar 3.8 Class Diagram Iterasi 2

Class diagram pada iterasi kedua seperti ditampilkan pada Gambar 3.8, memperbarui *class diagram* pada iterasi pertama yang semula terdiri dari tiga tabel, menjadi 9 tabel yaitu *users*, *classrooms*, *classroom_member*, *materials*, *tasks*, *questions*, *answer_keys*, *student_answer*, dan *student_score*.

Untuk tabel *user*, *classrooms*, dan *classroom_member*, tidak memiliki perbedaan dengan iterasi pertama, sehingga berikut hanya dijelaskan enam tabel lainnya :

- 1) Tabel basis data *materials*, mewakili entitas materi pelajaran yang dapat dikelola oleh Guru dan dapat diakses Siswa, terdiri dari lima atribut, seperti yang dideskripsikan pada Tabel 3.9 :

Tabel 3.9 Atribut pada Tabel *materials*

No	Atribut	Tipe data	Keterangan
1	id	bigint	<i>Primary Key</i>

No	Atribut	Tipe data	Keterangan
2	classroom_id	bigint	<i>Foreign Key</i> , yang merujuk pada entitas <i>classroom</i> , dimana materi ini dikelola oleh Guru.
3	title	varchar	Judul materi
4	description	varchar	Deskripsi materi
5	file	varchar	Alamat <i>file</i> pendukung materi di penyimpanan.

- 2) Tabel basis data *tasks*, mewakili entitas tugas/ujian yang dapat dikelola oleh Guru dan dapat dilihat/dikerjakan oleh Siswa yang berhak, memiliki delapan atribut seperti dideskripsikan pada Tabel 3.10 :

Tabel 3.10 Atribut pada Tabel *tasks*

No	Atribut	Tipe data	Keterangan
1	id	bigint	<i>Primary Key</i>
2	classroom_id	bigint	<i>Foreign Key</i> , yang merujuk pada entitas <i>classroom</i> , dimana tugas/ujian ini dikelola oleh Guru.
3	title	varchar	Judul tugas/ujian
4	description	varchar	Deskripsi tugas/ujian
5	type	integer	Tipe dari penugasan yang diberikan, 0 untuk tugas dan, 1 untuk ujian.
6	start_time	datetime	Waktu mulai tugas/ujian.
7	end_time	datetime	Waktu selesai tugas/ujian.
8	is_open	boolean	Status tugas/ujian, <i>true</i> untuk terbuka, artinya tugas/ujian masih bisa dikerjakan <i>false</i> untuk tertutup, artinya tugas/ujian sudah tidak dapat dikerjakan.

- 3) Tabel *questions*, mewakili entitas pertanyaan yang dapat dikelola oleh Guru dalam suatu tugas/ujian dan dapat dilihat oleh Siswa saat mengerjakan tugas/ujian, memiliki lima atribut seperti dideskripsikan pada tabel 3.11 :

Tabel 3.11 Atribut pada Tabel *questions*

No	Atribut	Tipe data	Keterangan
1	id	bigint	<i>Primary Key</i>
2	task_id	bigint	<i>Foreign Key</i> , yang merujuk pada entitas <i>task</i> .
3	question_text	string	Teks pertanyaan.
4	type	int	Tipe pertanyaan, dengan dua pilihan nilai : 0 untuk <i>essay</i> , 1 untuk pilihan ganda.
5	answer_key	string	Kunci jawaban yang hanya diisi ketika pertanyaan bertipe <i>essay</i> .
6	max_score	float	Nilai maksimum yang bisa didapatkan Siswa bila jawabannya untuk pertanyaan ini benar.

- 4) Tabel basis data *answer_keys*, mewakili entitas kunci jawaban yang terhubung ke entitas *question*-nya masing-masing, entitas ini dapat dikelola oleh Guru dan dapat dilihat oleh Siswa ketika mengerjakan tugas/ujian. Entitas ini bersifat kondisional bergantung pada entitas *question*-nya, bila *question*-nya bertipe *essay* maka nilai dari atribut *answer_text* miliknya yang akan digunakan sebagai pembanding pada model AES untuk melakukan penilaian otomatis. Sementara bila *question*-nya bertipe pilihan ganda, maka yang perlu diperhatikan adalah nilai dari atribut *is_true* miliknya, yang akan menentukan apakah jawaban Siswa benar atau salah. Pada entitas ini terdiri dari empat atribut, seperti yang dideskripsikan pada tabel 3.12 :

Tabel 3.12 Atribut pada Tabel *answer_keys*

No	Atribut	Tipe data	Keterangan
1	id	bigint	<i>Primary Key</i>
2	question_id	bigint	<i>Foreign Key</i> , yang merujuk pada entitas <i>question</i> , dimana jawaban ini berlaku.
3	answer_text	varchar	Berisi teks jawaban.
4	is_true	boolean	Status apakah jawaban ini bernilai

No	Atribut	Tipe data	Keterangan
			benar/atau salah, untuk pertanyaan bertipe pilihan ganda.

- 5) Tabel basis data *student_answer*, merupakan tabel yang mewakili jawaban Siswa, yang bersifat kondisional. Ketika pertanyaan bertipe *essay*, maka atribut *answer_id* harus bernilai *null*, sehingga ketika sistem mendeteksi bahwa *answer_id* bernilai *null*, maka sistem akan melakukan penilaian menggunakan model *AES* dengan menggunakan atribut *answer_text* pada entitas ini dan mengakses atribut *answer_key* pada entitas *question* melalui atribut *question_id* pada entitas ini, kemudian dilakukan perbandingan. Demikian sebaliknya, ketika pertanyaan bertipe pilihan ganda, maka atribut *answer_id* harus diisi, dan atribut *answer_text* dan atribut *question_id* harus bernilai *null*, dan sistem akan secara otomatis mengecek nilai atribut *is_true* pada entitas *answer_key* menggunakan *answer_id* yang diberikan, untuk menentukan apakah jawaban Siswa tersebut benar atau salah.

Tabel basis data ini memiliki lima atribut seperti yang ditampilkan pada tabel 3.13 :

Tabel 3.13 Atribut pada Tabel *student_answer*

No	Atribut	Tipe data	Keterangan
1	<i>id</i>	bigint	<i>Primary Key</i>
2	<i>student_id</i>	bigint	<i>Foreign Key</i> , yang merujuk pada entitas <i>user</i> , yang mengirimkan jawaban ini.
3	<i>question_id</i>	bigint	<i>Foreign Key</i> , yang merujuk pada entitas <i>question</i> , yang menjadi pasangan pertanyaan jawab ini, hanya diisi bila pertanyaan bertipe <i>essay</i> .
4	<i>answer_id</i>	varchar	<i>Foreign Key</i> , yang merujuk pada entitas <i>answer_key</i> , bila diisi pada pertanyaan bertipe pilihan ganda.
5	<i>answer_text</i>	varchar	Teks jawaban yang dikirimkan Siswa, pada pertanyaan bertipe <i>essay</i> .

No	Atribut	Tipe data	Keterangan
6	score	float	Tipe dari penugasan yang diberikan, 0 untuk tugas dan, 1 untuk ujian.

- 6) Tabel *student_score*, mewakili entitas total skor yang didapatkan oleh seorang Siswa pada suatu tugas/ujian. Entitas ini dibuat ketika seorang siswa mengakhiri dan telah mengirimkan seluruh jawaban pada suatu tugas/ujian, dengan mengecek seluruh entitas *student_answer* yang tersedia yang berhubungan dengan entitas *user* (Siswa) dan *question* yang diakses melalui atribut *student_id* dan *question_id*, kemudian menjumlahkan seluruh atribut *score* dari mereka menjadi nilai yang akan disimpan pada atribut *total_score* pada entitas ini. Entitas ini memiliki empat atribut seperti dideskripsikan pada tabel 3.8 :

Tabel 3.14 Atribut pada Tabel *student_score*

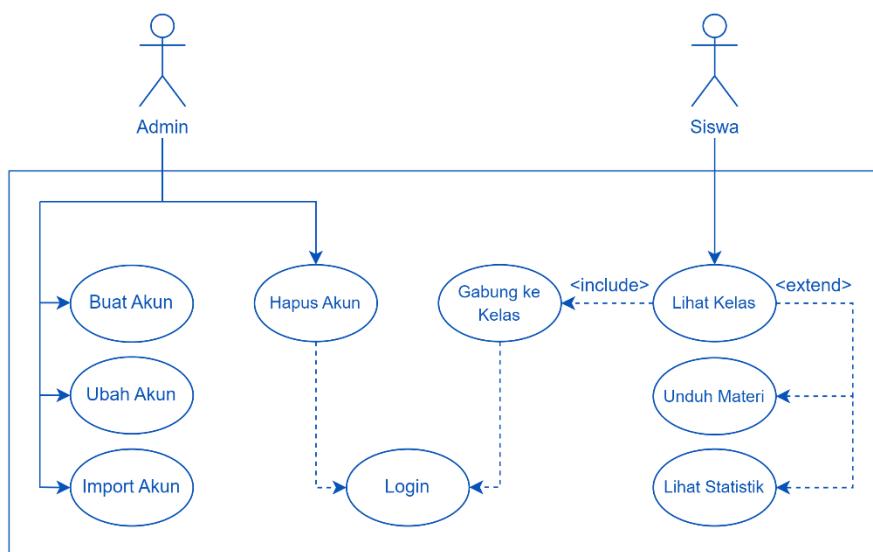
No	Atribut	Tipe data	Keterangan
1	id	bigint	<i>Primary Key</i>
2	student_id	bigint	<i>Foreign Key</i> , yang merujuk pada entitas <i>user</i> (Siswa).
3	question_id	bigint	<i>Foreign Key</i> , yang merujuk pada entitas <i>question</i> .
4	total_score	float	Total nilai yang didapatkan Siswa pada tugas/ujian terkait.

3. Iterasi Ketiga

Pada iterasi ketiga, terdapat user story yang menyangkut semua aktor (*Story-03*), yaitu adanya role Admin, yang dapat mengelola data secara keseluruhan, khususnya data akun. Pada implementasinya, di aplikasi ini tidak akan tersedia fitur pendaftaran akun pribadi oleh pengguna, karena seluruh akun akan didaftarkan sendiri oleh Admin. Hal ini dirasa cukup baik dalam meminimalisir penggunaan penyimpanan pada server, dan mempersingkat proses perbaikan bila terjadi kesalahan dalam proses autentikasi pengguna lainnya. Selain itu, untuk lebih mempermudah pekerjaan Admin, dibuatkan fitur Impor Akun, untuk

memungkinkan Admin membuat banyak akun sekaligus dengan mengunggah file .excel dengan format tertentu ke dalam aplikasi.

User story lainnya yang akan dikerjakan pada iterasi pertama adalah *Story-08* dan *Story-09*. Pada kedua *story* tersebut, Siswa akan dimungkinkan untuk melihat data statistik sederhana terkait, hasil penggeraan tugas dan ujian yang telah ia lakukan dalam satu kelas secara keseluruhan, selain itu Siswa juga akan dimungkinkan untuk mengunduh materi yang dibagikan oleh guru di kelas.

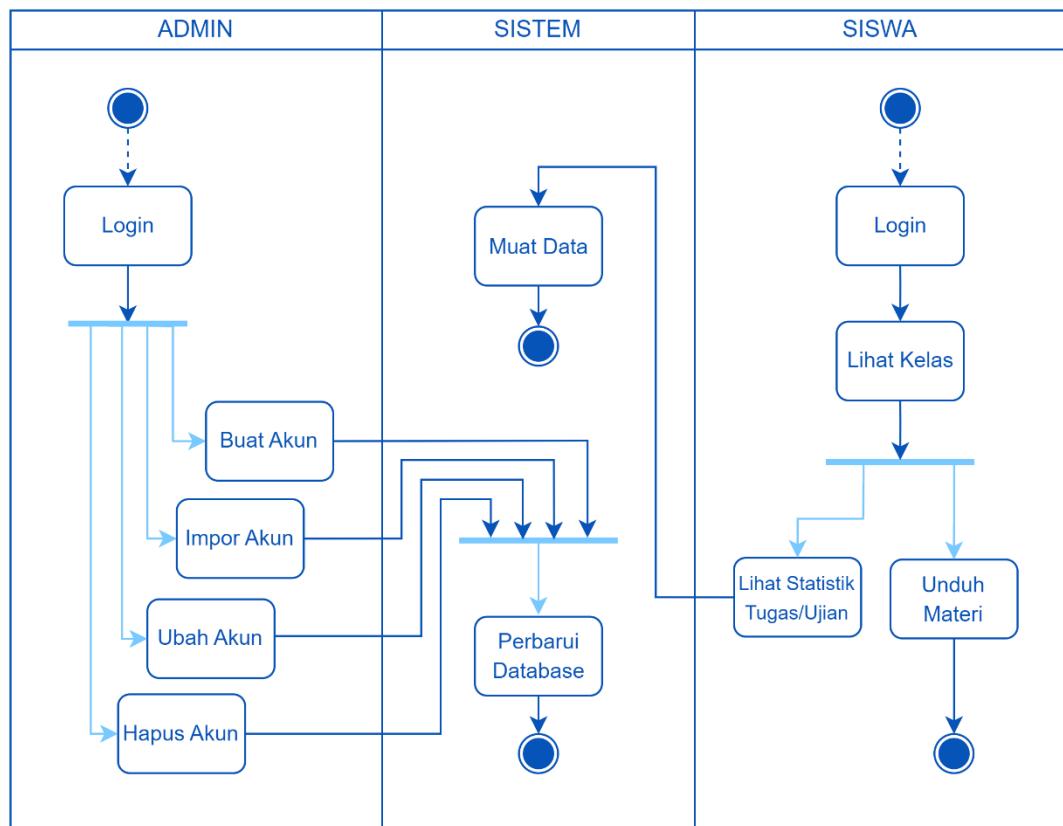


Gambar 3.9 Use Case Diagram Iterasi 3

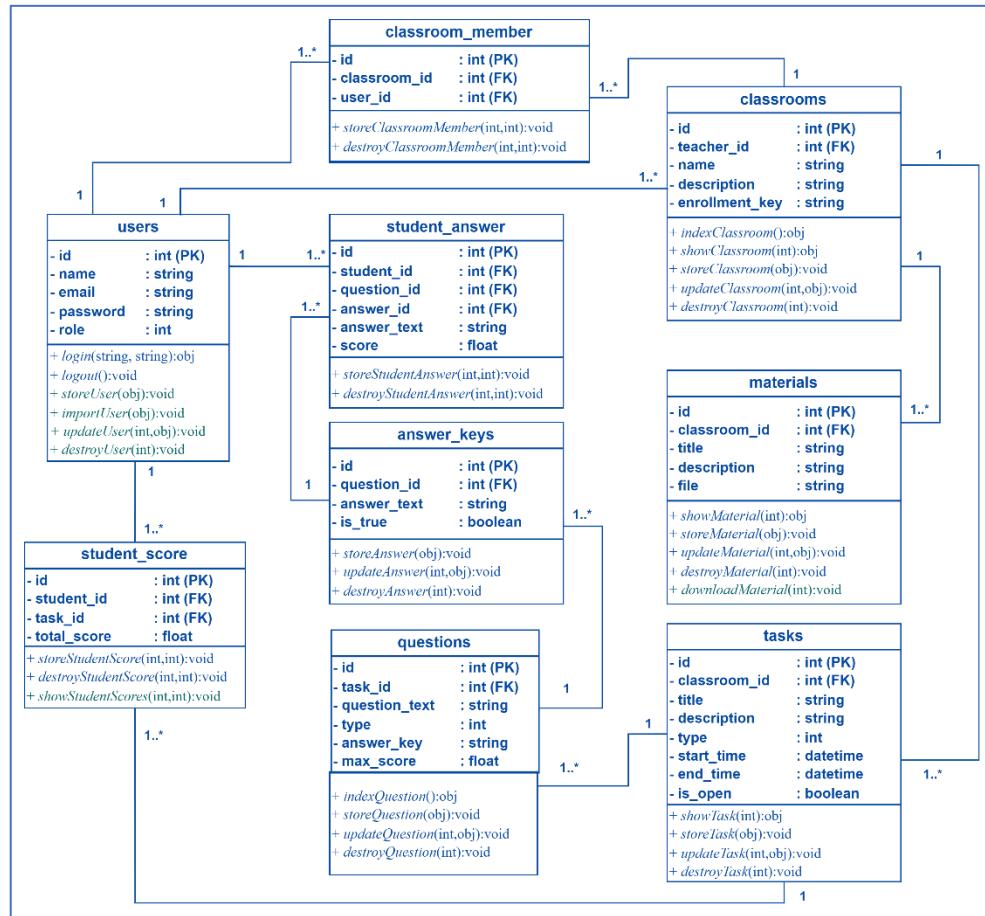
Pada iterasi ketiga ini, melalui Gambar 3.9 dapat dilihat peran admin, dimana admin dapat mengelola akun, meliputi membuat, mengimpor, mengubah, maupun menghapus akun. Dari Use Case tersebut dapat dilihat bahwa Admin pun harus login terlebih dahulu sebelum dapat mengelola akun lainnya, dengan demikian, khusus untuk akun Admin, akan dibuatkan secara *hard-coded* pada masa pengembangan.

Di lain sisi, dapat dilihat pada iterasi ini Siswa sudah dapat mengunduh materi yang dibagikan oleh guru setelah bergabung ke kelas, dan tentunya setelah login menggunakan akunnya. Selain mengunduh materi Siswa juga dimungkinkan untuk melihat data statistik sederhana dari seluruh hasil penggerjaan tugas dan

ujian yang pernah ia lakukan. *Use case* pada Gambar 3.9 ditampilkan dalam bentuk *activity diagram* pada Gambar 3.10.



Gambar 3.10 *Activity Diagram Iterasi 3*

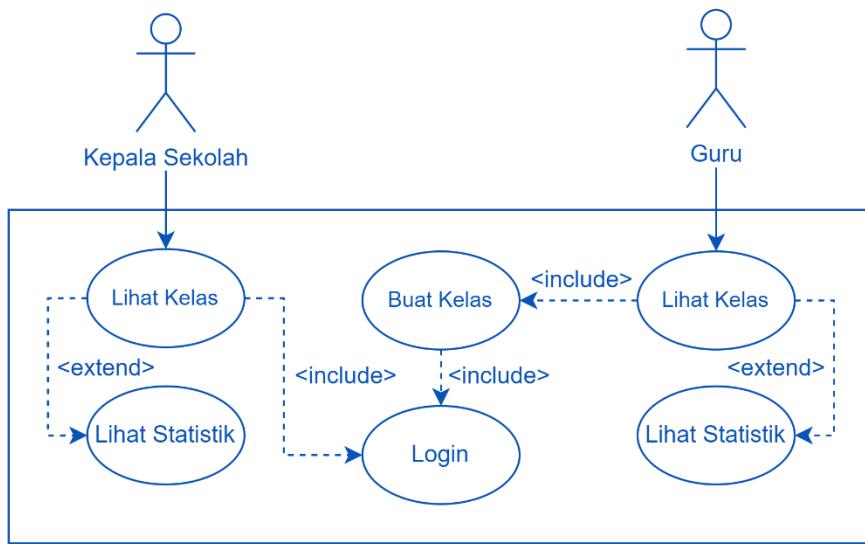


Gambar 3.11 *Class Diagram* Iterasi 3

Pada iterasi ketiga, tidak terdapat penambahan tabel basis data seperti ditampilkan pada Gambar 3.11, namun terdapat penambahan fungsi pada tiga tabel yaitu *users*, *materials*, dan *student_score*.

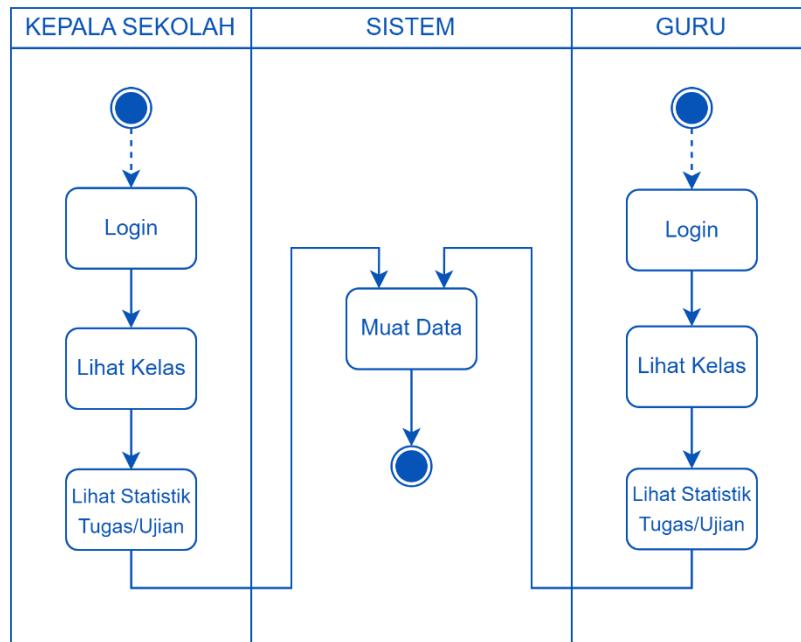
4. Iterasi Keempat

Pada iterasi keempat, akan dikerjakan task berdasarkan *Story-01* dan *Story-07* yang memungkinkan Kepala Sekolah dan Guru untuk melihat data statistik sederhana terkait capaian belajar seorang siswa, maupun seluruh siswa dalam satu kelas, dari keaktifan dan hasil penggerjaan tugas/ujian dari Siswa tersebut.



