Rancang Bangun Learning Management System Berbasis Web dengan Online Compiler dan OOP Analyzer

Muhammad Zaini Rochman[1], Martini Dwi Endah Susanti[2], Rindu Puspita Wibawa[3], Mohammad Wildan Habibi[4]

[1,2,3,4] Pendidikan Teknologi Informasi, Universitas Negeri Surabaya
Surabaya, Indonesia
[1]muhammadzaini.22036@unesa.ac.id
[2]martinisusanti@unesa.ac.id
[3]rinduwibawa@unesa.ac.id
[4]mohammadhabibi@unesa.ac.id

Abstrak*—Penelitian ini bertujuan untuk merancang dan membangun sebuah Learning Management System (LMS) berbasis website yang dilengkapi fitur Online Compiler dan OOP Analyzer untuk mendukung praktik Pemrograman Berorientasi Objek (OOP) pada siswa Sekolah Menengah Kejuruan (SMK). Penelitian ini menggunakan metode Research and Development (R&D) dengan model pengembangan ADDIE (Analysis, Design, Development, Implementation, Evaluation). LMS dikembangkan dengan arsitektur full-stack menggunakan Next.js dan basis data Supabase (PostgreSQL). Fitur Java Online Compiler diintegrasikan dengan Piston API, sedangkan OOP Analyzer dibangun berbasis Regular Expression (Regex) untuk mendeteksi penerapan lima pilar OOP secara real-time. Sistem ini juga dilengkapi modul Catatan Aktivitas (Logbook) Proyek untuk mendukung monitoring proyek pada pembelajaran model Project-Based Learning (PjBL). Perancangan sistem dimodelkan menggunakan use case diagram, activity diagram, dan class diagram. Pengujian fungsional dan non-fungsional melalui metode blackbox testing menunjukkan sistem berjalan dengan validitas 99%. Hasil ini menunjukkan bahwa seluruh modul sistem, termasuk Online Compiler dan OOP Analyzer, dapat berjalan sesuai rancangan tanpa kecacatan (bug) mayor. Dengan demikian, LMS yang dikembangkan dapat diimplementasikan sebagai media pembelajaran praktik pemrograman berbasis website yang dapat dijalankan melalui browser tanpa memerlukan instalasi compiler tambahan pada perangkat siswa.

Kata Kunci— Learning Management System, Online Compiler, OOP Analyzer, Next.js, Supabase, Research and Development, Blackbox Testing.

I. PENDAHULUAN

Perkembangan teknologi pada era Revolusi Industri 4.0 menjadikan digitalisasi sebagai tulang punggung di berbagai sektor, sehingga kebutuhan terhadap talenta digital yang kompeten terus meningkat [1]. Kebutuhan tenaga kerja di bidang teknologi informasi, khususnya *programmer*, juga mengalami peningkatan signifikan seiring masifnya ekosistem digital pada berbagai perusahaan [2]. Sekolah Menengah Kejuruan (SMK), khususnya program keahlian Rekayasa Perangkat Lunak (RPL), berada di garda terdepan dalam usaha mencetak calon *programmer* yang siap kerja. Pendidikan kejuruan membekali siswa dengan keterampilan praktis sesuai dengan standar industri [3], [4].

Salah satu kompetensi fundamental yang wajib dikuasai oleh *programmer* adalah Pemrograman Berorientasi Objek (OOP). OOP secara spesifik menguraikan kemampuan yang dibutuhkan, yaitu membuat program dengan memanfaatkan *class*, menerapkan konsep *inheritance* (pewarisan), menggunakan *interface*, serta mengompilasi program. Penguasaan OOP menjadi krusial karena mendasari pengembangan sebagian besar perangkat lunak modern, mulai dari aplikasi *desktop*, *website*, hingga *mobile*. Memahami konsep OOP menuntut lebih dari sekadar paparan teori; siswa memerlukan pendekatan yang memungkinkan mereka menulis kode secara langsung dan melihat hasil kompilasinya, sehingga dibutuhkan sistem yang dilengkapi *compiler* dan umpan balik otomatis terhadap penerapan konsep OOP pada kode yang ditulis.