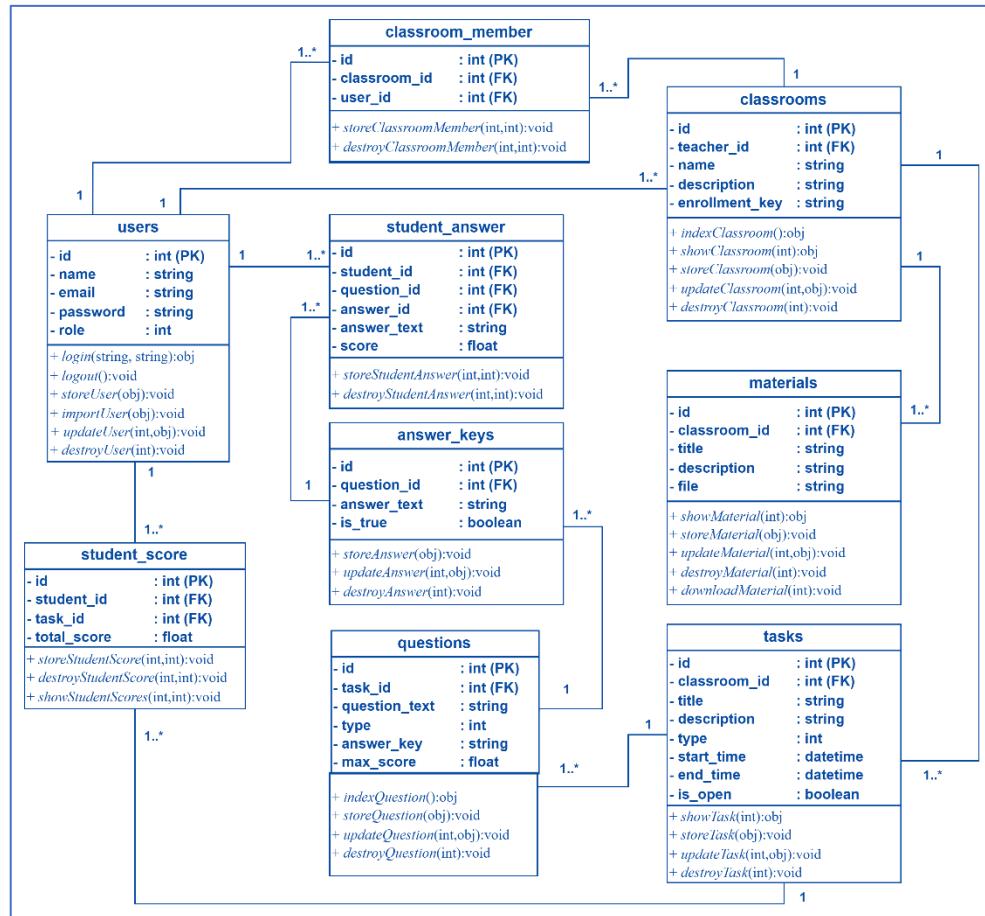
Gambar 3.12 *Use Case Diagram* Iterasi 4

Pada iterasi keempat ini, melalui Gambar 3.10 dapat dilihat bahwa Kepala Sekolah dan Guru dapat melihat statistik capaian belajar dari Siswa. *Use case* pada gambar 3.10 ditampilkan dalam bentuk *activity diagram* pada gambar 3.11.



Gambar 3.13 *Activity Diagram* Iterasi 4

Pada iterasi keempat, tidak terdapat perubahan pada tabel-tabel basis data dari iterasi ketiga, dapat dilihat di *class diagram* pada gambar 3.12.

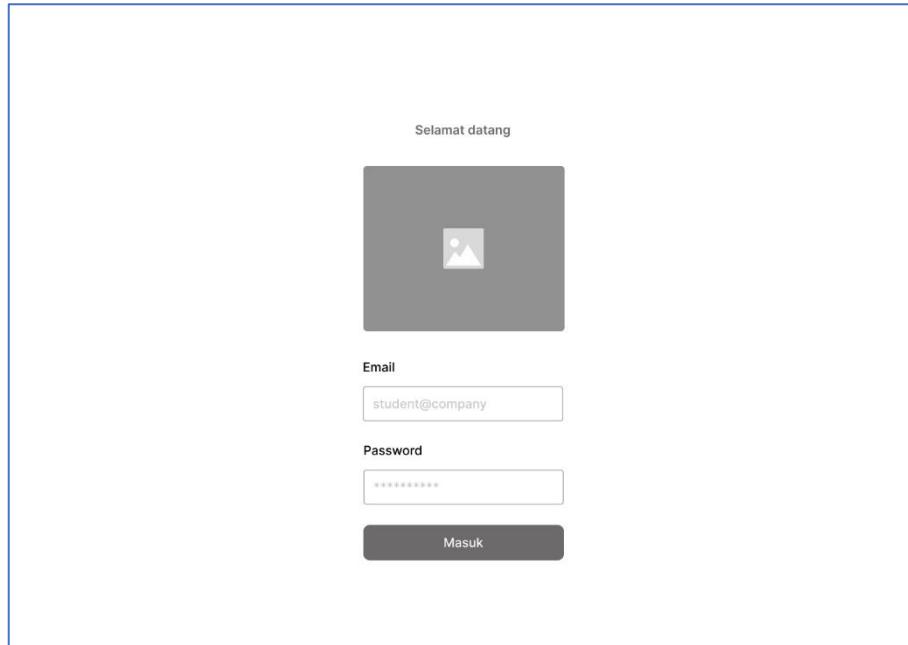


Gambar 3.14 Class Diagram Iterasi 4

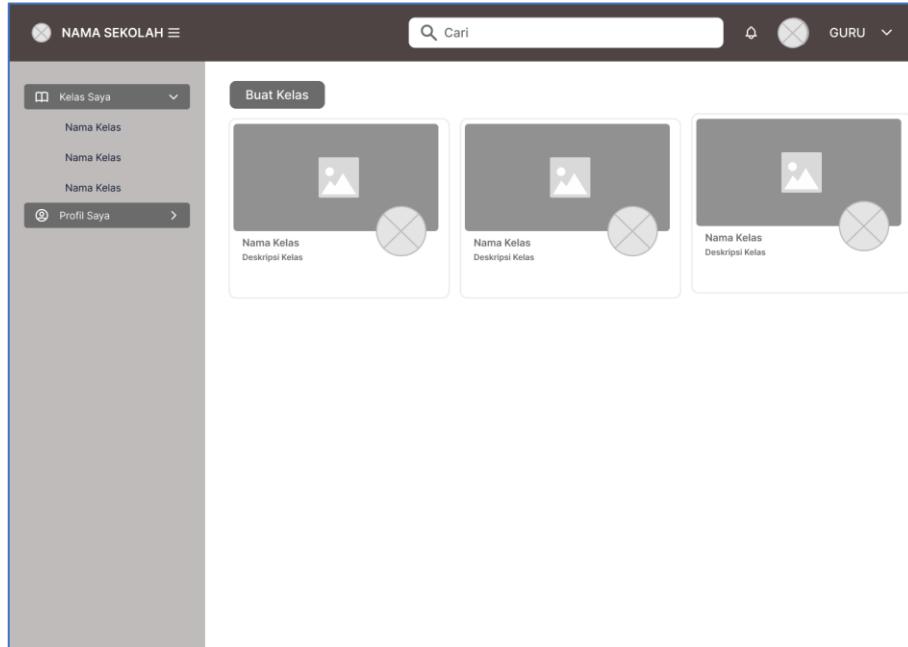
3.4.4. Perancangan

Pada tahap perancangan, peneliti yang sekaligus pengembang membuat desain untuk semua *user story* dalam pengembangan aplikasi ini. Desain dibuat sederhana untuk memodelkan sebuah iterasi yang sedang berlangsung. Pengembang memodelkan dengan membuat *low fidelity prototype* untuk setiap iterasi.

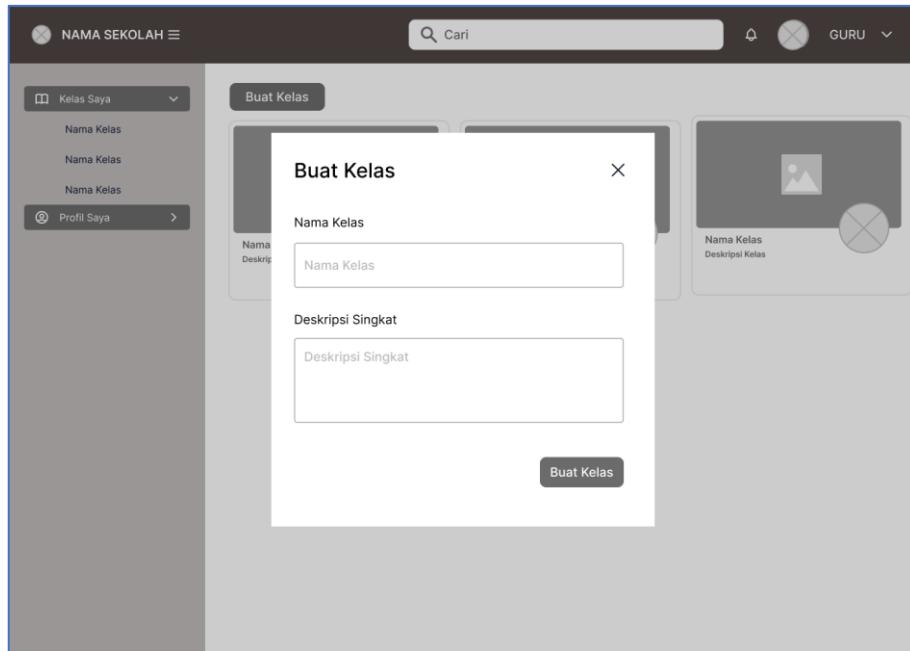
1. Iterasi Pertama



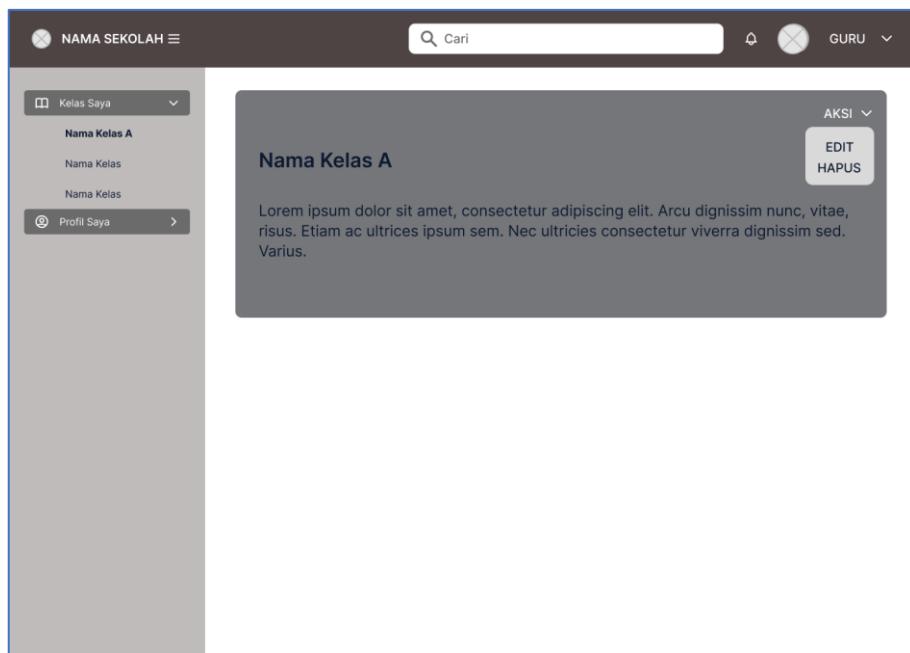
Gambar 3.15 Halaman Login Seluruh User



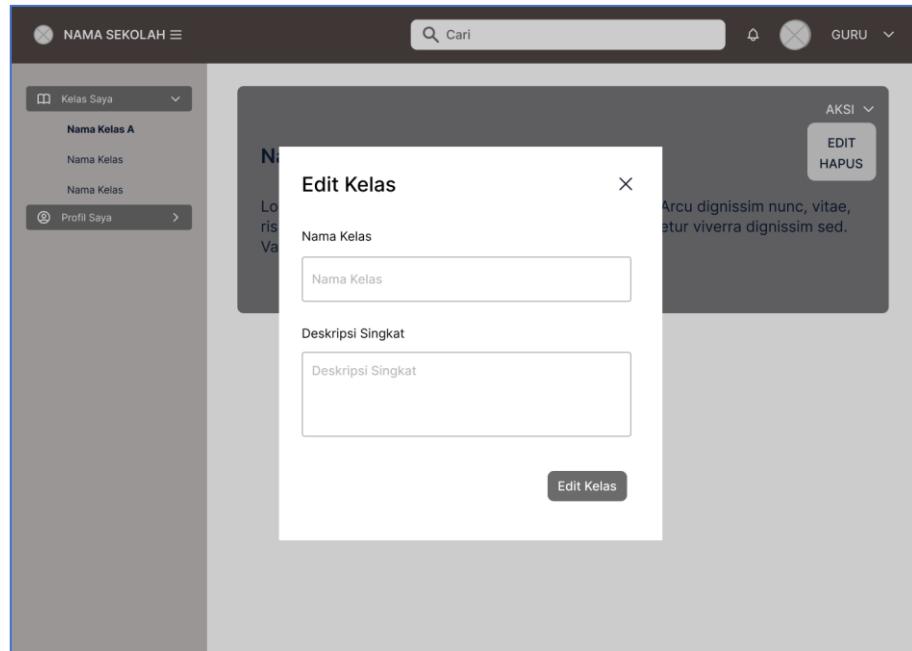
Gambar 3.16 Halaman Awal Guru Setelah Login



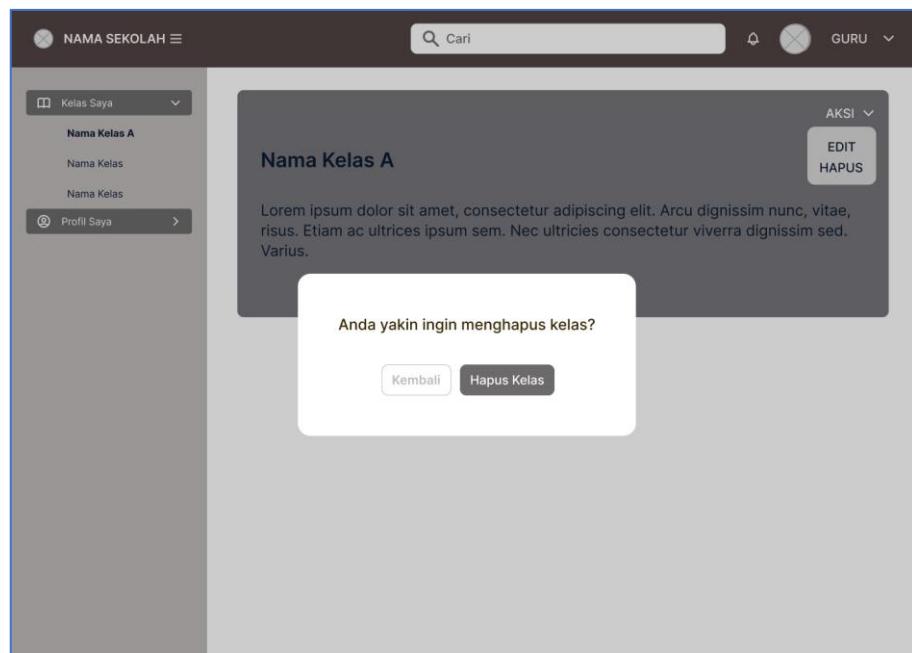
Gambar 3.17 Tampilan Modal “Buat Kelas” untuk Guru



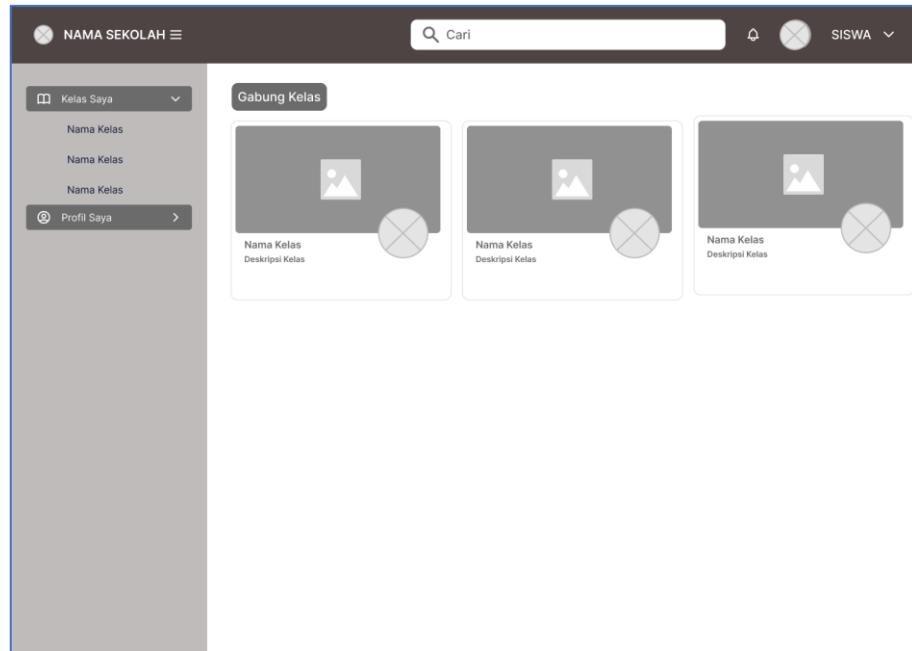
Gambar 3.18 Tampilan “Lihat Kelas” untuk Guru



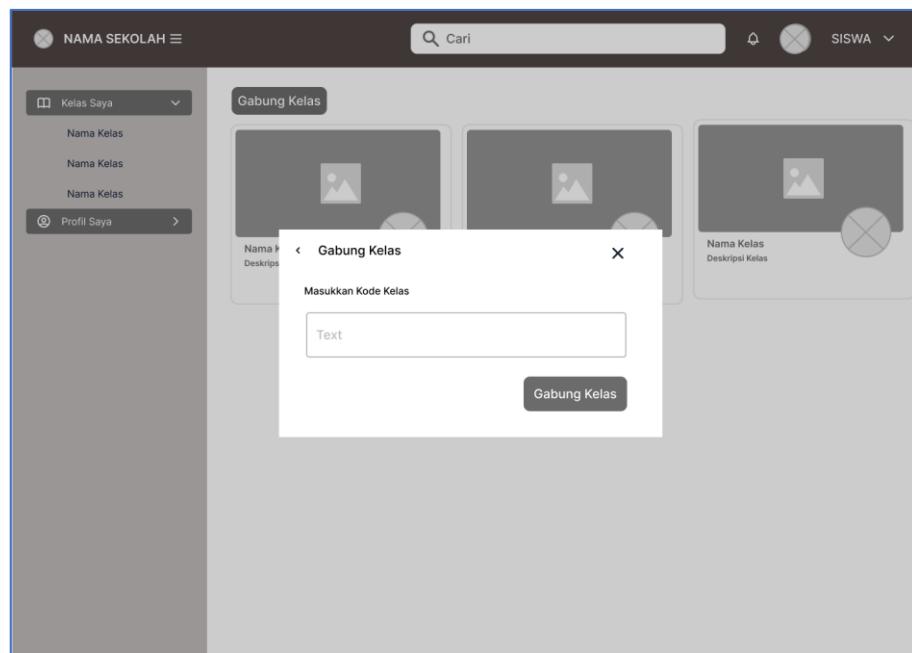
Gambar 3.19 Tampilan “Edit Kelas” untuk Guru



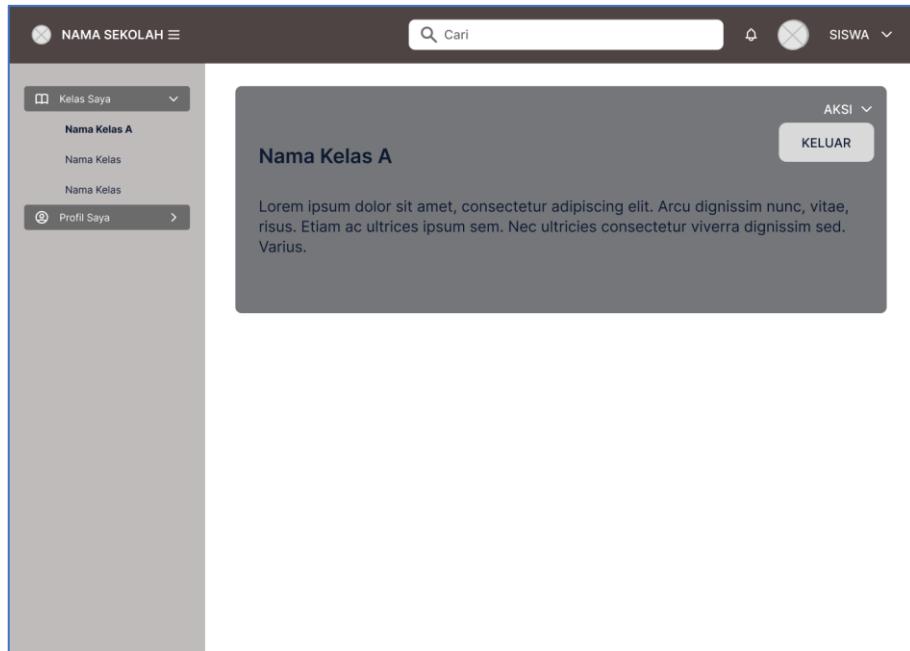
Gambar 3.20 Tampilan Konfirmasi "Penghapusan Kelas" untuk Guru



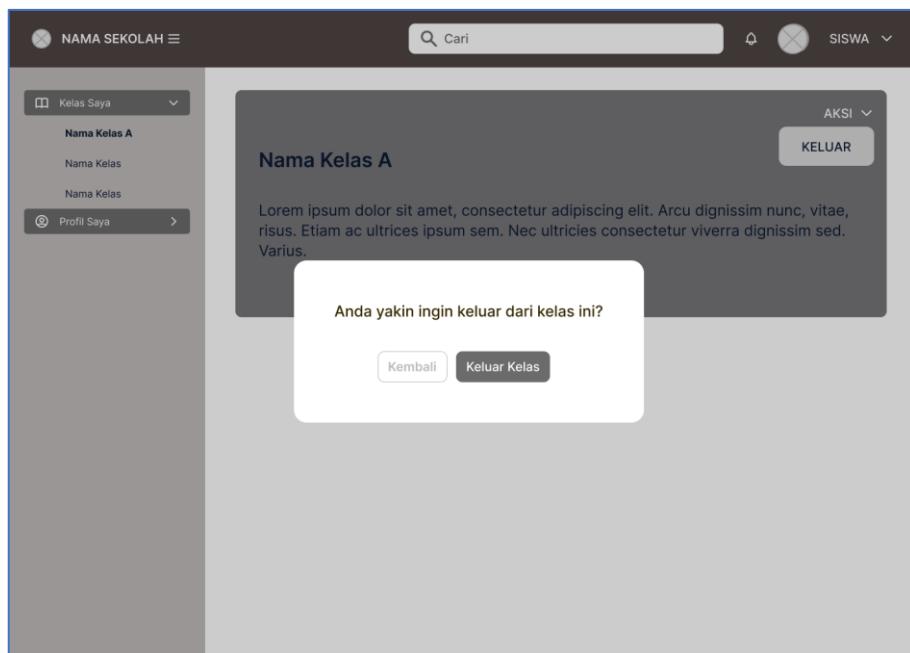
Gambar 3.21 Halaman Awal Siswa Setelah Login