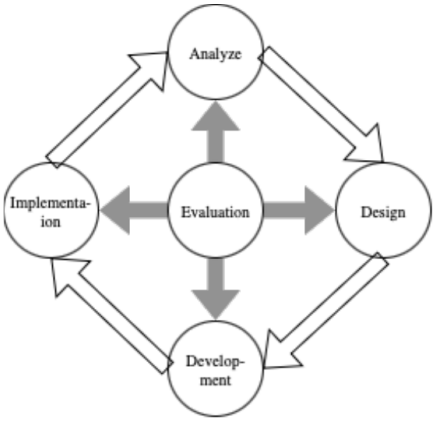
Berdasarkan wawancara dengan siswa kelas X RPL di SMK Antartika 2 Sidoarjo yang dilakukan pada tanggal 18 November 2025, permasalahan utama terletak pada sistem pembelajaran yang berjalan saat ini. Implementasi pembelajaran masih mengandalkan Google Workspace, yang menyebabkan materi, tugas, dan komunikasi tersebar di berbagai platform seperti Google Classroom dan Google Drive. Sistem yang tidak terpusat ini menyulitkan siswa dalam mengakses sumber belajar secara terstruktur [5]. Lebih lanjut, penerapan *Project-Based Learning* (PjBL) yang sudah diterapkan di sekolah cenderung masih sebatas pemberian tugas proyek akhir saja, tanpa proses *monitoring* yang terstruktur. Kendala lainnya adalah mayoritas siswa kelas X belum memiliki laptop pribadi. Akibatnya, mereka merasa waktu praktik di laboratorium sekolah tidak mencukupi untuk mengasah kemampuan *coding* secara mandiri.

Berdasarkan permasalahan tersebut, dibutuhkan sebuah *Learning Management System* (LMS) berbasis *website* yang terpusat dan dilengkapi modul-modul teknis untuk menjawab permasalahan tersebut [6]. Berbeda dengan LMS berbasis Moodle pada penelitian terdahulu yang sudah mengintegrasikan PjBL namun masih bergantung pada aplikasi pengolah kode (*Integrated Development Environment*) eksternal [4], penelitian ini mengembangkan LMS yang dirancang khusus untuk mata pelajaran pemrograman dengan dua modul teknis utama. *Pertama*, modul *Java Online Compiler* yang terintegrasi dengan *Piston API* dan dilengkapi *OOP Analyzer* berbasis *Regular Expression* (Regex) untuk memberikan umpan balik penerapan konsep OOP secara *real-time*, sehingga siswa dapat menulis dan menjalankan kode langsung melalui *browser* tanpa terbatas oleh spesifikasi perangkat. *Kedua*, modul Catatan Aktivitas (*logbook*) Proyek yang memungkinkan guru memantau kemajuan dan kendala

setiap kelompok proyek secara berkala. Penelitian ini bertujuan merancang, membangun, dan menguji LMS tersebut menggunakan metode *Research and Development* (R&D) dengan model ADDIE, sehingga dihasilkan sebuah sistem yang valid dan fungsional untuk diimplementasikan pada pembelajaran pemrograman di SMK.

II. METODE PENELITIAN

Penelitian ini menggunakan model ADDIE (*Analysis, Design, Development, Implementation,* dan *Evaluation*) sesuai Gbr. 1 sebagai siklus pengembangan perangkat lunak (*software development life cycle*) yang diadaptasi pada konteks pengembangan media pembelajaran. Model ini terbagi menjadi lima tahapan sebagai berikut.



Gbr. 1 Model Pengembangan ADDIE sebagai Siklus Pengembangan Perangkat Lunak (Sumber: [9])

A. Tahap Analisis (analysis)

Tahap analisis merupakan fondasi dari seluruh proses pengembangan yang bertujuan untuk mengidentifikasi akar permasalahan dan kebutuhan pengguna. Analisis dilakukan melalui observasi dan wawancara dengan guru mata pelajaran Pemrograman Berorientasi Objek (OOP) dan siswa kelas X RPL di SMK Antartika 2 Sidoarjo pada tanggal 18 November 2025. Hasil analisis kemudian dirumuskan menjadi kebutuhan perangkat lunak, yaitu kebutuhan fungsional dan non-fungsional, yang menjadi dasar perancangan LMS terintegrasi *Project-Based Learning* (PjBL).

B. Tahap Perancangan (design)

Tahap ini merupakan proses perancangan cetak biru (*blueprint*) sistem sebelum dilakukan *development*. Perancangan meliputi pemodelan arsitektur sistem *full-stack*, pemodelan perilaku pengguna menggunakan *Use Case Diagram* dan *Activity Diagram*, serta perancangan struktur basis data berupa *Class Diagram*. Pada tahap ini juga dirancang mekanisme integrasi fitur *Java Online Compiler* dengan *Piston API* serta algoritma *OOP Analyzer*, agar selaras dengan tahapan pengerjaan proyek siswa.

Secara teknis, mekanisme kerja *Online Compiler* dirancang sebagai berikut:

a. Siswa menuliskan kode program (bahasa Java) pada *editor* teks yang tersedia di antarmuka LMS.
b. Sistem LMS mengirimkan kode tersebut melalui protokol HTTP dengan metode *POST request* ke *endpoint* API.
c. API melakukan kompilasi dan eksekusi kode dalam lingkungan terisolasi (*sandboxed environment*) yang aman pada server penyedia.
d. Hasil eksekusi berupa keluaran (*output*) kode program yang telah dituliskan dikembalikan dan ditampilkan kepada pengguna.

Desain ini memungkinkan siswa untuk melakukan kompilasi kode program Java hanya dengan menggunakan peramban (*browser*), termasuk pada perangkat berspesifikasi rendah.

C. Tahap Pengembangan (development)

Tahap pengembangan adalah proses merealisasikan seluruh rancangan menjadi sebuah LMS yang fungsional. Pengembangan sisi *frontend* dan *backend* dibangun dalam satu arsitektur menggunakan *framework* Next.js dengan Tailwind CSS untuk penataan antarmuka. Manajemen basis data, otentikasi pengguna, dan keamanan akses dikelola menggunakan layanan Supabase (PostgreSQL) yang dilengkapi *Row Level Security* (RLS). Eksekusi kompilasi kode pada fitur *Online Compiler* diintegrasikan melalui *Piston API*, sedangkan *deployment* sistem dilakukan pada layanan *cloud* Vercel dengan *version control* menggunakan GitHub. Setelah proses pengodean dan *deployment* selesai, produk awal (*prototype*) diuji coba dan dievaluasi pada dua tahap berikutnya.

D. Tahap Implementasi (implementation)

Sistem yang telah selesai dikembangkan kemudian *di-deploy* ke layanan *cloud* Vercel agar dapat diakses secara publik melalui *browser*. Uji coba terbatas dilakukan untuk memastikan integrasi basis data dan fitur *Online Compiler* berjalan stabil ketika menerima beban akses dan penyimpanan data secara bersamaan oleh peran admin, guru, dan siswa pada kelas X RPL SMK Antartika 2 Sidoarjo.

E. Tahap Evaluasi (evaluation)

Tahap evaluasi difokuskan pada pengujian fungsional dan non-fungsional sistem menggunakan metode *Blackbox Testing* yang dilakukan oleh sepuluh responden (admin, guru, dan siswa) sebagai calon pengguna. Pengujian ini bertujuan memastikan seluruh fitur sistem, mulai dari otentikasi, manajemen kelas dan pengguna, manajemen proyek PjBL, Catatan Aktivitas (*logbook*), hingga *Online Compiler* dan *OOP Analyzer*, berjalan sesuai rancangan tanpa adanya *error* atau kecacatan sistem (*bug*).

III. HASIL DAN PEMBAHASAN

Hasil pengembangan *Learning Management System* berbasis *website* berfitur *Online Compiler* dan *OOP Analyzer* ini mengacu pada lima tahapan model ADDIE, dengan penjabaran hasil sebagai berikut.

A. Tahap Analisis

Hasil dari tahap analisis adalah kebutuhan sistem yang dirancang untuk mengatasi permasalahan fragmentasi media pembelajaran, keterbatasan *device* siswa, dan minimnya pemantauan proyek PjBL. Kebutuhan ini dibagi menjadi kebutuhan fungsional dan non-fungsional. Adapun analisis kebutuhan pada *Learning Management System* yang dikembangkan, sebagai berikut:

1) Identifikasi Pengguna: terdapat tiga peran pengguna, yaitu (a) *Admin*, sebagai pengelola akun guru dan siswa, serta menambahkan kelas dan melakukan pendaftaran pengguna pada kelas; (b) *Guru*, sebagai pengelola pembelajaran yang memiliki akses untuk mengunggah materi, asesmen, dan tahapan proyek, serta memberikan umpan balik terhadap *logbook* proyek; dan (c) *Siswa*, sebagai pengguna utama yang memiliki akses untuk melihat materi, mengerjakan asesmen, hingga mengisi *logbook* proyek.