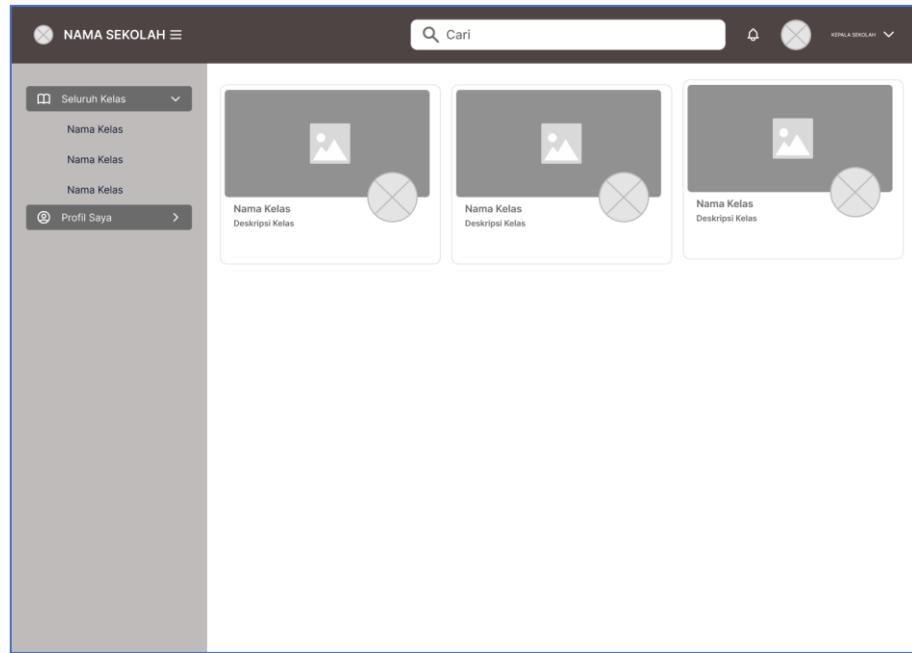
Gambar 3.22 Tampilan "Gabung Kelas" untuk Siswa



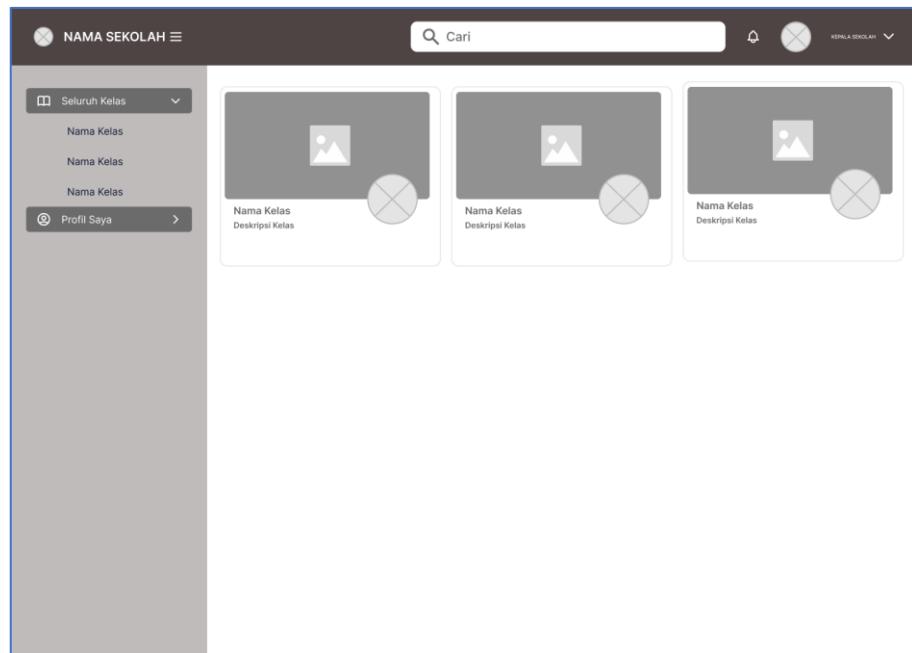
Gambar 3.23 Tampilan "Lihat Kelas" untuk Siswa



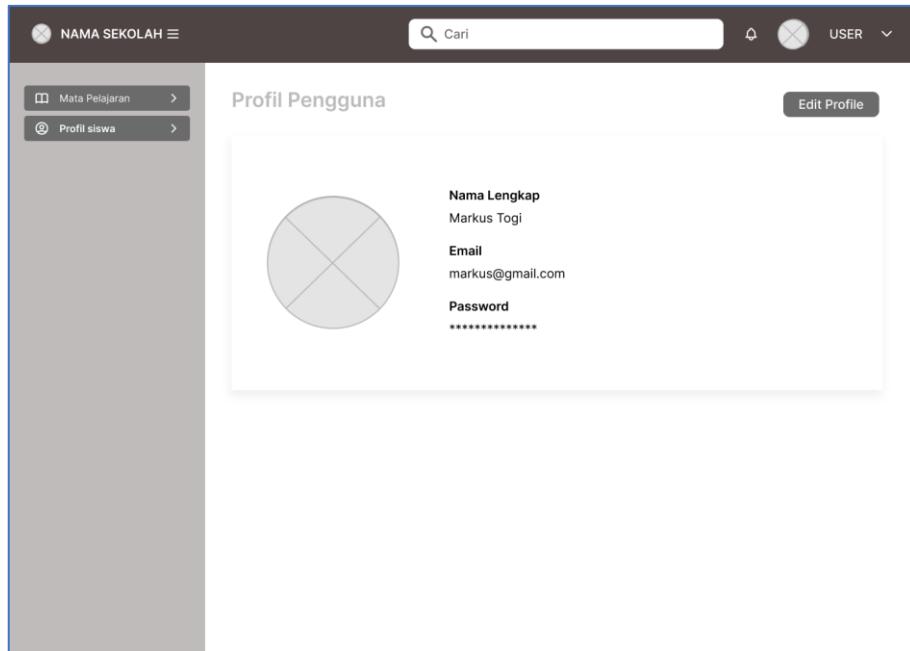
Gambar 3.24 Tampilan Konfirmasi "Keluar Kelas" untuk Siswa



Gambar 3.25 Tampilan Awal Kepala Sekolah setelah Login



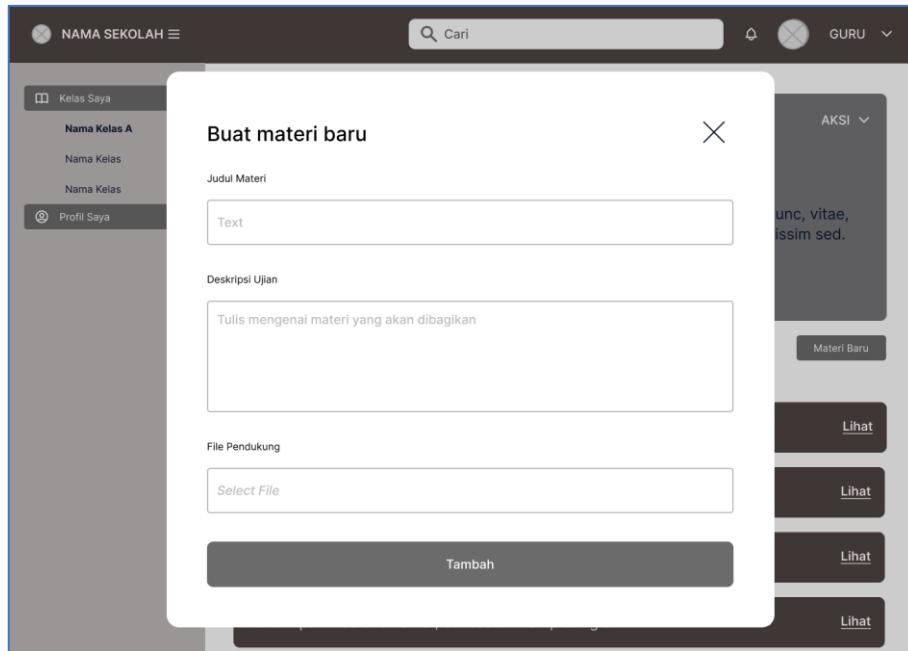
Gambar 3.26 Tampilan "Lihat Kelas" untuk Kepala Sekolah



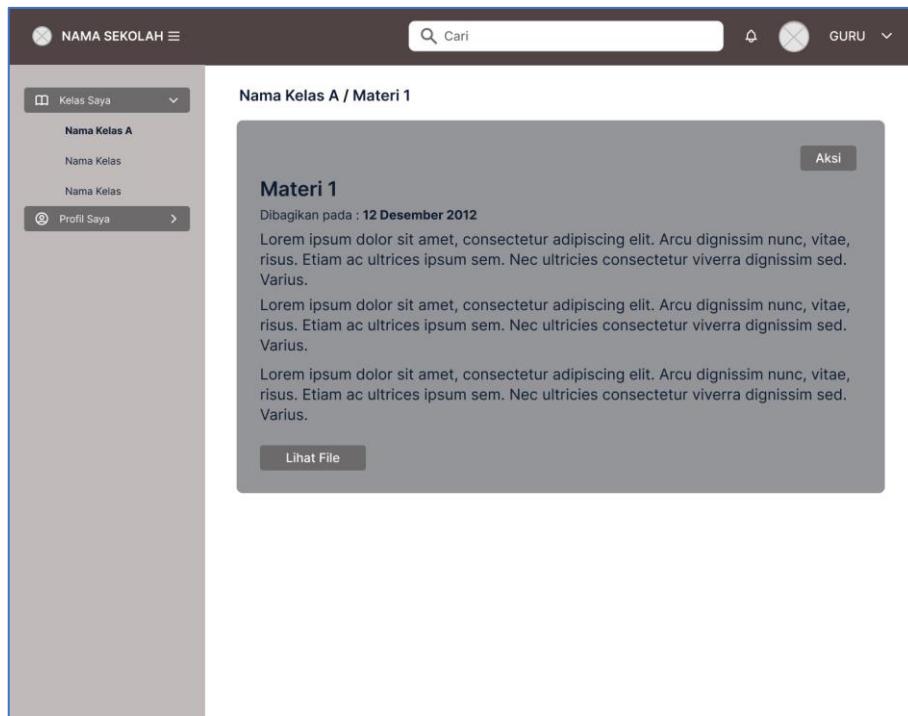
Gambar 3.27 Halaman Profil untuk Seluruh Guru dan Siswa

2. Iterasi Kedua

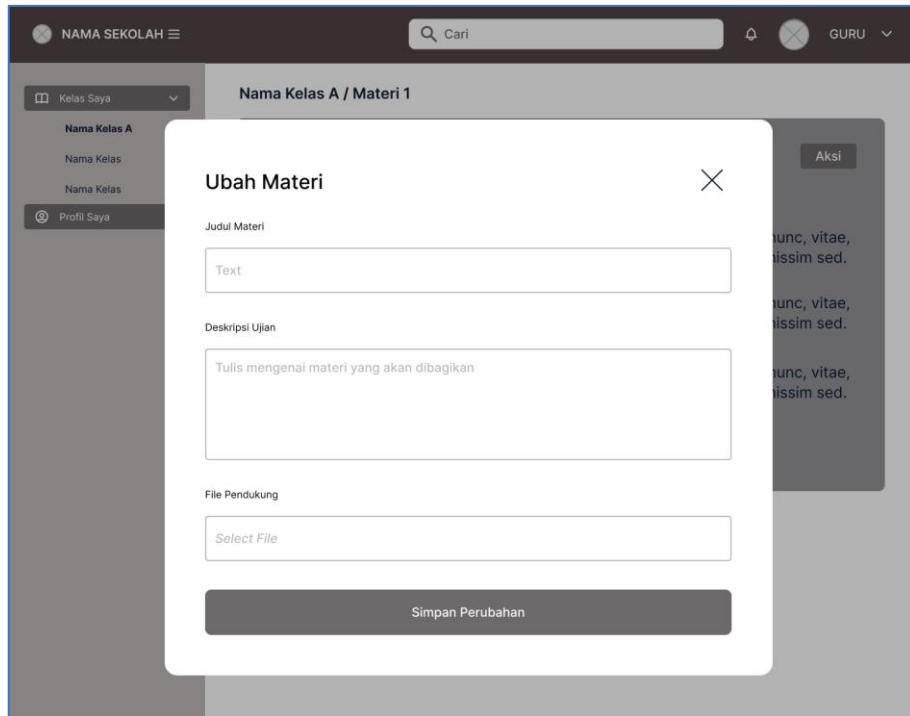
Gambar 3.28 Tampilan “Daftar Materi” untuk Guru



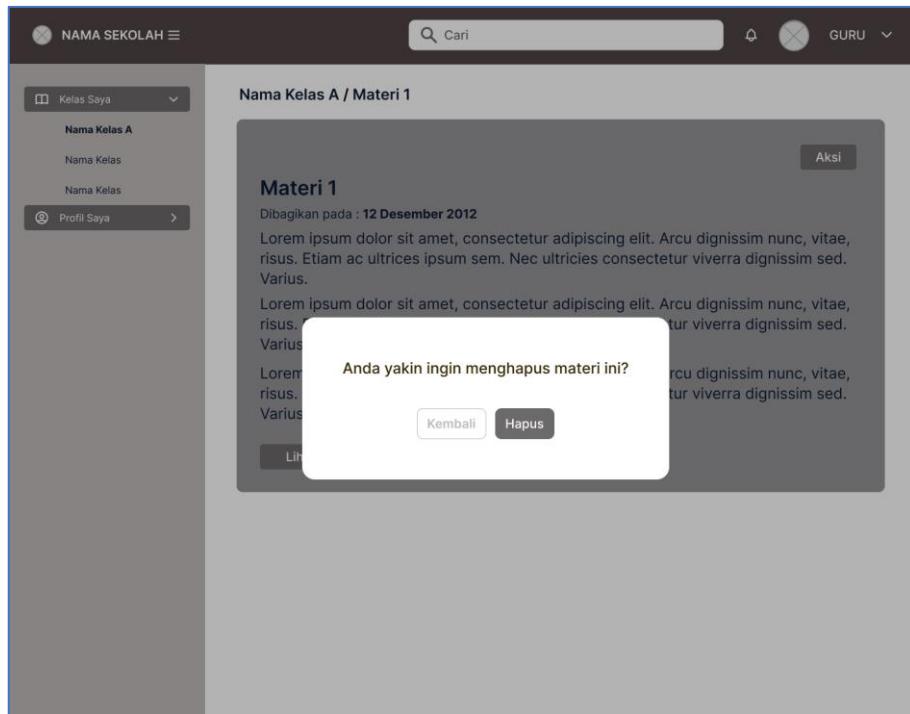
Gambar 3.29 Tampilan “Buat Materi” Baru untuk Guru



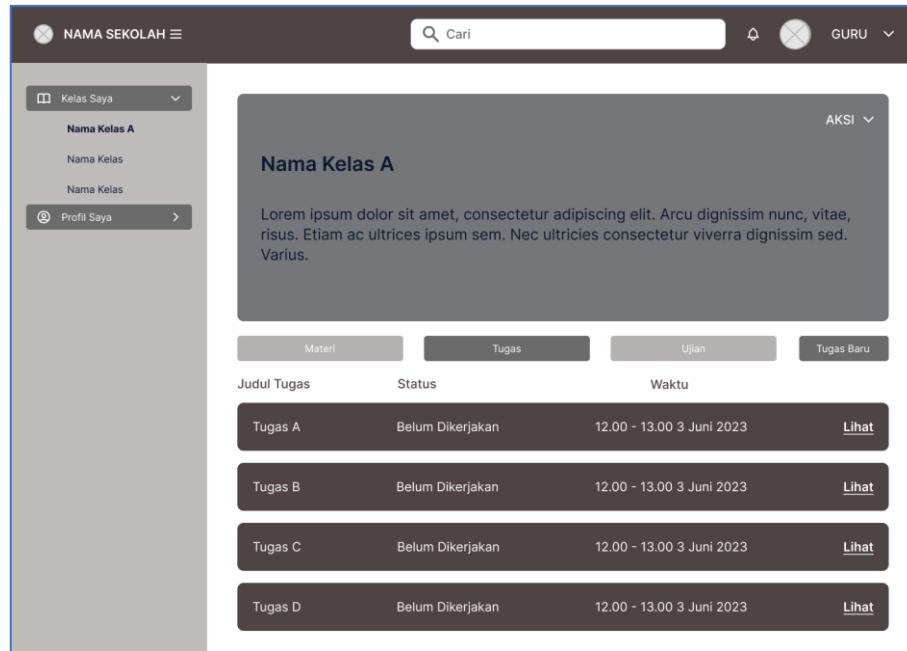
Gambar 3.30 Halaman “Detail Materi” untuk Guru



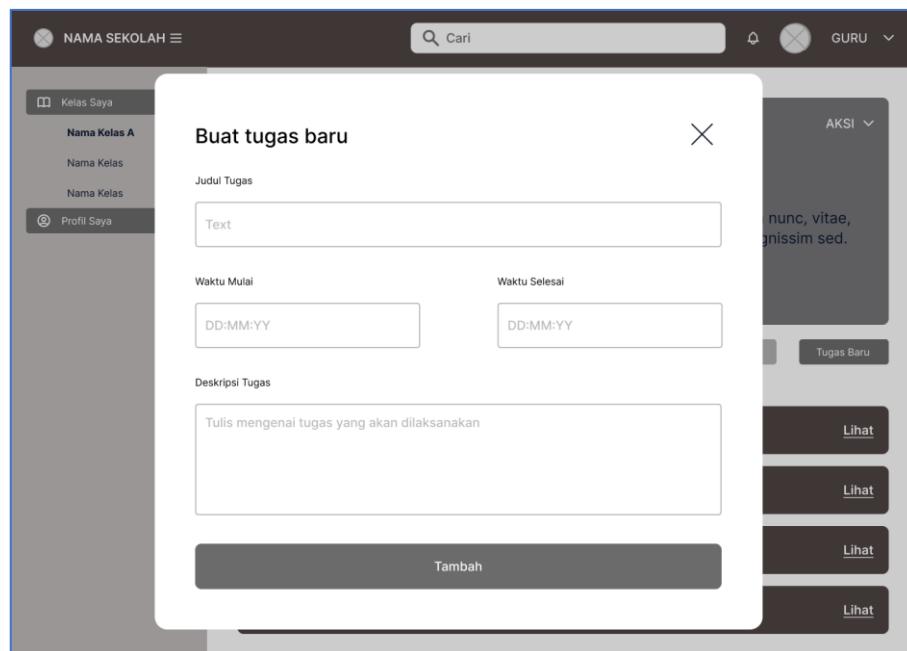
Gambar 3.31 Tampilan “Ubah Materi” untuk Guru



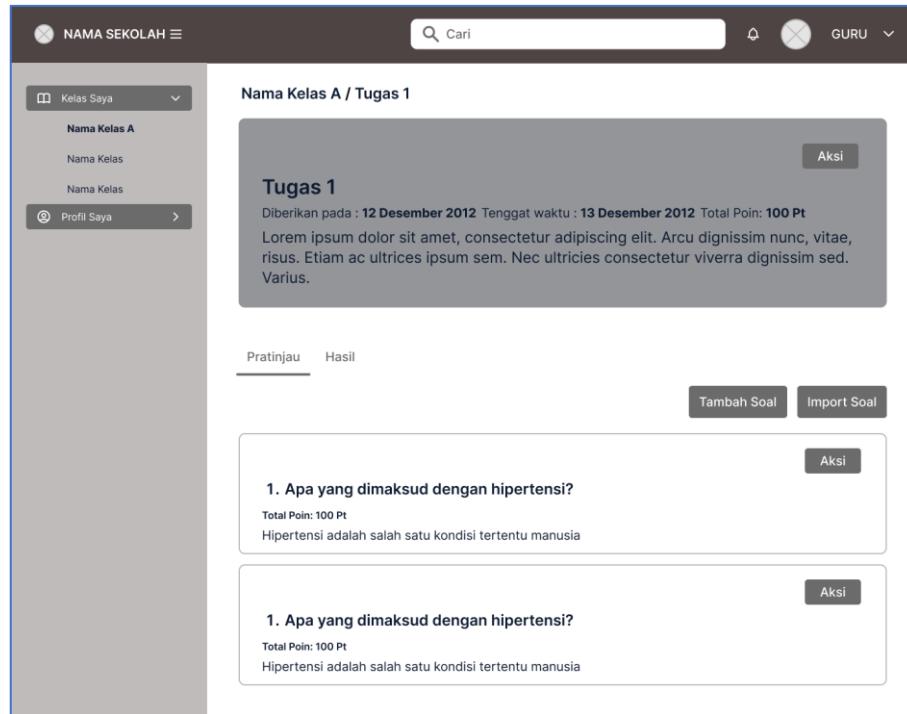
Gambar 3.32 Tampilan Konfirmasi “Penghapusan Materi” untuk Guru