2) Kebutuhan Fungsional: berdasarkan wawancara dan studi literatur, diperoleh hasil analisis kebutuhan fungsional pada Tabel 1 berikut.

Tabel 1 Kebutuhan Fungsional

Fungsi	Deskripsi	Aktor
Login dan logout	Pengguna dapat masuk dan keluar dari sistem menggunakan akun terdaftar.	Admin, guru, siswa
Kelola pengguna	Mengelola data pengguna (tambah, edit, hapus data guru dan siswa).	Admin
Kelola kelas	Mengelola (tambah, edit, hapus) data kelas dan melakukan pendaftaran (enrolment) siswa ke kelas.	Admin
Kelola materi	Mengunggah, mengedit, dan menghapus materi pelajaran (modul, video, presentasi) agar terpusat.	Guru
Akses materi	Melihat dan mengunduh materi pelajaran yang telah diunggah oleh guru.	Siswa, guru
Kelola proyek (PjBL)	Membuat deskripsi proyek, menentukan tenggat waktu, dan mengunggah panduan proyek (Jobsheet).	Guru
Akses proyek (PjBL)	Mengakses detail tugas proyek dan mengunggah hasil akhir proyek.	Siswa, guru
Catatan Aktivitas (logbook) Proyek	Mengisi catatan aktivitas terkait pengerjaan proyek (apa yang dikerjakan, kendala, progres).	Siswa
Monitoring aktivitas proyek	Memantau isian catatan aktivitas siswa, memberikan komentar/umpan balik, dan menandainya sebagai diterima.	Guru
Java Online Compiler	Menulis, mengompilasi, dan menjalankan kode program Java secara real-time.	Siswa, guru
Kelola asesmen	Membuat asesmen/penilaian untuk mengevaluasi pengetahuan siswa.	Guru
Akses asesmen	Mengerjakan asesmen yang telah diberikan oleh guru.	Siswa
Penilaian	Memberikan nilai pada proyek PjBL.	Guru
Beranda	Melihat ringkasan data berdasarkan peran pengguna.	Admin, guru, siswa

3) Kebutuhan Non-Fungsional: analisis kebutuhan non-fungsional dilakukan untuk menetapkan batasan dan standar kualitas sistem agar dapat berjalan secara optimal, dengan merujuk pada standar kualitas perangkat lunak dalam studi literatur terdahulu yang menekankan aspek kinerja dan kemudahan akses [7]. Hasil analisis kebutuhan non-fungsional ditunjukkan pada Tabel 2 berikut.

Tabel 2 Kebutuhan Non-Fungsional

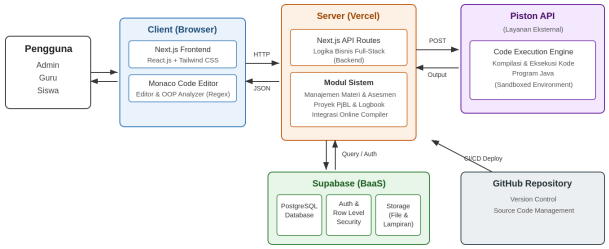
Aspek	Deskripsi
Kinerja (performance)	Waktu muat kurang dari 1000 ms dengan koneksi internet standar. Sistem membatasi ukuran maksimal file yang dapat diunggah.
Kompatibilitas (compatibility)	Sistem dapat diakses dari browser apapun (Google Chrome, Mozilla Firefox, Safari, Microsoft Edge).
Ketergunaan (usability)	Akun hanya dapat dibuat/didaftarkan oleh admin untuk mencegah pendaftaran akun palsu.
Keamanan (security)	Sistem melindungi data nilai siswa dan mekanisme otentikasi login, serta membatasi jenis file yang dapat diunggah, yaitu PDF dan tautan.
Ketersediaan (availability)	Sistem dapat diakses 24 jam sehari dan 7 hari seminggu (24/7) untuk memfasilitasi pembelajaran mandiri di luar jam sekolah.

B. Tahap Perancangan (design)

Berdasarkan hasil analisis kebutuhan, dirancang arsitektur sistem dan alur interaksi pengguna. Arsitektur aplikasi didesain dengan memisahkan antarmuka pengguna (*client*), logika server (*server*), dan layanan pihak ketiga, yaitu Supabase untuk manajemen basis data serta *Piston API* untuk mesin kompilasi kode.