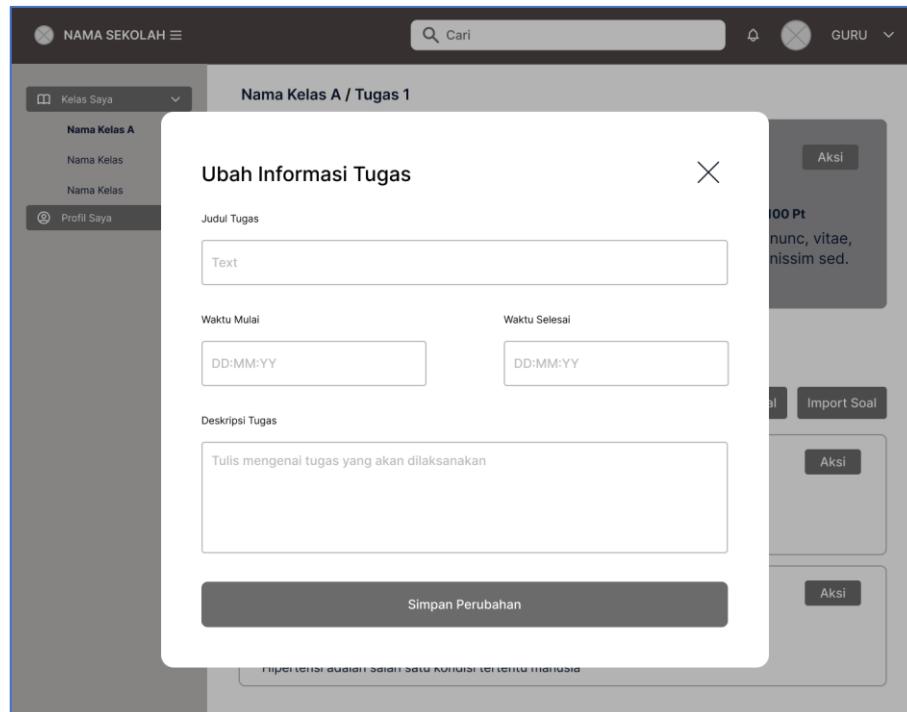
Gambar 3.33 Tampilan “Daftar Tugas” untuk Guru



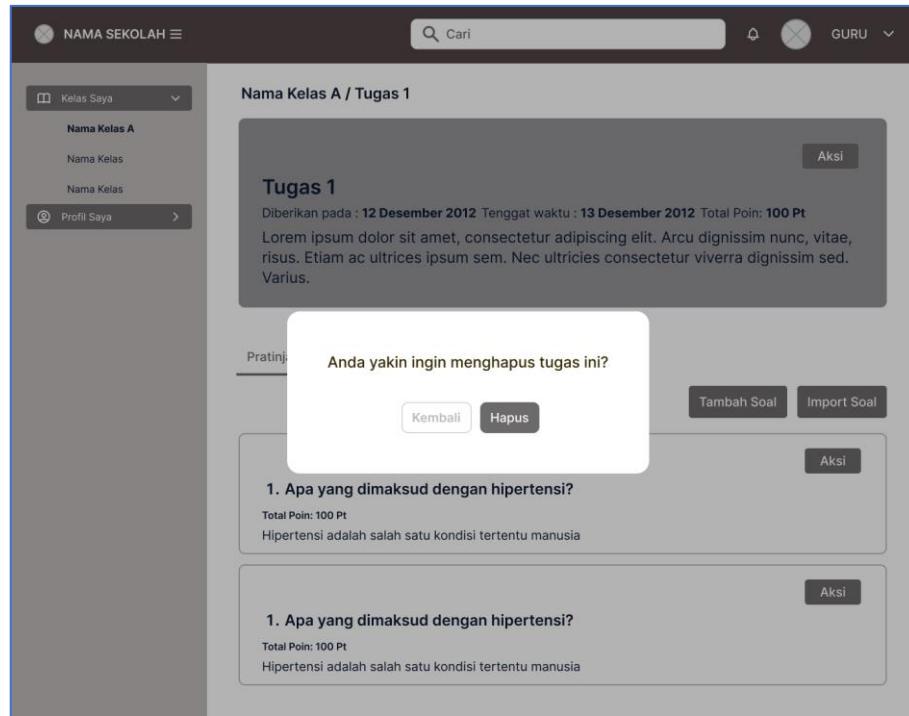
Gambar 3.34 Tampilan “Buat Tugas” Baru untuk Guru



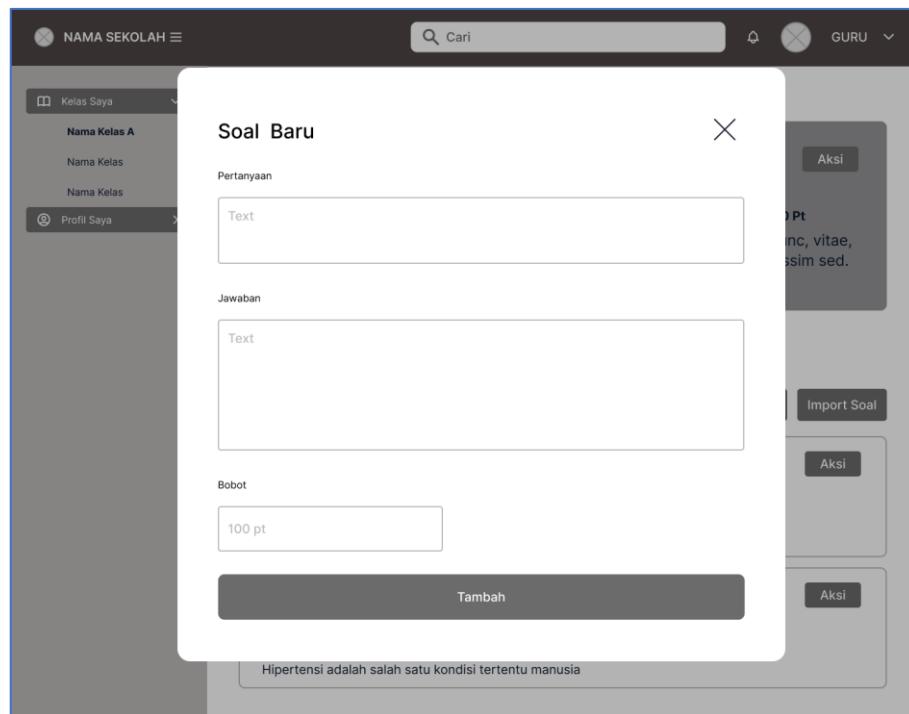
Gambar 3.35 Halaman “Lihat Tugas” untuk Guru



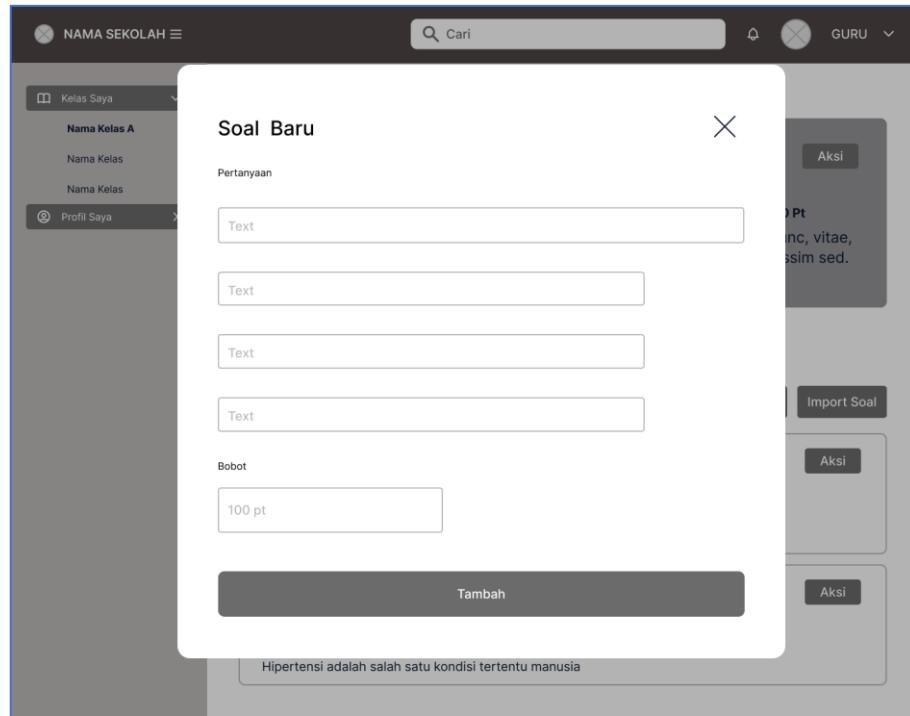
Gambar 3.36 Tampilan “Ubah Tugas” untuk Guru



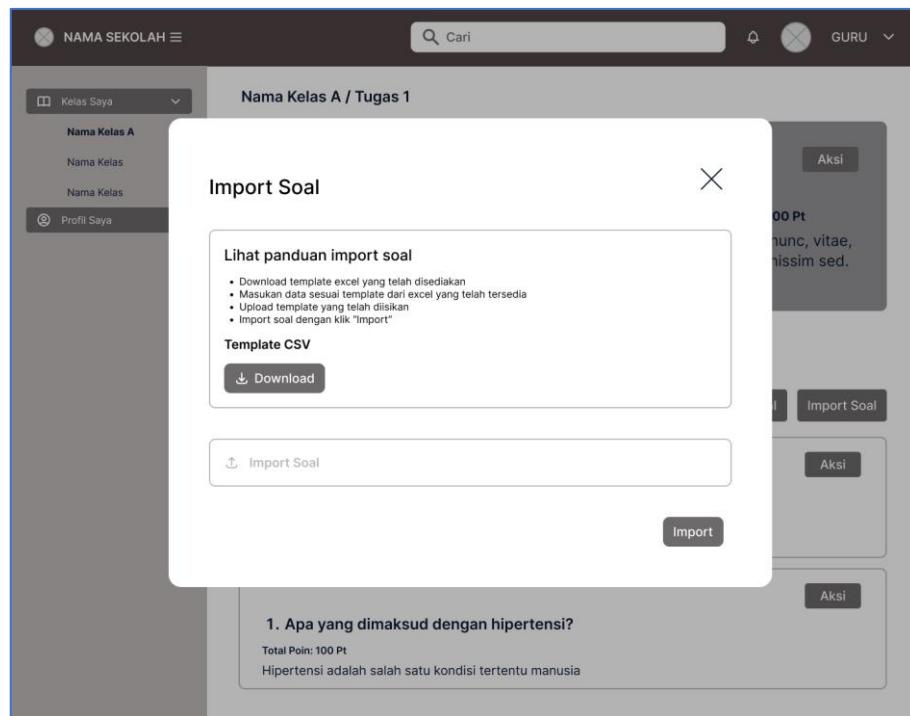
Gambar 3.37 Tampilan Konfirmasi "Penghapusan Tugas" untuk Guru



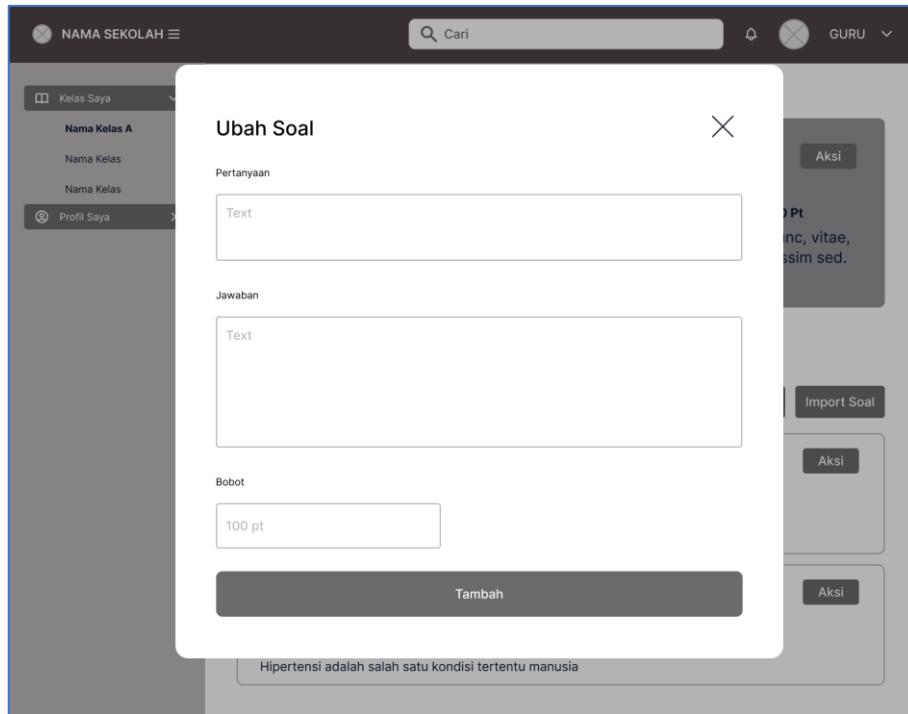
Gambar 3.38 Tampilan "Tambah Soal Essay"



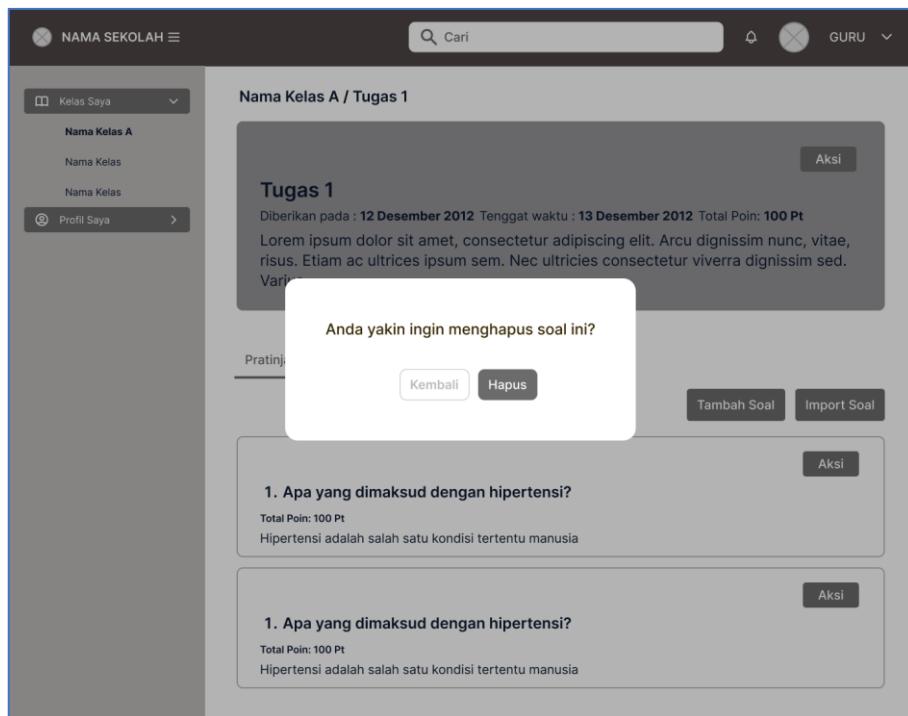
Gambar 3.39 Tampilan "Tambah Soal Pilihan Ganda"



Gambar 3.40 Tampilan “Impor Soal” pada Tugas untuk Guru



Gambar 3.41 Tampilan "Ubah Soal" pada Tugas untuk Guru



Gambar 3.42 Tampilan Konfirmasi “Penghapusan Soal”

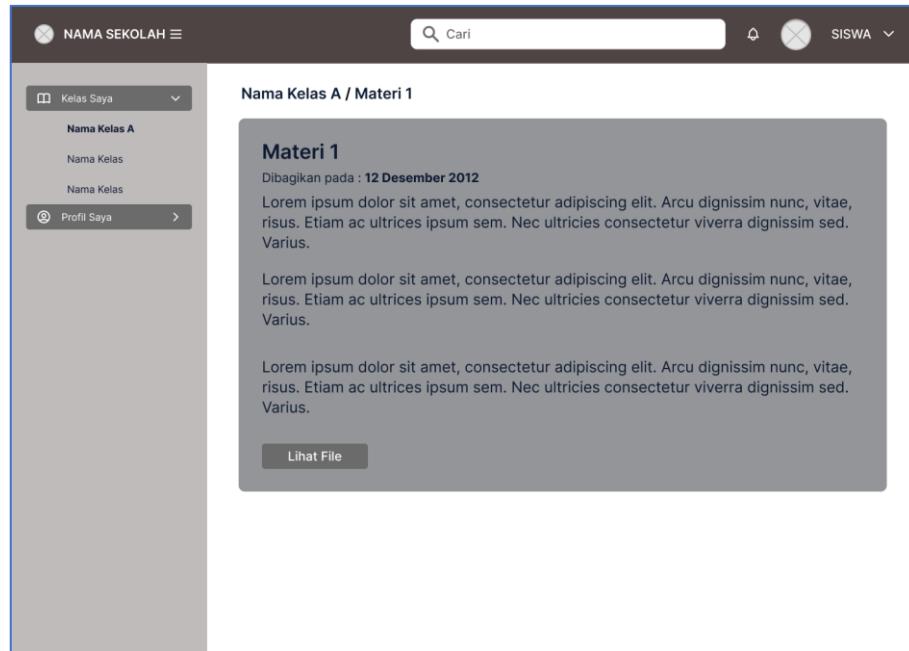
Nama Siswa	Status	Waktu	Nilai
Dibyo	-	12.00 3 Juni 2022	-
Dibyo	-	12.00 3 Juni 2022	-
Dibyo	-	12.00 3 Juni 2022	-
Dibyo	-	12.00 3 Juni 2022	-
Dibyo	-	12.00 3 Juni 2022	-
Dibyo	-	12.00 3 Juni 2022	-

Gambar 3.43 Tampilan Lihat dan Unduh Hasil Pengerajan Tugas

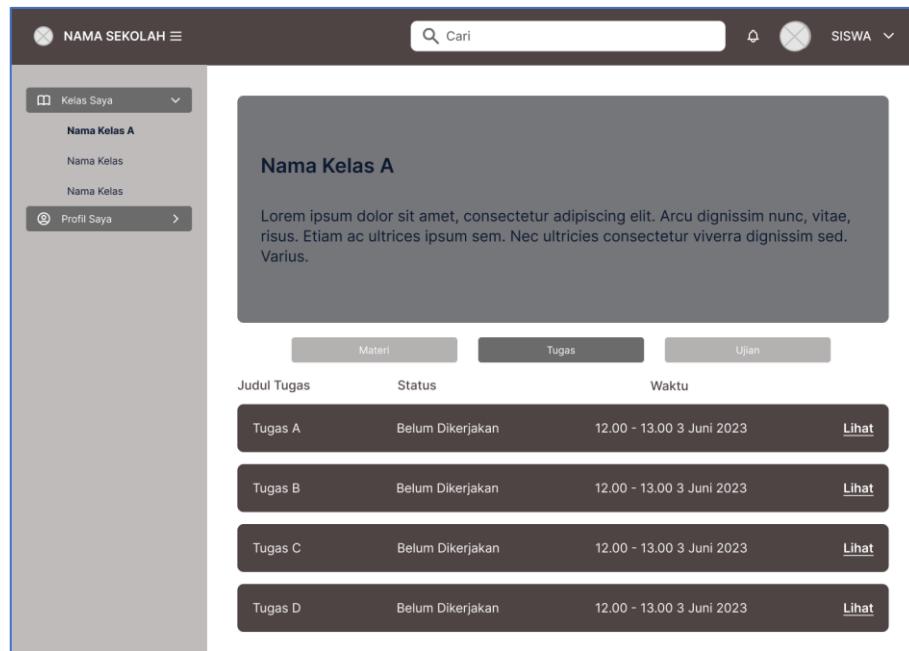
Berikutnya untuk Ujian, rancangan antarmuka yang digunakan sama seperti rancangan antarmuka Tugas dari Gambar 3.33 sampai Gambar 3.43.

Judul Materi	Waktu	Lihat
Lorem ipsum dolor sit amet, consectetur adipiscing elit.	1 Juni 2023	Lihat
Lorem ipsum dolor sit amet, consectetur adipiscing elit.	1 Juni 2023	Lihat
Lorem ipsum dolor sit amet, consectetur adipiscing elit.	1 Juni 2023	Lihat
Lorem ipsum dolor sit amet, consectetur adipiscing elit.	1 Juni 2023	Lihat

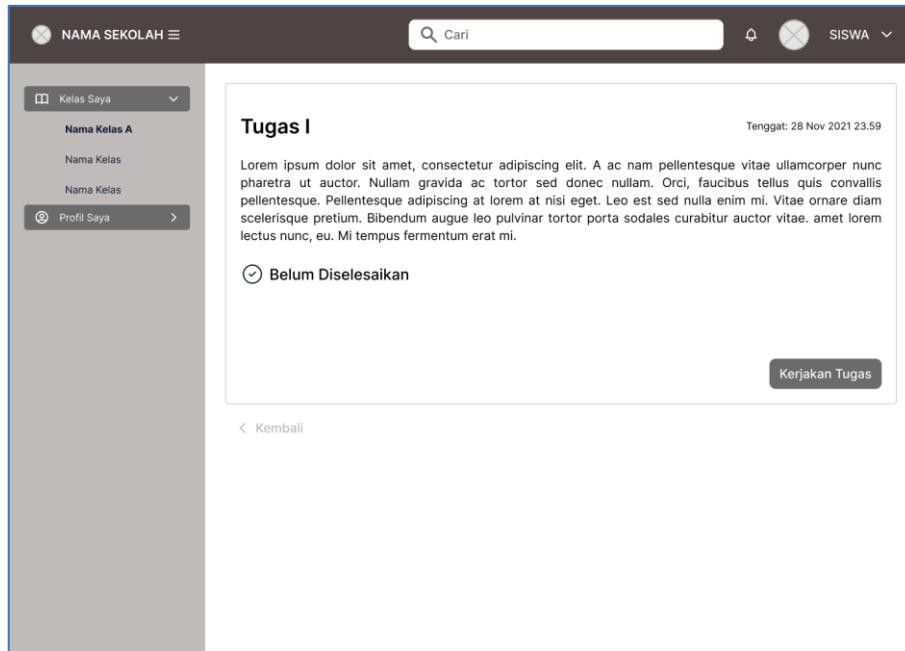
Gambar 3.44 Halaman "Daftar Materi" untuk Siswa



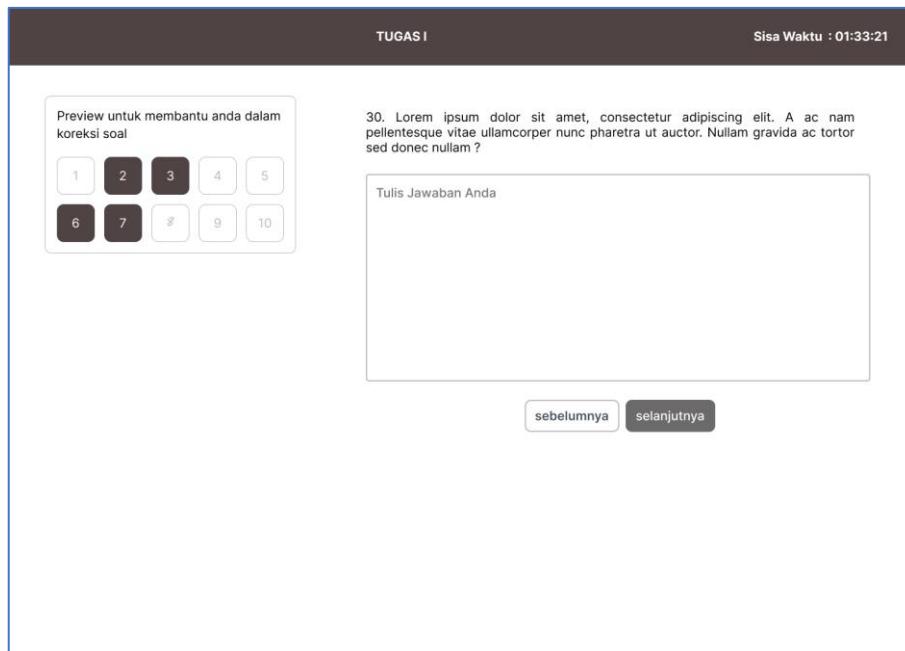
Gambar 3.45 Halaman "Lihat Materi" untuk Siswa



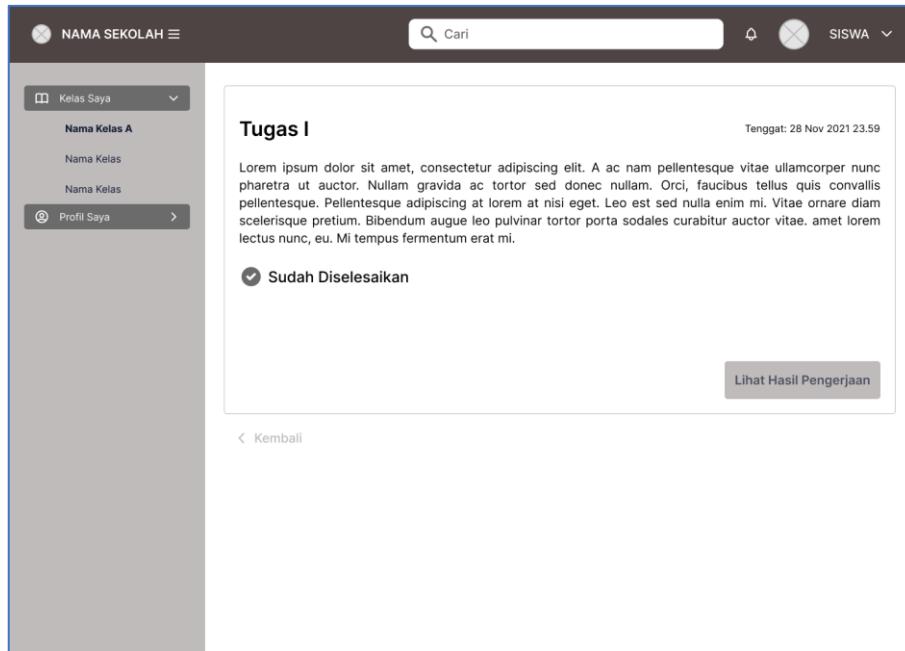
Gambar 3.46 Halaman "Daftar Tugas" untuk Siswa



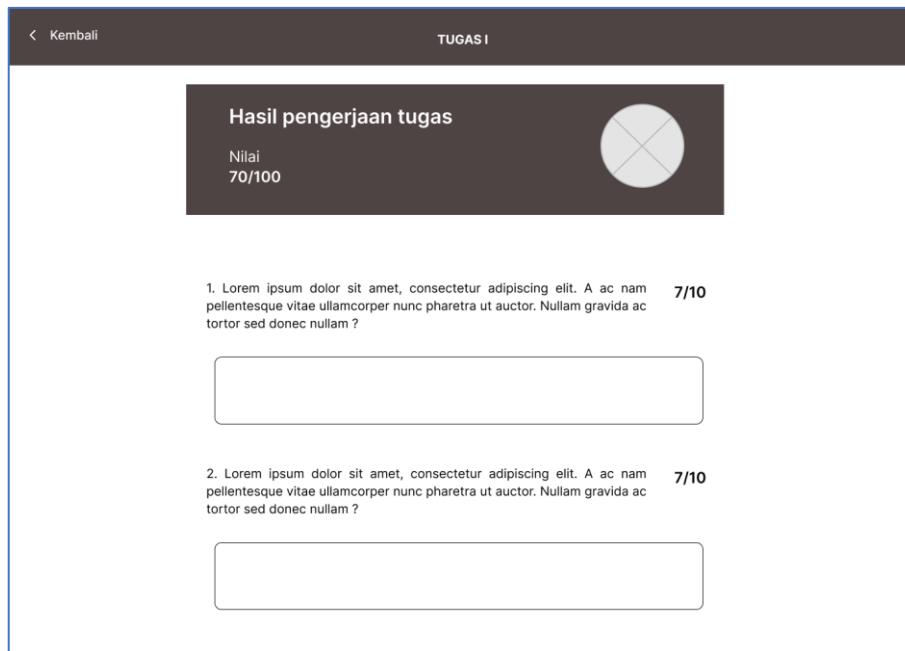
Gambar 3.47 Halaman Pendahuluan Sebelum Mengerjakan Tugas



Gambar 3.48 Halaman Pengerjaan Tugas



Gambar 3.49 Halaman Pendahuluan Setelah Mengerjakan Tugas



Gambar 3.50 Halaman Detail Hasil Pengerjaan Tugas

Berikutnya untuk Ujian pada Siswa, rancangan antarmuka yang digunakan sama seperti rancangan antarmuka Tugas dari Gambar 3.46 sampai Gambar 3.50.

3. Iterasi Ketiga

Data Akun

ID	Nama	Email	Password	Status	Action
01	Togi	Togi@gmail.com	bismillahA	Siswa	Ubah Hapus
01	Togi	Togi@gmail.com	bismillahA	Siswa	Ubah Hapus
01	Togi	Togi@gmail.com	bismillahA	Siswa	Ubah Hapus
01	Togi	Togi@gmail.com	bismillahA	Siswa	Ubah Hapus
01	Togi	Togi@gmail.com	bismillahA	Siswa	Ubah Hapus
01	Togi	Togi@gmail.com	bismillahA	Siswa	Ubah Hapus
01	Togi	Togi@gmail.com	bismillahA	Siswa	Ubah Hapus
01	Togi	Togi@gmail.com	bismillahA	Siswa	Ubah Hapus

Showing 5 of 17 entries

Gambar 3. 51 Halaman "Kelola Akun" untuk Admin

Data Kelas

ID	Mata Pelajaran	Keterangan Kelas	Kelas	Pembimbing
K01	Bahasa Indoensia	Kelas ini untuk X MIPA 3	Gilang	K01wFc
K01	Bahasa Indoensia	Kelas ini untuk X MIPA 3	Gilang	K01wFc
K01	Bahasa Indoensia	Kelas ini untuk X MIPA 3	Gilang	K01wFc
K01	Bahasa Indoensia	Kelas ini untuk X MIPA 3	Gilang	K01wFc
K01	Bahasa Indoensia	Kelas ini untuk X MIPA 3	Gilang	K01wFc
K01	Bahasa Indoensia	Kelas ini untuk X MIPA 3	Gilang	K01wFc
K01	Bahasa Indoensia	Kelas ini untuk X MIPA 3	Gilang	K01wFc
K01	Bahasa Indoensia	Kelas ini untuk X MIPA 3	Gilang	K01wFc

Showing 5 of 17 entries

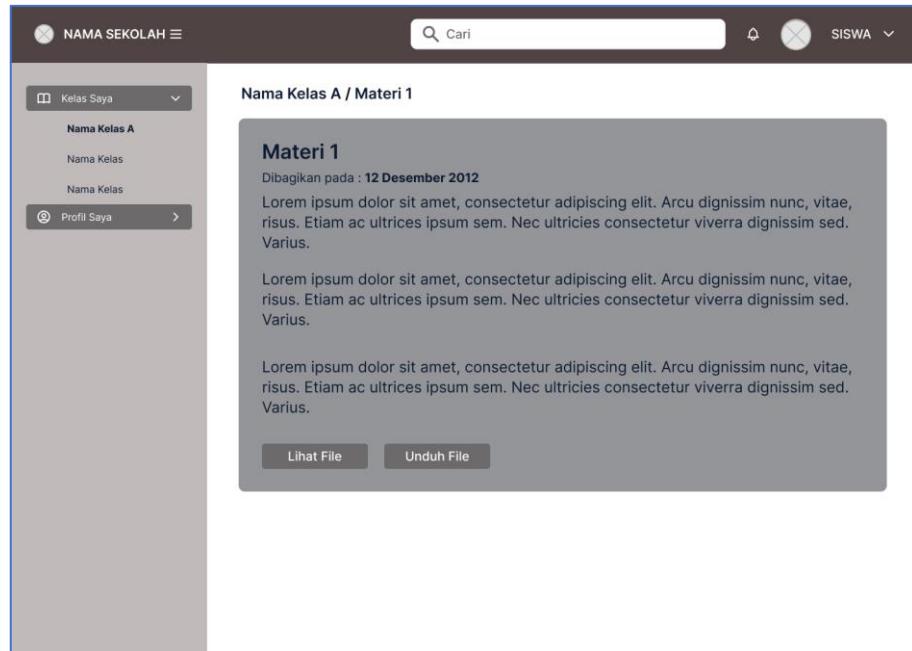
Gambar 3.52 Halaman Lihat Data Kelas untuk Admin

The screenshot shows a modal window titled "Tambah Akun". It contains four input fields: "Nama Lengkap" (Full Name), "Email", "Password", and "Level". Each field has a placeholder text ("Text") and is enclosed in a light gray border. Below the "Level" field is a dropdown menu labeled "Pilih Level". At the bottom right of the form is a dark gray button labeled "Tambah Akun".

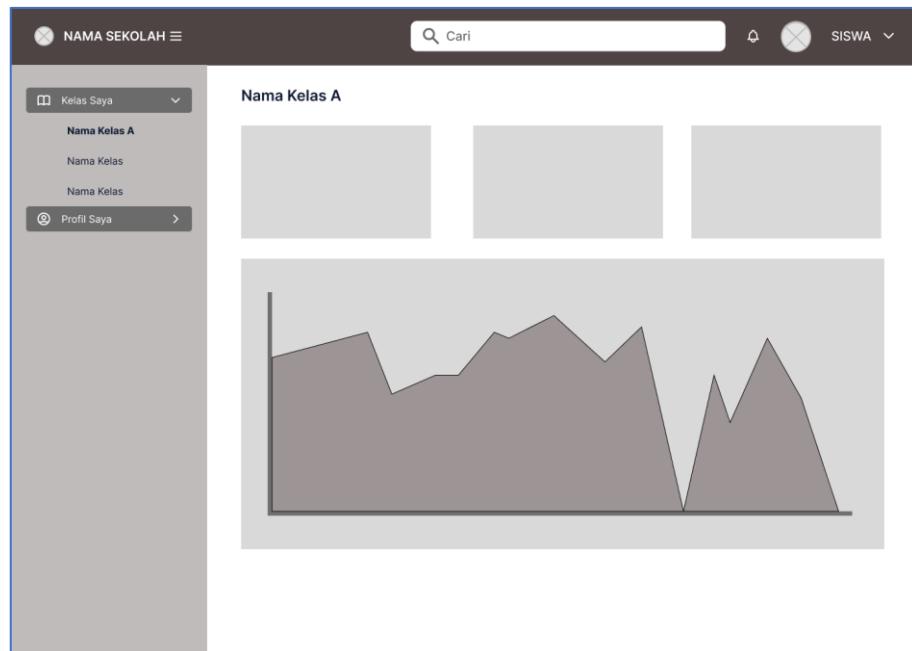
Gambar 3.53 Tampilan “Tambah Akun” untuk Admin

The screenshot shows a modal window titled "Ubah Akun". It contains four input fields: "Nama Lengkap" (Full Name), "Email", "Password", and "Level". Each field has a placeholder text ("Text") and is enclosed in a light gray border. Below the "Level" field is a dropdown menu labeled "Pilih Level". At the bottom right of the form is a dark gray button labeled "Ubah Akun".

Gambar 3.54 Tampilan Ubah Akun untuk Admin

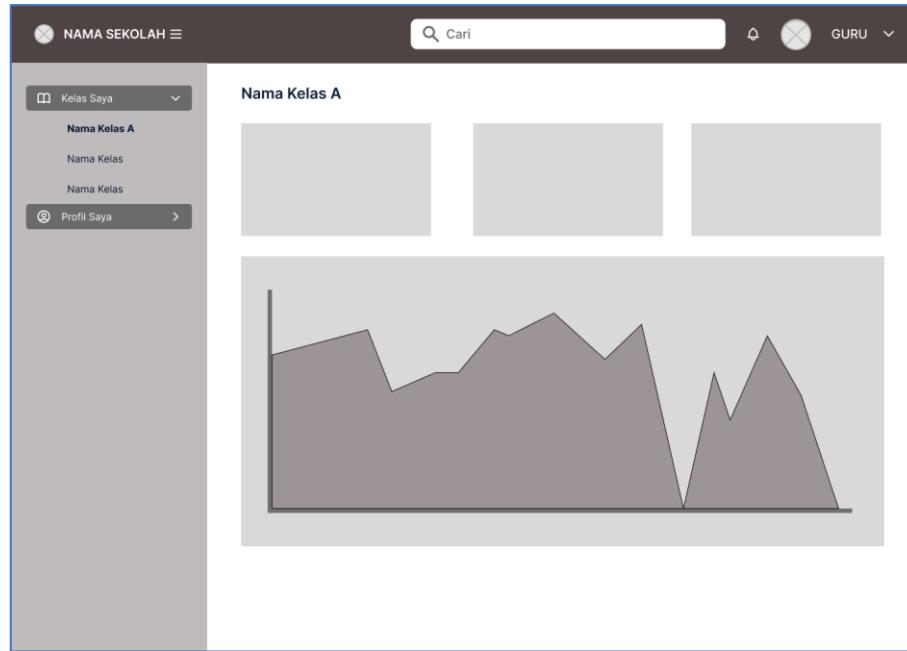


Gambar 3.55 Tampilan Untuk Melihat dan Mengunduh Materi

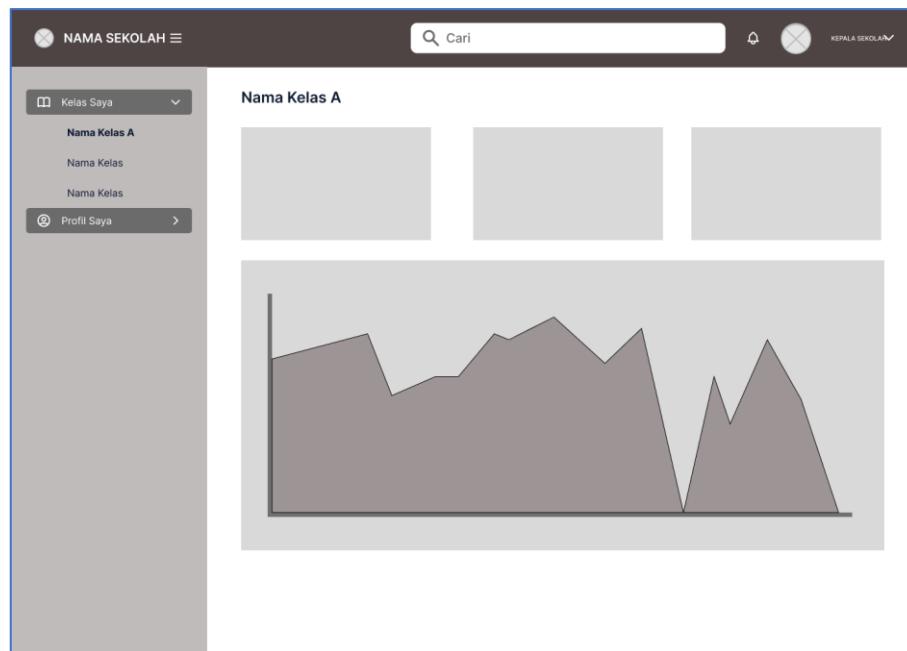


Gambar 3.56 Halaman Informasi Tugas dan Ujian untuk Siswa

4. Iterasi Keempat



Gambar 3.57 Halaman Informasi Tugas dan Ujian untuk Guru



Gambar 3.58 Halaman Informasi Tugas dan Ujian untuk Kepala Sekolah

3.4.5. Implementasi

Implementasi merupakan proses menuliskan rancangan pada tahap perancangan kedalam code program. Implementasi dilakukan dengan pendekatan *Test Driven Development* (TDD). TDD memiliki tiga tahap diantaranya *unit testing*, *code generation*, dan *refactoring* yang akan diterapkan secara berulang pada masing-masing

user stories. Tiga tahap TDD dibuat dalam bentuk implementasi kode dan dilakukan pengujian setelahnya. Bagian *refactoring* digunakan untuk optimasi kode sehingga dapat digunakan apabila diperlukan saja. Tahap implementasi ini digunakan untuk menuangkan semua hasil perancangan dari setiap iterasi ke dalam kode sehingga sistem dapat digunakan oleh client dan diimplementasikan di SMP Negeri 10 Kotabumi.

3.4.6. Pengujian Sistem

Tahapan ini merupakan pengujian hasil implementasi menggunakan metode *Black Box Testing*. Pengujian dilakukan oleh pihak SMP Negeri 10 Kotabumi didampingi oleh pengembang. Klien menguji fitur apakah sesuai dengan kebutuhan awal pada tahap analisis dan perencanaan.

1. Iterasi Pertama

Tabel 3.15 Pengujian Sistem Iterasi 1

No	Skenario Pengujian	Aktor	Hasil Diharapkan
1	Login ke dalam aplikasi	Seluruh aktor	Seluruh aktor / pengguna dapat login ke dalam aplikasi menggunakan akun dengan <i>role</i> -nya masing-masing
2	Login ke dalam aplikasi dan melakukan pengelolaan kelas	Guru	Aktor yang dapat membuat, mengubah dan menghapus kelas hanyalah Guru.
3	Login ke dalam aplikasi dan melakukan bergabung ke atau keluar dari kelas	Siswa	Aktor Siswa dapat bergabung ke suatu kelas jika memasukkan kode yang sesuai dan berikutnya dapat mengakses data yang sesuai dalam kelas tersebut.
4	Login ke dalam aplikasi dan melihat kelas	Kepala Sekolah	Aktor Kepala Sekolah tidak dapat membuat, mengubah, maupun menghapus kelas, namun bisa mengakses seluruh kelas yang tersimpan tanpa perlu memasukkan kode kelas tersebut melalui fitur Gabung ke Kelas.

2. Iterasi Kedua

Tabel 3.16 Pengujian Sistem Iterasi 2

No	Skenario Pengujian	Aktor	Hasil Diharapkan
1	Login ke dalam aplikasi dan melakukan pengelolaan materi,	Guru	Aktor yang dapat membuat, mengubah dan menghapus materi, tugas, atau ujian hanyalah Guru.

No	Skenario Pengujian	Aktor	Hasil Diharapkan
	tugas, maupun ujian		
2	Login ke dalam aplikasi dan melihat materi, tugas, maupun ujian di dalam kelas yang sudah dimasuki.	Siswa	Aktor Siswa dapat melihat seluruh materi, tugas, maupun ujian di dalam suatu kelas setelah dia bergabung ke dalam kelas tersebut.
3	Melakukan pengelolaan soal di dalam tugas atau ujian	Guru	Aktor yang dapat membuat, mengimpor, mengubah dan menghapus soal dalam tugas, atau ujian hanyalah Guru.
4	Melihat dan mengerjakan soal tugas maupun ujian di dalam kelas yang sudah dimasuki.	Siswa	Aktor Siswa dapat melihat dan mengerjakan soal tugas maupun ujian di dalam suatu kelas setelah dia bergabung ke dalam kelas tersebut, selama tugas maupun soal tersebut masih berstatus terbuka.
5	Mengunduh hasil pengerjaan tugas / ujian	Guru	Guru dapat mengunduh hasil pengerjaan tugas atau ujian yang dilakukan oleh Siswa di dalam kelas yang dikelolanya.

3. Iterasi Ketiga

Tabel 3.17 Pengujian Sistem Iterasi 3

No	Skenario Pengujian	Aktor	Hasil Diharapkan
1	Login ke dalam aplikasi dan melakukan pengelolaan akun	Guru	Aktor yang dapat membuat, mengimpor, mengubah dan menghapus akun hanyalah Admin.
2	Melihat informasi pengerjaan tugas atau ujian milik pribadi secara keseluruhan.	Siswa	Aktor Siswa dapat melihat seluruh hasil pengerjaan tugas maupun ujian di dalam suatu kelas.
3	Mengunduh materi	Siswa	Aktor Siswa dapat mengunduh materi yang dibagikan Guru di dalam kelas yang dia telah bergabung ke dalamnya.

4. Iterasi Keempat

Tabel 3.18 Pengujian Sistem Iterasi 4

No	Skenario Pengujian	Aktor	Hasil Diharapkan
1	Melihat informasi pengerjaan tugas atau ujian milik pribadi secara	Kepala Sekolah	Aktor Kepala Sekolah dapat melihat seluruh hasil pengerjaan tugas maupun ujian di seluruh kelas.

No	Skenario Pengujian	Aktor	Hasil Diharapkan
	keseluruhan.		
2	Melihat informasi penggerjaan tugas atau ujian milik pribadi secara keseluruhan.	Guru	Aktor Guru dapat melihat seluruh hasil penggerjaan tugas maupun ujian di seluruh kelas yang dikelola olehnya.