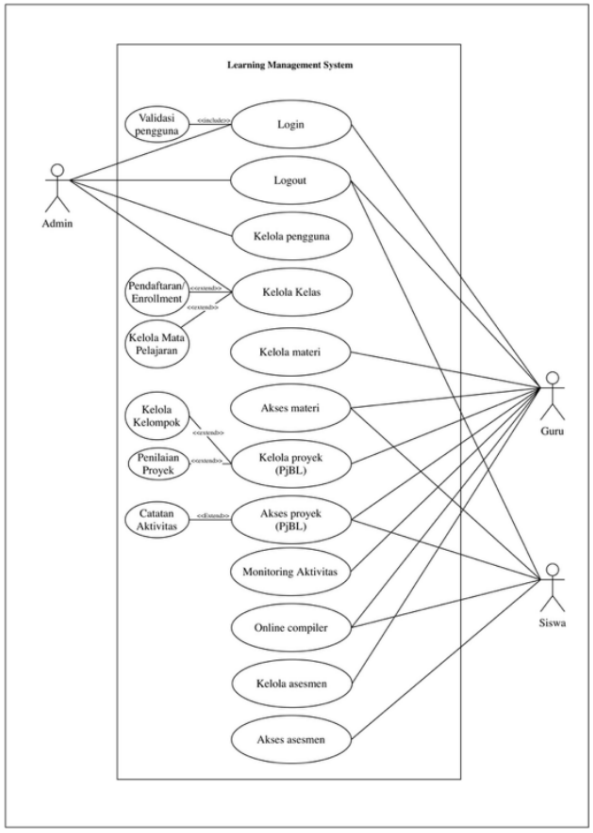
1) Arsitektur Sistem: arsitektur LMS yang dikembangkan terdiri atas tiga lapisan utama sebagaimana ditunjukkan pada Gambar 2. Lapisan *client* berupa *browser* yang menjalankan *Next.js Frontend* (React.js dan Tailwind CSS) beserta *Monaco Code Editor* untuk penulisan kode dan *OOP Analyzer* berbasis *Regex*. Lapisan *server* berupa *Next.js API Routes* yang dihosting pada Vercel, menangani seluruh logika bisnis *full-stack* termasuk manajemen materi, asesmen, proyek PjBL, *logbook*, dan integrasi *Online Compiler*. Lapisan layanan pihak ketiga terdiri atas Supabase sebagai *Backend-as-a-Service* (BaaS) yang menyediakan basis data PostgreSQL,

otentikasi, *Row Level Security*, dan *storage*, serta *Piston API* sebagai mesin eksekusi kode Java. GitHub digunakan sebagai *version control* yang terhubung dengan *pipeline* CI/CD pada Vercel.



Gbr. 2 Arsitektur Sistem LMS Berbasis Website dengan Online Compiler dan OOP Analyzer (Sumber: Peneliti, 2026)

2) Use Case Diagram: Gbr 3 menggambarkan interaksi antara pengguna (admin, guru, siswa) dengan sistem [8]. Diagram ini berfungsi mengidentifikasi fitur-fitur yang dapat diakses oleh setiap peran pengguna dan menjadi acuan dalam pengembangan modul-modul LMS pada penelitian ini. Spesifikasi dari beberapa *use case* utama yang berkaitan dengan modul teknis hasil pengembangan ditunjukkan pada Tabel 3.