3.4.7. Retrospektif

Tahapan ini melakukan verifikasi terhadap semua *user stories* yang telah diimplementasikan dan dilakukan pengujian. Verifikasi dilakukan untuk perbandingan waktu estimasi dengan waktu realisasi sehingga dapat diketahui kendala-kendala penyebab *over* atau *under* estimasi pada pelaksanaan penelitian. Verifikasi ini bertujuan untuk mencegah perbedaan waktu estimasi pada penelitian selanjutnya.

3.5. Evaluasi Akhir Aplikasi

Untuk evaluasi akhir aplikasi yang dikembangkan akan digunakan metode *System Usability Scale* (SUS). *System Usability Scale* (SUS) merupakan pengujian yang dilakukan untuk menilai suatu sistem untuk mengukur tingkat *usability* dari aplikasi yang dibangun. SUS ini berisi 10 peryataan yang akan diberikan kepada calon pengguna aplikasi setelah aplikasi selesai dibangun. Pernyataan SUS terdapat pernyataan dengan kalimat positif pada nomor ganjil dan pernyataan negatif pada nomor genap [32]. Jumlah responden yang digunakan untuk pengujian usability minimal 10 responden sudah cukup akurat untuk mendapatkan kualitas penelitian [33]. Pengujian SUS dalam penelitian ini akan diberikan kepada 10 responden yang terdiri dari pihak SMPN 10 Kotabumi, dan rekan-rekan mahasiswa ITERA.

BAB IV

HASIL PENELITIAN DAN PEMBAHASAN

4.1. Lingkungan Pengembangan Aplikasi

Pada bagian ini akan dijabarkan proses iterasi yang dilakukan pada sub bab 3.4.3.

4.2. Personal eXtreme Programming

Proses pengembangan sistem dilakukan sesuai dengan tahapan yang ada pada *System Development Life Cycle Personal eXtreme Programming* yaitu dari tahapan Analisis Kebutuhan sampai semua fase iterasi yang memenuhi kebutuhan user.

4.2.1 Analisis Kebutuhan

Tahap analisis kebutuhan ini diperoleh dari *client* melalui wawancara dan diskusi bersama ibu Eny Ros selaku Wakil Kepala Bidang Kurikulum di SMP Negeri 10 Kotabumi, dan menyesuaikan kebutuhan aplikasi dengan model AES yang telah dikembangkan oleh PURINO Kecerdasan Buatan ITERA. Hasil yang didapatkan oleh pengembang dituliskan dalam bentuk *user stories* dengan format “Sebagai <jenis pengguna>, saya ingin <melakukan tindakan tertentu>, sehingga <mendapatkan manfaat dari tindakan tersebut>”, seperti pada tabel 3.2. *User stories* ini telah berhasil menjadi sarana komunikasi yang mudah dipahami oleh user dan pengembang. Hal ini didasari fakta bahwa seluruh *user stories* sudah dapat diselesaikan, seperti yang akan dijelaskan pada beberapa sub bab berikutnya.

4.2.2 Perencanaan

Fase perencanaan memiliki tiga tahapan yaitu memperkirakan waktu pengerjaan *user story*, menentukan prioritas *user story*, serta penentuan *velocity*. Tahapan memperkirakan waktu pengerjaan *user story* seperti pada tabel 3.3 membutuhkan pengalaman pengembang dalam pengembangan sebelumnya. Tahapan menentukan prioritas *user story* dilakukan menggunakan aturan MoSCoW, memberikan kemudahan pengembang dalam klasifikasi prioritas *user story*.

User story yang terbagi menjadi dua klasifikasi (*Must Have* dan *Should Have*) yang dijabarkan pada tabel 3.4 memudahkan user dalam memahami bobot setiap *user story* yang ada, hal ini dikarenakan klasifikasi dibagi menjadi dua bagian sederhana namun terlihat jelas perbedaan klasifikasinya. Pada tahapan penentuan *velocity*, pengembang

menentukan nilai maksimal sebesar 5 berdasarkan pengalaman dari pengembang dalam pengembangan aplikasi berbasis web.

4.2.3 Inisialisasi Iterasi

Pemodelan sistem menggunakan UML berdasarkan *user story* mempermudah komunikasi antara user dan pengembang sebelum membangun sistem yang sesuai dengan kebutuhan user.

4.2.4 Perancangan

Pemodelan antarmuka pengguna berdasarkan UML pada tahap inisialisasi iterasi dalam bentuk *low-fidelity prototype* berhasil dikembangkan. *Low-fidelity prototype* yang telah dibuat dijadikan sebagai acuan dalam pengembangan antarmuka pengguna sehingga user antarmuka pengguna yang dibuat pada masa pengembangan tidak bertambah secara signifikan. Hasil user interface yang dikembangkan akan dijabarkan pada setiap iterasi di beberapa sub bab berikutnya.

4.2.5 Iterasi Pertama

Pada tahap ini terdapat dua *user stories* yang akan dikerjakan. Kedua user story tersebut adalah *Story-02* dan *Story-04*. Total *Story Point* pada iterasi ini adalah sebesar 5, dan sesuai dengan nilai *velocity* dari pengembang yang sebesar 5. Maka seluruh *user story* pada iterasi diharapkan dapat selesai dalam kurun waktu 10 hari kerja.

4.2.5.1. Implementasi

User stories yang akan dikerjakan pada iterasi menyangkut kebutuhan autentikasi setiap user terhadap aplikasi yang dikembangkan serta kebutuhan Guru dalam pengelolaan kelas di aplikasi sebagai titik awal dimulainya siklus belajar mengajar, termasuk pemberian tugas dan ujian.

Pada awal tahap implementasi di iterasi pertama ini, pengembang terlebih dahulu membuatkan tabel basis data sesuai dengan *Class Diagram* iterasi pertama seperti yang ditampilkan pada Gambar 3.5. Karena aplikasi berbasis web ini dikembangkan dengan menggunakan bahasa pemrograman PHP dengan bantuan *framework* pengembangan Laravel, proses pembuatan basis data dilakukan dengan fitur *migration* pada *framework* tersebut. Pembuatan basis data ini bersamaan dengan pembuatan *model*, yang merupakan bagian dari konsep *Model, View, Controller* (MVC) pada

pengembangan aplikasi berbasis web. Pada iterasi ini dihasilkan dua *model* dan dua *migration*, yang masing-masing mewakili entitas *user* dan entitas *classroom*, dan satu *migration* tambahan yang menjadi *pivot table* *classroom_member*, yang digunakan untuk menyederhanakan hubungan *many-to-many* dari dua entitas tersebut. Berikut ini akan dijelaskan potongan kode program dari *model* dan *migration* yang telah dibuat :

```
1 Schema::create('users', function (Blueprint $table) {
2     $table->id();
3     $table->string('name');
4     $table->string('email')->unique();
5     $table->string('password');
6     $table->integer('role');
7     $table->timestamp('email_verified_at')->nullable();
8     $table->rememberToken();
9     $table->timestamps();
10});
```

Gambar 4.1 *Migration* Tabel *users*

Pada potongan kode di atas dapat dilihat pada baris 1 merupakan perintah untuk membuat tabel *users*, dengan atribut yang ditampilkan seperti pada baris 2 hingga baris 9. Dari baris 2 hingga baris 6 dapat dilihat 5 atribut utama dari entitas *user* seperti yang telah disebutkan pada Tabel 3.6, sementara untuk atribut lainnya pada baris 7 hingga 9, merupakan atribut yang secara otomatis ditambahkan oleh Laravel.

```
1 Schema::create('classrooms', function (Blueprint $table) {
2     $table->id();
3     $table->foreignId('teacher_id')
4         ->references('id')->on('users')
5         ->constrained()->cascadeOnDelete();
6     $table->string('name');
7     $table->text('description');
8     $table->string('enrollment_key')->unique();
9     $table->timestamps();
10});
```

Gambar 4.2 *Migration* Tabel *classrooms*

Berikutnya, pada Gambar 4.2 ditampilkan kode program yang merupakan *migration* berisi perintah untuk membuat tabel *classrooms* pada baris 1, dengan atribut seperti yang ditampilkan pada baris 2 hingga baris 9. Dari baris 2 hingga baris 8 dapat dilihat 5 atribut utama dari entitas *user* seperti yang telah disebutkan pada Tabel 3.7. Perlu diperhatikan pada baris 3 sampai 4, baris kode tersebut dijalankan untuk

membuat kolom bertipe *foreign key* yang bernama ***teacher_id*** yang merujuk pada kolom ***id*** di tabel ***users***.

```

1 Schema::create('classroom_member', function (Blueprint $table) {
2     $table->id();
3     $table->foreignIdFor(Classroom::class)
4         ->constrained()->cascadeOnDelete();
5     $table->foreignId('student_id')
6         ->references('id')->on('users')
7         ->constrained()->cascadeOnDelete();
8     $table->timestamps();
9 });

```

Gambar 4.3 *Migration* Tabel Pivot *classroom_member*

Berikutnya pada Gambar 4.3 ditampilkan potongan kode program *migration* untuk membuat *pivot table* ***classroom_member*** untuk menyederhanakan hubungan *many-to-many* entitas *user* dengan entitas *classroom*. Pada implementasinya, *pivot table* ini dibuat untuk menyesuaikan aplikasi dengan kebutuhan bahwa seorang siswa bisa berada di dalam banyak kelas sekaligus, dan di saat yang bersamaan sebuah kelas bisa memiliki banyak siswa sekaligus. Pada potongan gambar tersebut, dapat dilihat juga pada baris 2 hingga 7, dibuat 3 atribut yang sesuai dengan yang ditampilkan pada Tabel 3.8. Perlu diperhatikan pada baris 3, *foreign key* langsung merujuk pada *model* Classroom, dimana Laravel akan secara otomatis memberikan nama *classroom_id* pada atribut tersebut, yang mana atribut tersebut akan terhubung ke kolom ***id*** di tabel ***classrooms***. Sementara pada baris 5, pengembang memberikan nama *student_id* yang merujuk apda kolom ***id*** di tabel ***users***.

```

1 protected $fillable = [
2     'name',
3     'email',
4     'password',
5     'role',
6 ];
7
8 protected $hidden = [
9     'password',
10    'remember_token',
11];
12
13 public function classrooms(): HasMany
14 {
15     return $this->hasMany(Classroom::class);
16 }
17
18 public function enrolled_classrooms(): BelongsToMany
19 {
20     return $this->belongsToMany(
21         Classroom::class,
22         'classroom_member',
23         'student_id',
24         'classroom_id',
25         'id',
26         'id'
27     );
28 }

```

Gambar 4.4 Model User

Berikutnya pada Gambar 4.4 ditampilkan potongan kode program yang menampilkan *model users*. Pada baris 2 hingga 5 dapat dilihat atribut yang dapat diisi dan diubah secara manual. Kemudian pada baris 13 hingga 16, merupakan fungsi *classrooms*, yang dibuat untuk mendefinisikan hubungan antara tabel *users* dan tabel *classrooms* yaitu, *one-to-many*, dimana seorang *user* bisa memiliki banyak kelas sekaligus. Selain itu, fungsi tersebut juga dibuat untuk memudahkan pengambilan data kelas yang berhubungan dengan satu *user*. Pada implementasinya, fungsi ini dibuat untuk menyesuaikan dengan kebutuhan bahwa seorang guru dapat mengajar di banyak kelas, namun satu kelas hanya dapat diajar oleh satu guru.

Selanjutnya pada baris 18 hingga 28 dapat dilihat pendefinisian fungsi *enrolled_classrooms*, yang menunjukkan hubungan *many-to-many* dengan tabel *classrooms*, namun melalui *pivot table classroom_member*, fungsi ini juga dibuat untuk memudahkan pengambilan data seluruh kelas yang di dalamnya seorang siswa telah terdaftar.

```

1 protected $fillable = [
2     'teacher_id',
3     'name',
4     'description',
5     'enrollment_key',
6 ];
7
8 public function teacher(): BelongsTo
9 {
10    return $this->belongsTo(User::class, 'teacher_id', 'id');
11 }
12
13 public function students(): BelongsToMany
14 {
15    return $this->belongsToMany(
16        User::class,
17        'classroom_member',
18        'classroom_id',
19        'student_id',
20        'id',
21        'id'
22    );
23 }

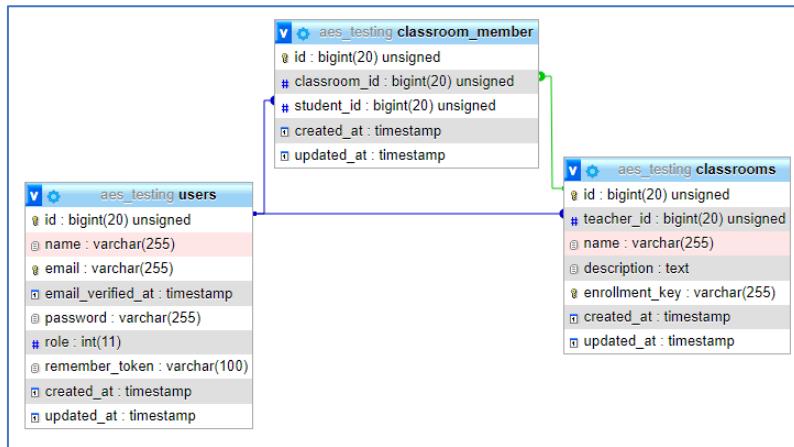
```

Gambar 4.5 *Model Classroom*

Berikutnya pada Gambar 4.5 ditampilkan potongan kode program yang menampilkan *model classroom*. Pada baris 2 hingga 5 dapat dilihat atribut yang dapat diisi dan diubah secara manual. Kemudian pada baris 8 hingga 11, merupakan fungsi *teacher*, yang dibuat untuk mendefinisikan hubungan antara tabel *classrooms* dan tabel *users*, dimana sebuah kelas dimiliki (dikelola/diajar) oleh seorang *user* (Guru). Selain itu, fungsi tersebut juga dibuat untuk memudahkan pengambilan data Guru yang mengelola suatu kelas.

Selanjutnya pada baris 13 hingga 23 dapat dilihat pendefinisan fungsi *students*, yang menunjukkan hubungan *many-to-many* dengan tabel *users*, namun melalui *pivot table* *classroom_member*, fungsi ini juga dibuat untuk memudahkan pengambilan data seluruh siswa telah terdaftar di dalam suatu kelas.

Menggunakan mode desainer pada aplikasi manajemen basis data *phpMyAdmin*, berikut ditampilkan visualisasi basis data yang telah dibuat pada iterasi pertama :



Gambar 4.6 Visualiasi Basis Data Iterasi Pertama

Selanjutnya, perlu diketahui juga, untuk memudahkan pengujian, pengembang menggunakan fitur *seeder* pada Laravel untuk membuatkan sejumlah data *dummy* ke dalam basis data. Adapun data yang telah dibuatkan sebagai berikut :

	id	name	email	password	role
1	Kepala Sekolah	kepala@sekolah	\$2y\$10\$92IXUNpkjO0rOQ5byMi.Ye4oKoEa3Ro9IIC/.og/at2...	0	
2	Guru A	guruA@sekolah	\$2y\$10\$92IXUNpkjO0rOQ5byMi.Ye4oKoEa3Ro9IIC/.og/at2...	1	
3	Guru B	guruB@sekolah	\$2y\$10\$92IXUNpkjO0rOQ5byMi.Ye4oKoEa3Ro9IIC/.og/at2...	1	
4	Guru C	guruC@sekolah	\$2y\$10\$92IXUNpkjO0rOQ5byMi.Ye4oKoEa3Ro9IIC/.og/at2...	1	
5	Siswa 1	siswa1@sekolah	\$2y\$10\$92IXUNpkjO0rOQ5byMi.Ye4oKoEa3Ro9IIC/.og/at2...	2	
6	Siswa 2	siswa2@sekolah	\$2y\$10\$92IXUNpkjO0rOQ5byMi.Ye4oKoEa3Ro9IIC/.og/at2...	2	
7	Siswa 3	siswa3@sekolah	\$2y\$10\$92IXUNpkjO0rOQ5byMi.Ye4oKoEa3Ro9IIC/.og/at2...	2	
8	Siswa 4	siswa4@sekolah	\$2y\$10\$92IXUNpkjO0rOQ5byMi.Ye4oKoEa3Ro9IIC/.og/at2...	2	
9	Siswa 5	siswa5@sekolah	\$2y\$10\$92IXUNpkjO0rOQ5byMi.Ye4oKoEa3Ro9IIC/.og/at2...	2	
10	Siswa 6	siswa6@sekolah	\$2y\$10\$92IXUNpkjO0rOQ5byMi.Ye4oKoEa3Ro9IIC/.og/at2...	2	

Gambar 4.7 Data *Dummy* Tabel *users*

Yang perlu diperhatikan pada Gambar 4.7 di atas adalah bahwa seluruh atribut yang ditampilkan sesuai dengan yang dijelaskan pada tahap inisialisasi iterasi pertama, dan pada atribut **role** digunakan kode 0 untuk Kepala Sekolah, 1 untuk Guru, dan 2 untuk Siswa.

Kemudian, pada tabel **classrooms** dan **classroom_member** juga terlebih dahulu dibuat data *dummy* seperti yang ditampilkan pada Gambar 4.8 dan Gambar 4.9.

id	teacher_id	name	description	enrollment_key
1	2	VII - A	Kelas VII - A	A2345678
2	2	VIII - A	Kelas VIII - A	B2345678
3	2	IX - A	Kelas IX - A	C2345678
4	3	VII - B	Kelas VII - B	D2345678
5	3	VIII - B	Kelas VIII - B	E2345678
6	3	IX - B	Kelas IX - B	F2345678
7	4	VII - C	Kelas VII - C	G2345678
8	4	VIII - C	Kelas VIII - C	H2345678
9	4	IX - C	Kelas IX - C	I2345678

Gambar 4.8 Data *Dummy* Tabel *classrooms*

id	classroom_id	student_id
1	1	5
2	4	5
3	7	5
4	1	6
5	4	6
6	7	6
7	2	7
8	5	7
9	8	7
10	2	8
11	5	8
12	8	8
13	3	9
14	6	9
15	9	9
16	3	10
17	6	10
18	9	10

Gambar 4.9 Data *Dummy* Tabel Pivot *classroom_member*

A. Unit Testing

Pengecekan fungsi yang sedang dikembangkan diperlukan sebagai dasar acuan sistem yang akan dibangun seperti yang dijelaskan pada Tabel 4.1

Tabel 4.1 *Unit Testing* Iterasi Pertama

No.	<i>Unit Test</i>	Skenario Pengujian	Aktor	Hasil yang diharapkan	Status
1	Login Aplikasi	Login dengan memasukkan email dan password yang terdaftar	Seluruh Aktor	Sistem menerima akses login	Berhasil
2	Login Aplikasi	Login dengan memasukkan email yang tidak terdaftar	Seluruh Aktor	Aplikasi menolak akses login	Berhasil
3	Login Aplikasi	Login dengan memasukkan password yang salah	Seluruh Aktor	Aplikasi menolak akses login	Berhasil
4	Membuat Kelas	Aktor guru membuat kelas baru	Guru	Kelas berhasil dibuat	Berhasil
5	Membuat Kelas	Aktor Guru membuat kelas baru, dan aplikasi akan membuatkan kode <i>enrollment</i> kelas yang unik	Guru	Aplikasi membuatkan kode yang unik (belum digunakan di basis data)	Berhasil
6	Membuat Kelas	Aktor yang bukan guru membuat kelas	Aktor selain guru	Kelas tidak dibuat	Berhasil
7	Bergabung ke Kelas	Siswa bergabung ke kelas baru	Siswa	Siswa berhasil bergabung ke kelas	Berhasil
8	Bergabung ke Kelas	Siswa bergabung ke kelas yang sama lebih dari sekali	Siswa	Siswa hanya akan bergabung di kelas yang sama satu kali	Berhasil

Unit Test pada iterasi pertama seperti yang ditampilkan pada Tabel ??? dijelaskan sebagai berikut melalui beberapa potongan kode program :

```

1 public function test_login_with_valid_registered_email_and_password(): void
2 {
3     $user = User::factory()->create();
4
5     $response = $this->post('/login', [
6         'email' => $user->email,
7         'password' => 'password',
8     ]);
9
10    $this->assertAuthenticated();
11    $response->assertRedirect(RouteServiceProvider::HOME);
12 }

```

Gambar 4.10 Kode *Unit Testing* Iterasi Pertama (1)

Potongan kode program di atas menampilkan pengujian unit logika ketika pengguna login ke dalam aplikasi menggunakan email dan password dari akun yang telah terdaftar. Pada baris 3, dibuatkan dahulu *user* baru menggunakan *factory*, kemudian pada baris 5 sampai 8, dilakukan *request* bertipe *POST* ke url '/login' dan mengirimkan kredensial berupa email dari *user* yang baru saja dibuat dan password berisi kata 'password', (ini berlaku sebagai hasil dari pengaturan di bagian *factory* sebelumnya). Setelah itu, pengujian menuntut nilai berupa status pengguna apakah telah terautentikasi melalui kode pada baris 10, dan menuntut nilai apakah *route* dialihkan ke halaman awal pada baris 11. Jika kedua tuntutan tersebut terpenuhi, maka pengujian akan dinyatakan lolos, dan menampilkan luaran seperti berikut ini :

0.32s

Gambar 4.11 Luaran *Unit Testing* Iterasi Pertama (1)

```

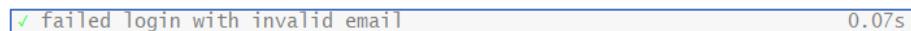
1 public function test_failed_login_with_invalid_email(): void
2 {
3     $user = User::factory()->create();
4
5     $response = $this->post('/login', [
6         'email' => 'wrong-email',
7         'password' => 'password',
8     ]);
9
10    $this->assertGuest();
11 }

```

Gambar 4.12 Kode *Unit Testing* Iterasi Pertama (2)

Potongan kode program di atas menampilkan *unit test* ketika pengguna login ke dalam aplikasi menggunakan email yang tidak terdaftar. Pada baris 3, dibuatkan dahulu *user* baru menggunakan *factory*, kemudian pada baris 5 sampai 8, dilakukan *request* bertipe *POST* ke url '/login' dan mengirimkan kredensial

berupa email sembarang dan password berisi kata ‘password’ Setelah itu, pengujian menuntut nilai berupa status pengguna apakah tetap tidak terautentikasi (sebagai tamu / *guest*) melalui kode pada baris 10. Jika tuntutan tersebut terpenuhi, maka pengujian akan dinyatakan lolos, dan menampilkan luaran seperti berikut ini :



✓ failed login with invalid email 0.07s

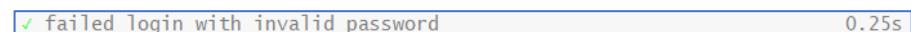
Gambar 4.13 Luaran *Unit Testing* Iterasi Pertama (2)

```

1 public function test_failed_login_with_invalid_password(): void
2 {
3     $user = User::factory()->create();
4
5     $response = $this->post('/login', [
6         'email'    => $user->email,
7         'password' => 'wrong-password',
8     ]);
9
10    $this->assertGuest();
11 }
```

Gambar 4.14 Kode *Unit Testing* Iterasi Pertama (3)

Potongan kode program di atas menampilkan *unit test* ketika pengguna login ke dalam aplikasi menggunakan email yang terdaftar, namun kata sandi yang salah. Pada baris 3, dibuatkan dahulu *user* baru menggunakan *factory*, kemudian pada baris 5 sampai 8, dilakukan *request* bertipe *POST* ke url ‘/login’ dan mengirimkan kredensial berupa email dari *user* yang baru saja dibuat dan password yang salah. Setelah itu, pengujian menuntut nilai berupa status pengguna apakah tetap tidak terautentikasi (sebagai tamu / *guest*) melalui kode pada baris 10. Jika tuntutan tersebut terpenuhi, maka pengujian akan dinyatakan lolos, dan menampilkan luaran seperti berikut ini :



✓ failed login with invalid password 0.25s

Gambar 4.15 Luaran *Unit Testing* Iterasi Pertama (3)

```

1 public function test_teacher_creating_classroom(): void
2 {
3     $user = User::factory()->create([
4         'role'      => 1
5     ]);
6
7     if($user->role == 1) {
8         $classroom = Classroom::factory()->create([
9             'teacher_id'    => $user->id,
10            'name'          => "New Classroom for Testing",
11        ]);
12    }
13
14    $this->assertDatabaseHas('classrooms', [
15        'name' => 'New Classroom for Testing',
16    ]);
17 }

```

Gambar 4.16 Kode *Unit Testing* Iterasi Pertama (4)

Potongan kode program di atas menampilkan *unit test* ketika aktor Guru hendak membuat kelas. Pada baris 3, dibuatkan dahulu *user* baru menggunakan *factory*, seperti yang telah disebutkan sebelumnya, role dengan kode 1 merupakan kode *role* untuk Guru, kemudian pada baris 7 dilakukan pengecekan apakah *role* dari *user* yang telah dibuat adalah Guru (bernilai 1), jika ya, maka akan dibuatkan kelas baru menggunakan *factory* dari model *classroom* dan diberikan nilai *teacher_id* dari nilai *id* dari *user* yang baru saja dibuat, dan diberi nama kelas “New Classroom for Testing”. Setelah itu, pengujian menuntut nilai *true* apakah di table *classrooms* di basis data terdapat baris yang memiliki nilai atribut *name* “New Classroom for Testing” melalui kode pada baris 14 sampai 16. Jika tuntutan tersebut terpenuhi, maka pengujian akan dinyatakan lolos, dan menampilkan luaran seperti berikut ini :

Gambar 4.17 Luaran *Unit Testing* Iterasi Pertama (4)

```

1 public function test_non_teacher_user_failed_on_creating_classroom(): void
2 {
3     $user = User::factory()->create([
4         'role'      => 0
5     ]);
6
7     if($user->role == 1) {
8         $classroom = Classroom::factory()->create([
9             'teacher_id'   => $user->id,
10            'name'          => "Classroom Created by Non-Teacher",
11        ]);
12    }
13
14    $this->assertDatabaseMissing('classrooms', [
15        'name' => 'Classroom Created by Non-Teacher',
16    ]);
17 }

```

Gambar 4.18 Kode *Unit Testing* Iterasi Pertama (5)

Potongan kode program di atas menampilkan *unit test* ketika aktor bukan Guru hendak membuat kelas. Pada baris 3, dibuatkan dahulu *user* baru menggunakan *factory*, seperti yang telah disebutkan sebelumnya, *role* dengan kode 1 merupakan kode *role* untuk Guru, namun yang digunakan pada pengujian tersebut adalah kode *role* 0. Kemudian pada baris 7 dilakukan pengecekan apakah *role* dari *user* yang telah dibuat adalah Guru (bernilai 1), jika tidak maka langsung menuju baris 14 sampai 16, dimana pengujian menuntut nilai *true* apakah di tabel *classrooms* di basis data tidak terdapat baris yang memiliki nilai atribut *name* “Classroom Created by Non-Teacher”. Jika tuntutan tersebut terpenuhi, maka pengujian akan dinyatakan lolos, dan menampilkan luaran seperti berikut ini :

Gambar 4.19 Luaran *Unit Testing* Iterasi Pertama (5)

```

1 public function test_creating_unique_classroom_enrollment_key(): void
2 {
3     $classrooms      = Classroom::all();
4     $enrollment_keys = $classrooms->pluck('enrollment_key')->toArray();
5
6     $chars = "1234567890ABCDEFGHIJKLMNPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz";
7     $random_string = "";
8     $is_key_enable = false;
9
10    while (!$is_key_enable){
11        for ($i = 0; $i < 10; $i++){
12            $index = rand(0, strlen($chars) - 1);
13            $random_string .= $chars[$index];
14        }
15        if (!in_array($random_string, $enrollment_keys)){
16            $is_key_enable = true;
17        }
18    }
19
20    $this->assertDatabaseMissing('classrooms', [
21        'enrollment_key' => $random_string,
22    ]);
23 }

```

Gambar 4.20 Kode *Unit Testing* Iterasi Pertama (6)

Potongan kode program di atas menampilkan *unit test* ketika membuatkan *enrollment key* yang unik ketika hendak membuat kelas. Pada baris 3, diambil terlebih dahulu seluruh data kelas yang tersimpan di basis data seperti yang ditampilkan pada Gambar 4.8, dan diambil nilai *enrollment_key*-nya saja seperti yang ditampilkan pada baris 4. Kemudian di baris 6 sampai 8 didefinisikan variabel yang dibutuhkan. Selanjutnya dijalankan proses pembuatan *random_string*, seperti yang ditampilkan pada baris 10 hingga 19. Di akhir, pengujian menuntut nilai *true* apakah di tabel *classrooms* di basis data tidak terdapat baris yang memiliki nilai atribut *enrollment_key* yang sama dengan nilai dari nilai variabel *random_string* yang baru saja dibuat. Jika tuntutan tersebut terpenuhi, maka pengujian akan dinyatakan lolos, dan menampilkan luaran seperti berikut ini :

✓ creating unique classroom enrollment key	0.05s
--	-------

Gambar 4.21 Luaran *Unit Testing* Iterasi Pertama (6)

```

1 public function test_student_enroll_to_new_classroom(): void
2 {
3     $user      = User::find(10);
4     $classroom = Classroom::find(1);
5     $classroom->load('students');
6     $classroom_students = $classroom->students->pluck('id')->toArray();
7
8     if(!in_array($user->id, $classroom_students)) {
9         $classroom->students()->attach($user->id);
10    }
11
12    $this->assertDatabaseHas('classroom_member', [
13        'classroom_id'  => $classroom->id,
14        'student_id'   => $user->id
15    ]);
16 }

```

Gambar 4.22 Kode *Unit Testing* Iterasi Pertama (7)

Potongan kode program di atas menampilkan *unit test* ketika seorang Siswa hendak bergabung ke dalam kelas. Pada baris 3, diambil terlebih dahulu salah satu data Siswa, dari entitas *user* dengan *id* 10, seperti ditampilkan pada Gambar 4.7 bahwa *user* dengan *id* 10 memiliki kode *role* 2 yang mewakili aktor Siswa. Kemudian dari baris 4 sampai 6 diambil salah satu data kelas dengan *id* 1, mengambil data seluruh siswanya, dan mengambil nilai *id* saja dari seluruh data siswa tersebut. Selanjutnya di baris 8, dilakukan pengecekan apakah *id* 10 dari *user* yang dipilih terdapat dalam *array id* yang diambil dari data seluruh siswa di kelas yang telah diambil. Jika tidak, maka buatkan koneksi baru pada *pivot table classroom_member* dengan kode seperti pada baris 9. Di akhir, pengujian menuntut nilai *true* apakah kini di tabel *classroom_member* di basis data terdapat baris yang memiliki nilai atribut *classroom_id* yang sama dengan nilai dari *id classroom* yang diambil, yaitu 1, dan memiliki nilai atribut *student_id* yang sama dengan nilai dari *id user* yang diambil, yaitu 10. Jika tuntutan tersebut terpenuhi, maka pengujian akan dinyatakan lolos, dan menampilkan luaran seperti berikut ini :

Gambar 4.23 Luaran *Unit Testing* Iterasi Pertama (7)

```

1 public function test_student_failed_to_enroll_to_same_classroom_more_than_once(): void
2 {
3     $user      = User::find(10);
4     $classroom = Classroom::find(1);
5     $classroom->load('students');
6     $classroom_students = $classroom->students->pluck('id')->toArray();
7
8     if(!in_array($user->id, $classroom_students)) {
9         $classroom->students()->attach($user->id);
10    }
11
12    $expectedCount = 1;
13
14    $count = DB::table('classroom_member')
15        ->where('classroom_id', $classroom->id)
16        ->where('student_id', $user->id)
17        ->count();
18
19    $this->assertEquals($expectedCount, $count);
20 }

```

Gambar 4.24 Kode *Unit Testing* Iterasi Pertama (8)

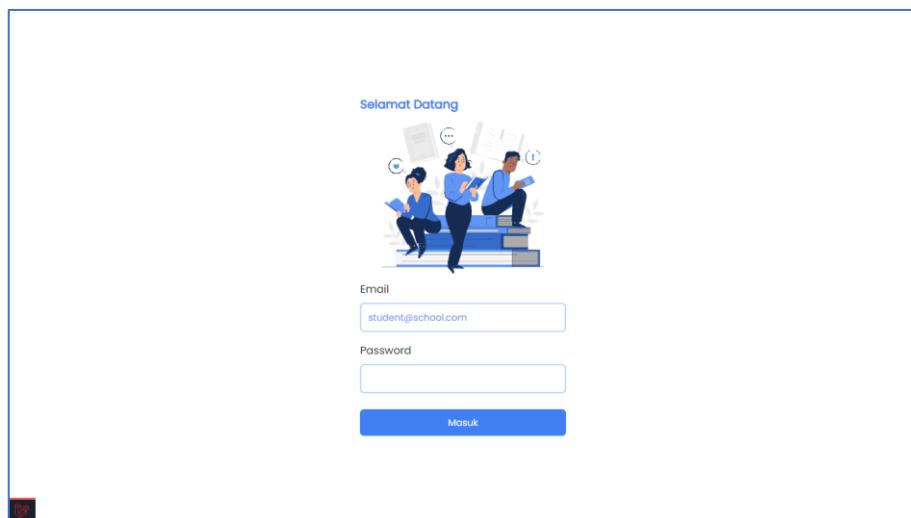
Potongan kode program di atas menampilkan *unit test* ketika seorang Siswa hendak bergabung ke dalam kelas yang sama lebih dari sekali. Pada baris 3, diambil terlebih dahulu salah satu data Siswa, dari entitas *user* dengan *id* 10, seperti ditampilkan pada Gambar 4.7 bahwa *user* dengan *id* 10 memiliki kode *role* 2 yang mewakili aktor Siswa. Kemudian dari baris 4 sampai 6 diambil salah satu data kelas dengan *id* 1, mengambil data seluruh siswanya, dan mengambil nilai *id* saja dari seluruh data siswa tersebut. Perlu diketahui pada *unit test* sebelumnya, basis data telah terpengaruh, dan kini sudah terdapat *id* 10 pada *array* berisi *id* seluruh Siswa di dalam kelas dengan *id* 3. Selanjutnya di baris 8, dilakukan pengecekan apakah *id* 10 dari *user* yang dipilih terdapat dalam *array* *id* yang diambil dari data seluruh siswa di kelas yang telah diambil. Jika ada, maka langsung menuju baris 12, untuk menetapkan nilai yang diharapkan, yaitu 1. Selanjutnya di baris 14 sampai 17, dilakukan *query* database untuk mengecek jumlah baris di tabel *classroom_member* yang memiliki nilai atribut *classroom_id* yang sama dengan nilai dari *id classroom* yang diambil, yaitu 1, dan memiliki nilai atribut *student_id* yang sama dengan nilai dari *id user* yang diambil, yaitu 10. Di akhir, pengujian menuntut nilai *true* apakah nilai yang didapatkan dari *query* basis data sama dengan nilai yang diharapkan, yaitu 1. Jika tuntutan tersebut terpenuhi, maka pengujian akan dinyatakan lolos, dan menampilkan luaran seperti berikut ini :

✓ student failed to enroll to same classroom more than once 0.08s

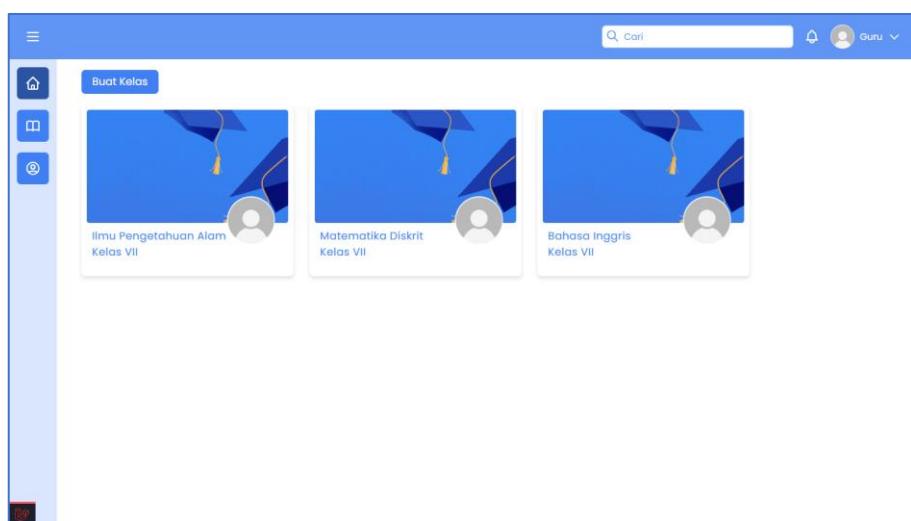
Gambar 4.25 Luaran *Unit Testing* Iterasi Pertama (8)

B. *Code Generation*

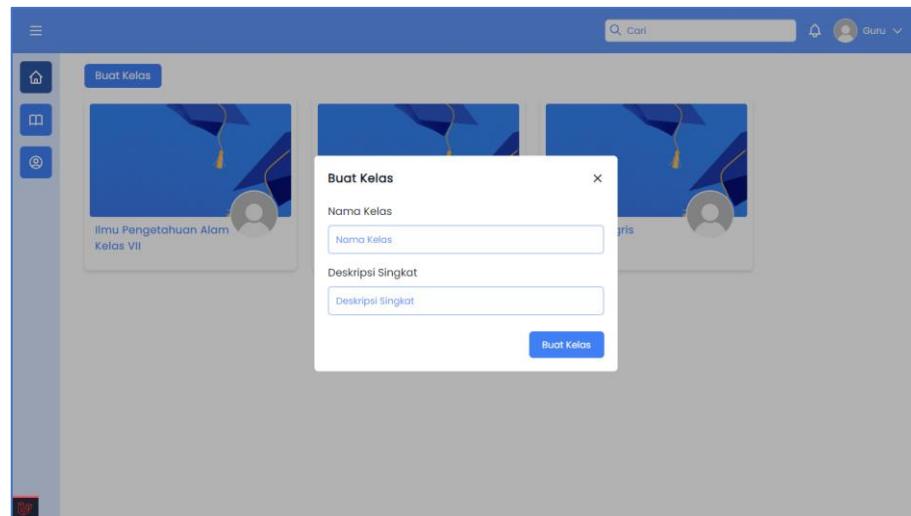
Berikut ini adalah tampilan sistem hasil implementasi pada iterasi ini :



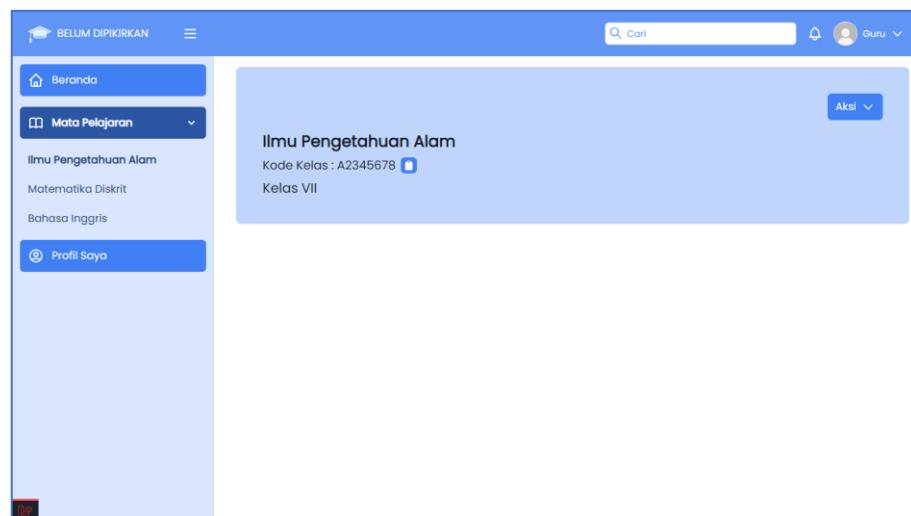
Gambar 4. 26 Tampilan Login Untuk Semua Aktor



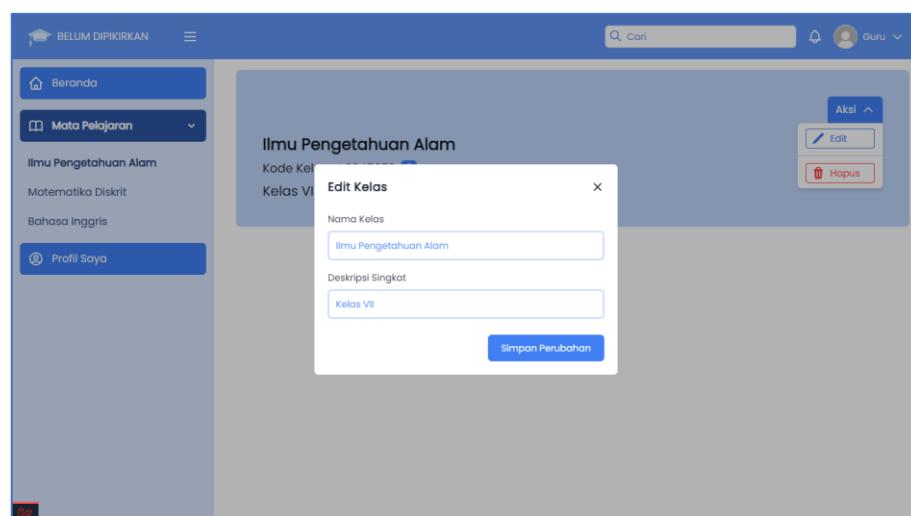
Gambar 4.27 Halaman Awal Setelah Login untuk Guru



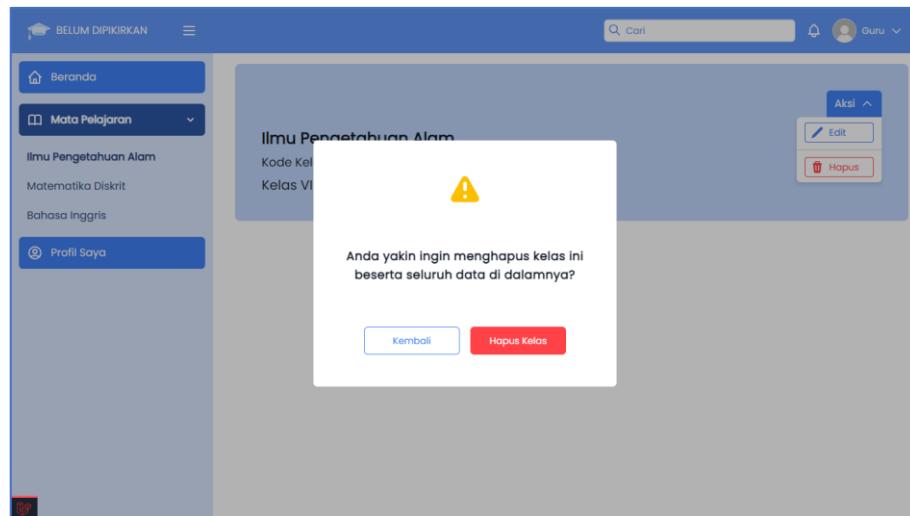
Gambar 4.28 Tampilan "Buat Kelas" untuk Guru



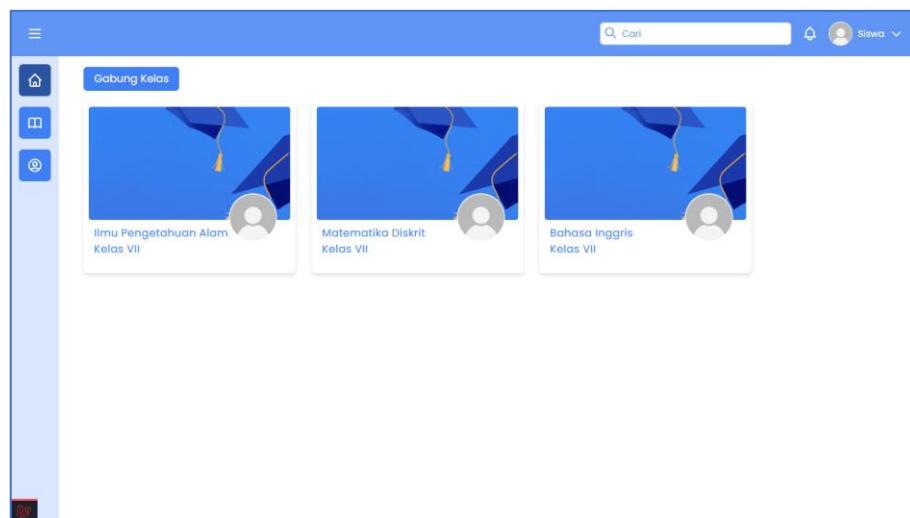
Gambar 4.29 Halaman Detail Kelas untuk Guru