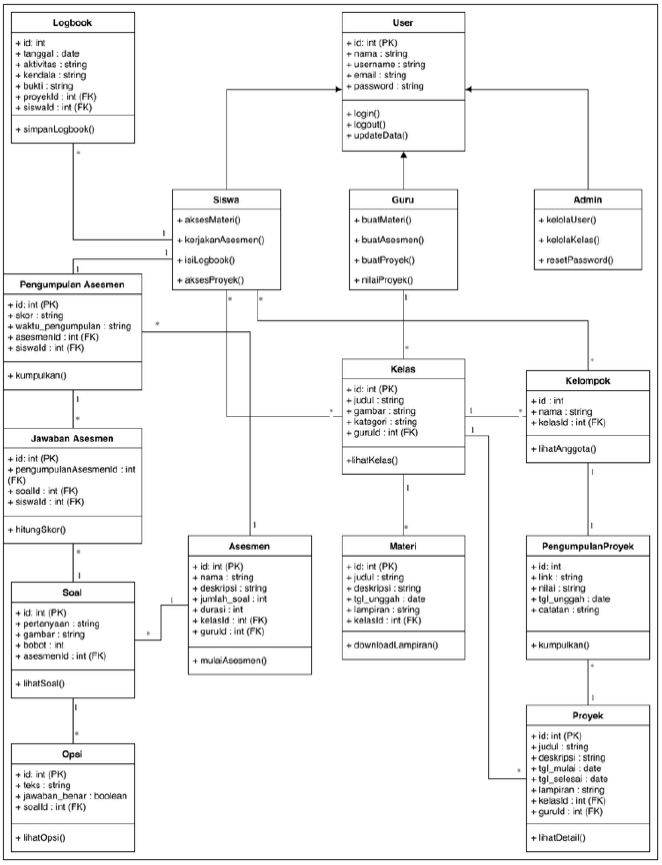
Gbr. 3 Use Case Diagram

Tabel 3 Spesifikasi Use Case

Use Case	Aktor	Kondisi Awal	Kondisi Akhir	Success Scenario	Alternative Scenario
Login/Masuk	Admin, guru, siswa	Pengguna belum masuk ke LMS.	Pengguna berhasil masuk ke LMS dan diarahkan ke dashboard.	Pengguna masuk ke halaman login; (2) pengguna memasukkan email dan password yang diberikan admin; (3) sistem melakukan validasi melalui Supabase Auth; (4) jika valid, pengguna diarahkan ke dashboard sesuai peran.	validasi gagal (email/password salah), sistem menampilkan pesan error.
Online Compiler	Guru, siswa	Online compiler menampilkan kode program Java kosong/template default.	Menampilkan output dari kode program yang dituliskan pengguna.	(1) Pengguna masuk halaman compiler; (2) pengguna menuliskan kode program Java; (3) OOP Analyzer memindai kode secara real-time; (4) pengguna menekan tombol “Jalankan Kode”; (5) sistem menampilkan output kode program yang dituliskan.	Jika kode program yang dituliskan error, sistem menampilkan pesan error pada Output Console.
Catatan Aktivitas (Logbook) Proyek	Siswa	Siswa sedang mengerjakan proyek pada tahapan sintaks PjBL	Catatan aktivitas tersimpan dan dapat dilihat oleh guru.	(1) Siswa masuk halaman catatan aktivitas; (2) siswa memilih tahapan	Jika validasi gagal (jenis/ukuran file tidak sesuai), sistem menampilk

		tertentu.		proyek yang sedang berjalan; (3) siswa mengisi form (tanggal, aktivitas, kendala, lampiran); (4) sistem menyimpan dan menampilkan data pada riwayat aktivitas.	an pesan error.
Monitoring Aktivitas Proyek	Guru	Siswa sudah mengisi catatan aktivitas sesuai tahapannya.	Guru melihat catatan aktivitas setiap siswa dan dapat memberikan umpan balik.	(1) Guru masuk halaman monitoring aktivitas; (2) guru memilih kelas/kelompok siswa; (3) guru membaca catatan aktivitas dan kendala siswa; (4) guru memberikan komentar dan menandai status catatan.	-
Kelola Proyek (PjBL)	Guru	Belum ada tahapan proyek PjBL pada kelas yang diampu.	Tahapan proyek berhasil ditambahkan dan ditampilkan pada halaman siswa.	(1) Guru masuk halaman kelola tahapan proyek; (2) guru memilih kelas dan tahapan sintaks PjBL; (3) guru mengisi deskripsi, tenggat waktu, dan lampiran jobsheet;	Jika validasi gagal, sistem menampilkan pesan error.
				(4) sistem melakukan validasi, menyimpan, dan menampilkan data.	

3) Perancangan Basis Data: basis data memiliki peran sentral dalam menyimpan seluruh data LMS, mulai dari data pengguna, materi, asesmen, catatan aktivitas, hingga proyek yang dikerjakan siswa. Struktur data direalisasikan dalam bentuk *class diagram* pada Gbr 4 yang memuat entitas (tabel), atribut, perilaku setiap *class*, serta relasi antar *class*, sebagai acuan dalam pembuatan skema tabel dan tipe data pada basis data PostgreSQL melalui Supabase.



Gbr. 4 Class Diagram

4) Activity Diagram: menggambarkan alur kerja proses yang dilakukan pengguna terhadap sistem. Dari seluruh modul LMS, dua alur kerja berikut menjadi fokus karena merupakan modul teknis utama hasil pengembangan, yaitu *Online Compiler* dan Catatan Aktivitas (logbook) Proyek.

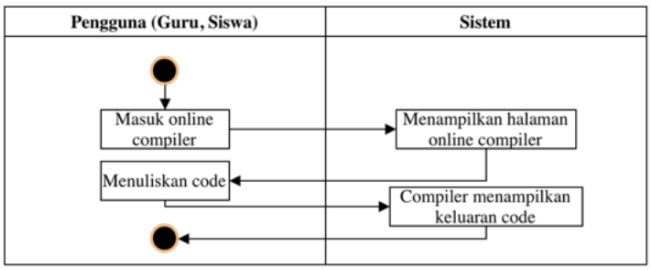
Gbr 5 menunjukkan alur kerja fitur *Online Compiler*. Pengguna (guru atau siswa) masuk ke halaman *online compiler* yang menampilkan templat kode program; pengguna menuliskan kode program, kemudian sistem mengirimkan kode tersebut ke *compiler* dan menampilkan keluaran (*output*) kode program pada antarmuka.

C. Tahap Pengembangan (development)

Tahap pengembangan menghasilkan sebuah *Learning Management System* berbasis *website* yang dapat diakses melalui tautan https://lms-antrek-2.vercel.app/, dengan kode sumber yang dikelola pada repositori GitHub https://github.com/zainiirochman/lms_antrek_2.git. Sistem ini menerapkan tiga peran pengguna, yaitu Admin, Guru, dan Siswa, dengan hak akses yang disesuaikan pada setiap modul.

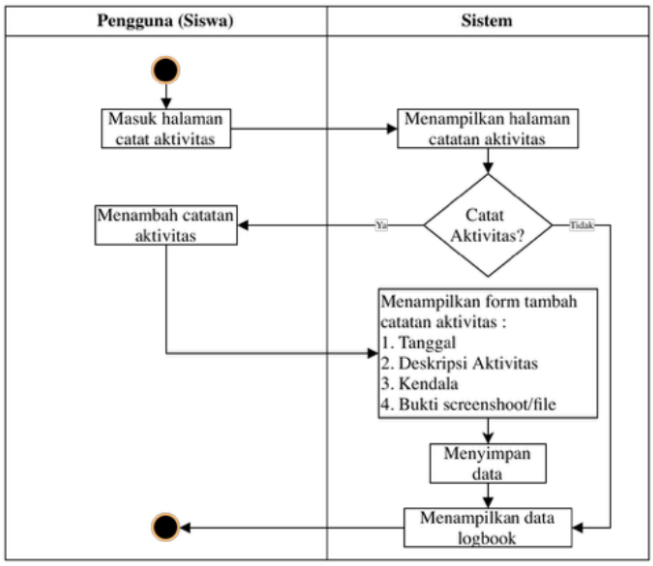
1) Lingkungan dan Tumpukan Teknologi (*Technology Stack*): pengembangan dilakukan menggunakan Visual Studio Code (VS Code) dengan dukungan GitHub Copilot untuk membantu penanganan kesalahan (*error handling*). Sisi *frontend* dan *backend* dibangun dalam satu arsitektur *full-stack* menggunakan *framework* Next.js, dengan Tailwind CSS untuk penataan antarmuka. Manajemen basis data, otentikasi, dan keamanan akses (*Row Level Security*) dikelola menggunakan Supabase berbasis PostgreSQL. *Version control* dikelola menggunakan GitHub, sedangkan *deployment* dilakukan pada layanan *cloud* Vercel sehingga sistem dapat diakses secara publik melalui *browser* dari berbagai perangkat dan sistem operasi.

2) Implementasi Antarmuka Pengguna: tata letak antarmuka dibagi menjadi dua area utama, yaitu panel navigasi (*sidebar*) di sisi kiri dan area konten utama di sisi kanan. Menu *sidebar* disesuaikan dengan kebutuhan masing-masing peran. Hasil implementasi antarmuka untuk ketiga peran dijabarkan pada Gbr 5 sebagai berikut.