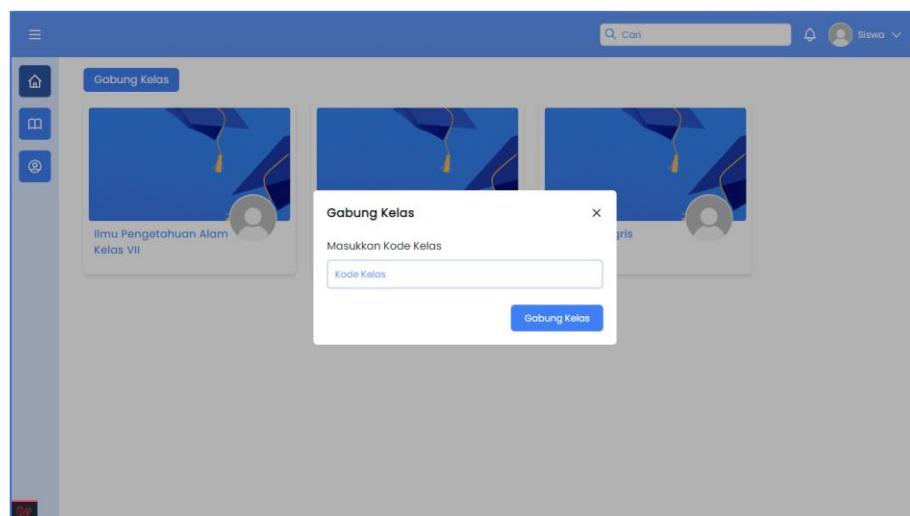
Gambar 4.30 Tampilan "Ubah Kelas" untuk Guru



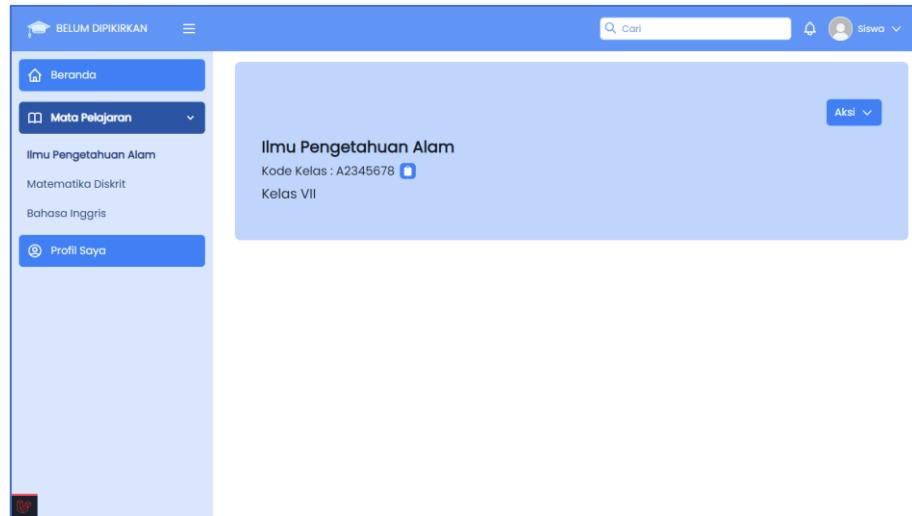
Gambar 4.31 Tampilan Konfirmasi "Hapus Kelas" untuk Guru



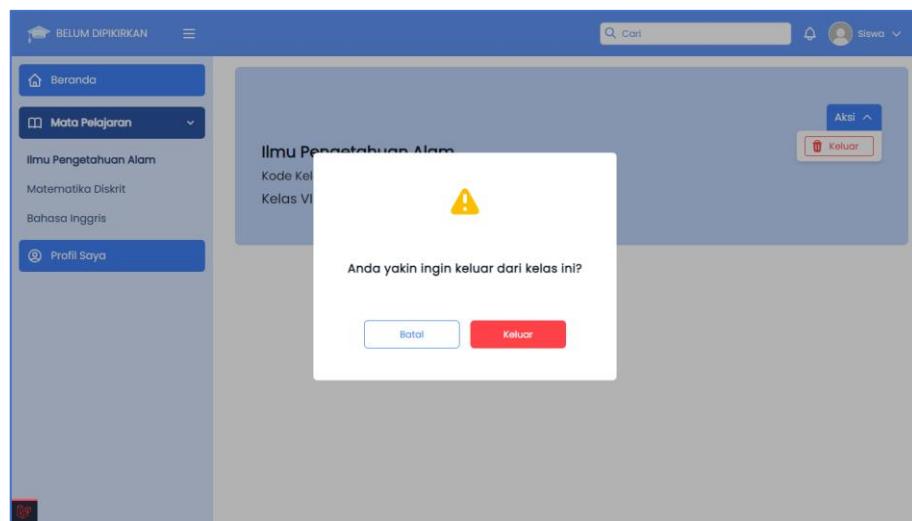
Gambar 4.32 Halaman Awal Setelah Login untuk Siswa



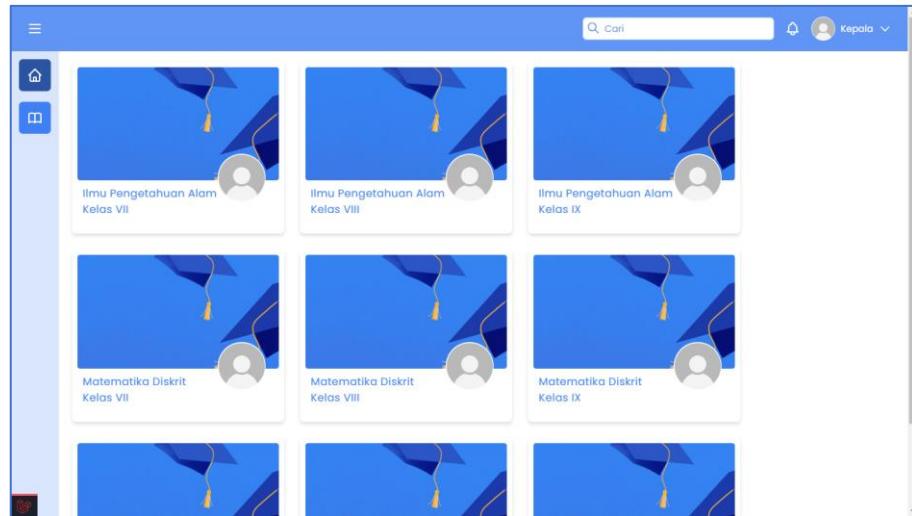
Gambar 4.33 Tampilan Form "Gabung Kelas" untuk Siswa



Gambar 4.34 Halaman "Detail Kelas" untuk Siswa



Gambar 4.35 Tampilan Konfirmasi "Keluar Kelas" untuk Siswa



Gambar 4.36 Halaman Awal Setelah Login untuk Kepala Sekolah

This screenshot shows the detailed view for the Science Class VII class. On the left, there is a sidebar menu with 'Beranda' and 'Mata Pelajaran' sections. The 'Mata Pelajaran' section lists various subjects and their corresponding class levels. The main content area displays the following information for the Science Class VII class:

Ilmu Pengetahuan Alam
Kode Kelas : A2345678
Kelas VII

Gambar 4.37 Halaman Detail Kelas untuk Kepala Sekolah

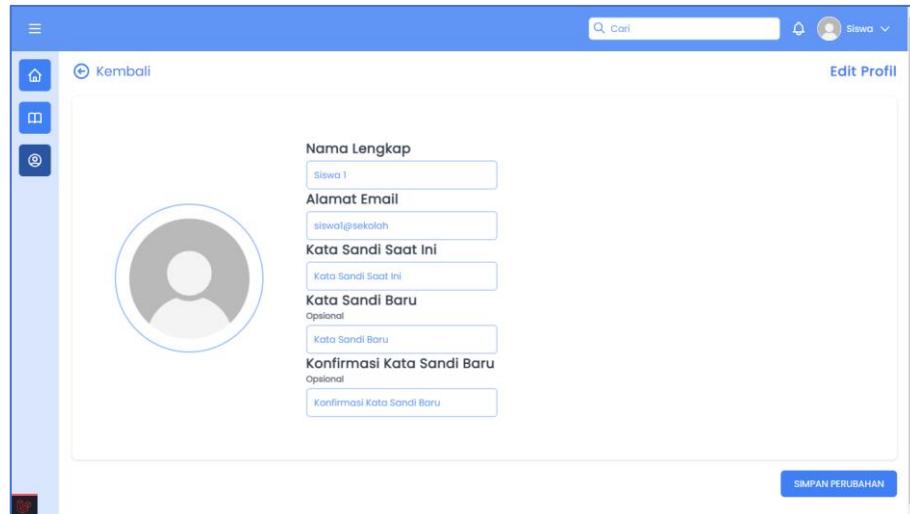
This screenshot shows the 'Lihat Profil' (View Profile) page for a student. The page has a sidebar with icons for home, dashboard, and profile. The main content area displays the student's profile information:

Profil Pengguna

Nama Lengkap
Siswa 1
Alamat Email
siswa1@sekolah

There is also an 'Edit Profil' button in the top right corner.

Gambar 4.38 Halaman "Lihat Profil" untuk Guru dan Siswa



Gambar 4.39 Halaman "Ubah Profil" untuk Guru dan Siswa

C. *Refactoring*

Pengembang akan menyesuaikan kembali sistem yang sedang dikembangkan dengan user acceptance test yang terdapat pada setiap user story. Berikut merupakan tabel kesesuaian user acceptance test pada iterasi ini.

Tabel 4.2 *Refactoring* Iterasi Pertama

No.	Kode User Stories	Judul	Aktor	User Acceptance Test	Status UAT
1	<i>Story-02</i>	Login Aplikasi	Seluruh Pengguna	Seluruh pengguna dapat login ke dalam aplikasi dengan akun dan <i>role</i> -nya masing-masing.	Terpenuhi
2	<i>Story-04</i>	Pengelolaan Kelas	Guru	Guru dapat melakukan pengelolaan kelas.	Terpenuhi

Kedua *user story* pada Tabel 4.2 berhasil dikembangkan sesuai dengan *user acceptance test*, dan tidak perlu dilakukan *refactoring* kode program. Berdasarkan hasil tersebut maka tahap implementasi telah selesai dan siklus pengembangan dapat berlanjut ke pengujian sistem.

4.2.5.2. Pengujian Sistem

Pengujian Sistem dalam iterasi ini dilakukan pengecekan sistem bersama aktor yang berkaitan dengan pengujian yang akan dilakukan

Tabel 4.3 Pengujian Sistem Iterasi Pertama

No	Skenario Pengujian	Aktor	Hasil Diharapkan	Status
1	Login ke dalam aplikasi	Seluruh aktor	Seluruh aktor / pengguna dapat login ke dalam aplikasi menggunakan akun dengan <i>role</i> -nya masing-masing	Berhasil
2	Login ke dalam aplikasi dan melakukan pengelolaan kelas	Guru	Aktor yang dapat membuat, mengubah dan menghapus kelas hanyalah Guru.	Berhasil
3	Login ke dalam aplikasi dan melakukan bergabung ke atau keluar dari kelas	Siswa	Aktor Siswa dapat bergabung ke suatu kelas jika memasukkan kode yang sesuai dan berikutnya dapat mengakses data yang sesuai dalam kelas tersebut.	Berhasil
4	Login ke dalam aplikasi dan melihat kelas	Kepala Sekolah	Aktor Kepala Sekolah tidak dapat membuat, mengubah, maupun menghapus kelas, namun bisa mengakses seluruh kelas yang tersimpan tanpa perlu memasukkan kode kelas tersebut melalui fitur Gabung ke Kelas.	Berhasil

4.2.5.3. Retrospektif

Di akhir iterasi pertama ini, dilakukan evaluasi terhadap pengembangan yang telah dilakukan, dijelaskan pada Tabel 4.4.

Tabel 4.4 Retrospektif Iterasi Pertama

No	Kode User Story	Prioritas	Estimasi (hari)	Realisasi (hari)	Catatan Pengembang
1	Story-02	Must Have	4	2	Realisasi pengerjaan lebih singkat dari waktu yang diestimasikan
2	Story-04	Must Have	6	5	Realisasi pengerjaan lebih singkat dari waktu yang diestimasikan

Pada iterasi pertama ini, pengembang menemukan bahwa waktu sebenarnya yang dibutuhkan untuk menyelesaikan implementasi *user stories* lebih singkat dari waktu yang telah diestimasikan sebelumnya pada tahap perencanaan. Disarankan untuk pengembangan di masa mendatang menggunakan metode PXP ini, perlu dilakukan analisis mendalam terkait kesulitan *user stories* dalam suatu iterasi, sehingga estimasi yang diberikan terkait waktu pengerjaannya, dapat lebih sesuai.

4.2.6 Iterasi Kedua**4.2.6.1. Implementasi**

- A. *Unit Testing*
- B. *Code Generation*
- C. *Refactoring*

4.2.6.2. Pengujian Sistem**4.2.6.3. Retrospektif****4.2.7 Iterasi Ketiga****4.2.7.1. Implementasi**

- A. *Unit Testing*
- B. *Code Generation*
- C. *Refactoring*

4.2.7.2. Pengujian Sistem**4.2.7.3. Retrospektif****4.2.8 Iterasi Keempat****4.2.8.1. Implementasi**

- A. *Unit Testing*
- B. *Code Generation*
- C. *Refactoring*

4.2.8.2. Pengujian Sistem**4.2.8.3. Retrospektif****4.3. Rangkuman Iterasi**

DAFTAR PUSTAKA

- [1] A. B. Sidiq and D. Kurniadi, “Perancangan Sistem Informasi Ujian Online Berbasis Web pada SMK N 1 Solok,” *Voteteknika (Vocational Teknik Elektronika dan Informatika)*, vol. 9, no. 2, p. 44, Jun. 2021, doi: 10.24036/voteteknika.v9i2.111521.
- [2] E. O. Choiri, “10 Aplikasi Ujian Online Terbaik, Gratis Pakai!,” 2021. Qwords.com (accessed Nov. 16, 2022).
- [3] D. Yan, A. A. Rupp, and P. W. Foltz, *Handbook of Automated Scoring Theory into Practice*. Boca Raton: Taylor & Francis Group, LLC, 2020.
- [4] M. D. Shermis and J. C. Burstein, *Automated Essay Scoring: A Cross-Disciplinary Perspective*. New Jersey: Lawrence Erlbaum Associates, Inc., 2003.
- [5] J. F. Dooley, *Software Development, Design and Coding*. Berkeley, CA: Apress, 2017. doi: 10.1007/978-1-4842-3153-1.
- [6] Y. Dzhurov, I. Krasteva, and S. Ilieva, “Personal Extreme Programming – An Agile Process for Autonomous Developers,” Jan. 2009.
- [7] M. van Deurzen, “The anatomy of the modern window manager,” Bachelor thesis, Radboud University, Nijmegen, 2019.
- [8] G. E. Iyawa, “Personal Extreme Programming: Exploring Developers’ Adoption,” in *AMCIS 2020 Proceedings*, bepress, 2020.
- [9] S. A. Asri, I. G. M. A. Sunaya, P. M. Prihatini, and W. Setiawan, “Comparing Traditional and Agile Software Development Approaches: Case of Personal Extreme Programming,” in *Proceedings of the International Conference on Science and Technology (ICST 2018)*, 2018. doi: 10.2991/icst-18.2018.116.
- [10] A. N. Yusril, I. Larasati, and P. Al Zukri, “Systematic Literature Review Analisis Metode Agile dalam Pengembangan Aplikasi Mobile,” *Sistemasi : Jurnal Sistem Informasi*, vol. 10, no. 2, May 2021.
- [11] M. Ulfia, G. I. Marthasari, and I. Nuryasin, “Implementasi Metode Personal Extreme Programming dalam Pengembangan Sistem Manajemen Transaksi Perusahaan,” *Jurnal Reppositor*, vol. 2, no. 3, Mar. 2020.
- [12] G. Marthasari, W. Suharso, and F. A. Ardiansyah, “Personal Extreme Programming with MoSCoW Prioritization for Developing Library Information

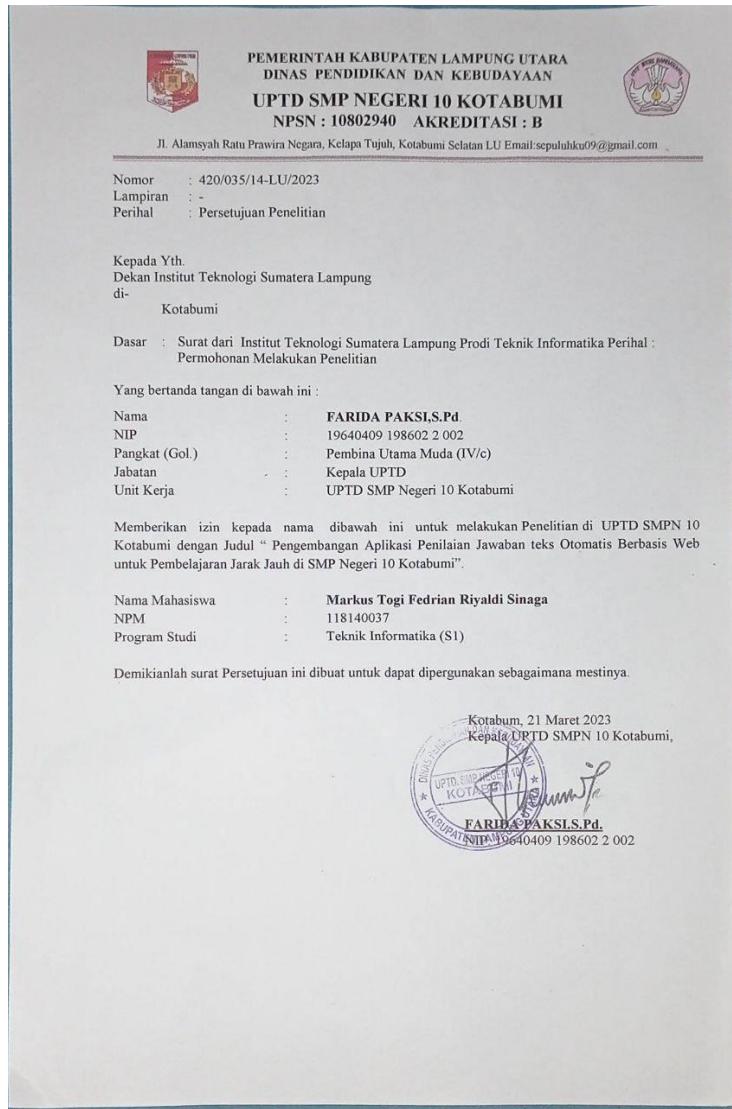
- System," *Proceeding of the Electrical Engineering Computer Science and Informatics*, vol. 5, no. 1, Nov. 2018, doi: 10.11591/eecsi.v5.1701.
- [13] A. F. Septiyanto, W. Suharso, and I. Nuryasin, "Sistem Informasi Program Keluarga Harapan (PKH) Menggunakan Metode Personal Extreme Programming dengan Metode Prioritas Ranking," *Jurnal Reppositor*, vol. 2, no. 12, pp. 1671–1678, Dec. 2020, doi: 10.22219/repositor.v2i12.607.
 - [14] M. Kancharla, G. S. Kamisetty, and S. H. Dudekula, "Muster: Virtual Classroom For Students Using D-Jango," *International Research Journal of Modernization in Engineering Technology and Science*, vol. 4, no. 5, May 2022.
 - [15] A. S. Rosa and M. Salahuddin, *Modul Pembelajaran Rekayasa Perangkat Lunak (Terstruktur dan Berorientasi Objek)*. Bandung: Bandung Modula, 2011.
 - [16] S. A. Asri and W. Setiawan, "Alternatif Penggunaan Model Pendekatan Agile pada Perancangan Sistem Informasi PKL Online," *Jurnal Manajemen Teknologi dan Informatika*, vol. 5, no. 3, 2017.
 - [17] G. Booch, J. Rumbaugh, and I. Jacobson, *The Unified Modeling Language User Guide*, 1st ed. Boston: Addison Wesley, 1998.
 - [18] M. R. Sanjaya, A. Saputra, and D. Kurniawan, "Penerapan Metode System Usability Scale (SUS) Perangkat Lunak Daftar Hadir Di Pondok Pesantren Miftahul Jannah Berbasis Website," *Jurnal Komputer Terapan*, no. Vol. 7 No. 1 (2021), pp. 120–132, Jun. 2021, doi: 10.35143/jkt.v7i1.4578.
 - [19] Rasmila, "Evaluasi Website Dengan Menggunakan System Usability Scale (SUS) Pada Perguruan Tinggi Swasta di Palembang," *JUSIFO : Jurnal Sistem Informasi*, vol. 4, no. 1, Jun. 2018.
 - [20] J. Sauro, "Measuring Usability With The System Usability Scale (SUS)," *Measuring U*, Feb. 03, 2011.
 - [21] F. Firmansyah, "Implementasi System Usability Scale pada Sistem Informasi Manajemen Anggaran dan Kegiatan di Badan Pusat Statistik," *Technologia: Jurnal Ilmiah*, vol. 12, no. 3, p. 165, Jul. 2021, doi: 10.31602/tji.v12i3.5180.
 - [22] A. Bangor, P. Kortum, and J. Miller, "Determining What Individual SUS Scores Mean: Adding an Adjective Rating Scale," *Journal of User Experience*, vol. 4, no. 3, 2009.

LAMPIRAN

No.	Lampiran
1	Folder Google Drive Rekaman Wawancara I 
2	Dokumentasi Wawancara II 

3

Surat Izin Melakukan Penelitian



label	Dokumen Fikri
label	Dokumen Roman
label	Draft Awal Pendaftaran TA
label	Pengajuan ke Dosen Pembimbing dan PURINO