Gbr. 5 Activity Diagram Java Online Compiler

Gbr 6 menunjukkan alur kerja fitur Catatan Aktivitas (logbook) Proyek. Siswa masuk ke halaman catat aktivitas, kemudian menambahkan catatan aktivitas melalui form yang memuat tanggal, deskripsi aktivitas, kendala yang dihadapi, serta bukti screenshot/file pendukung. Data yang disimpan akan ditampilkan pada riwayat logbook sehingga dapat dipantau secara berkala oleh guru.



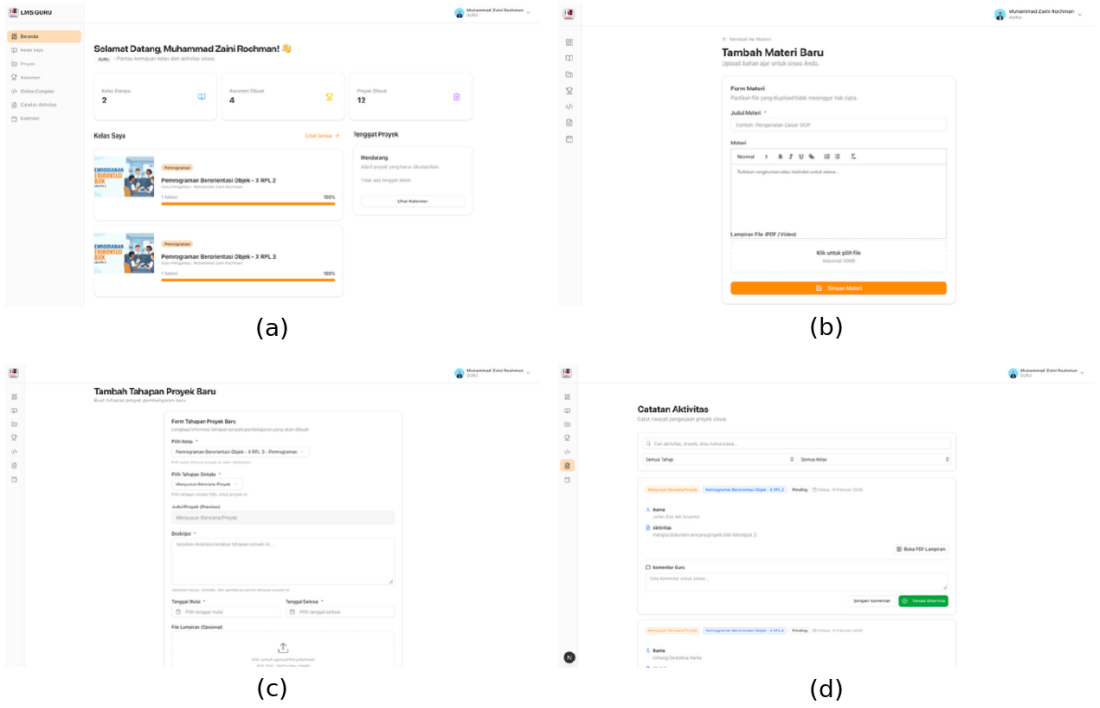
Gbr. 6 Activity Diagram Catatan Aktivitas (Logbook) Proyek

a) Antarmuka Guru: Gbr 7 menunjukkan implementasi antarmuka guru. Halaman *dashboard*/beranda (Gbr 7a) memuat statistik (*summary cards*) berupa total kelas yang diampu, asesmen yang dibuat, dan tahapan proyek yang telah dibuat. Ditampilkan juga kartu kelas dengan indikator progres (*progress bar*) penyelesaian materi, serta menu cepat (*shortcut*) untuk kelas dan *deadline* proyek. Halaman kelola materi (Gbr 7b) dilengkapi *rich text editor* untuk menulis ringkasan materi; materi pelajaran dapat diunggah, diedit, dan dihapus, serta dikelompokkan menjadi subbab secara otomatis. Halaman kelola tahapan proyek (Gbr 7c) berupa *form* yang memuat kelas, tahapan sintaks PjBL, deskripsi, lampiran *jobsheet*, serta tanggal mulai dan berakhir. Halaman *monitoring* aktivitas proyek (Gbr 7d) menampilkan seluruh data aktivitas proyek siswa pada kelas yang diampu, beserta label tahapan dan status (*pending*/diterima). Disediakan juga fitur pencarian serta kolom komentar guru.

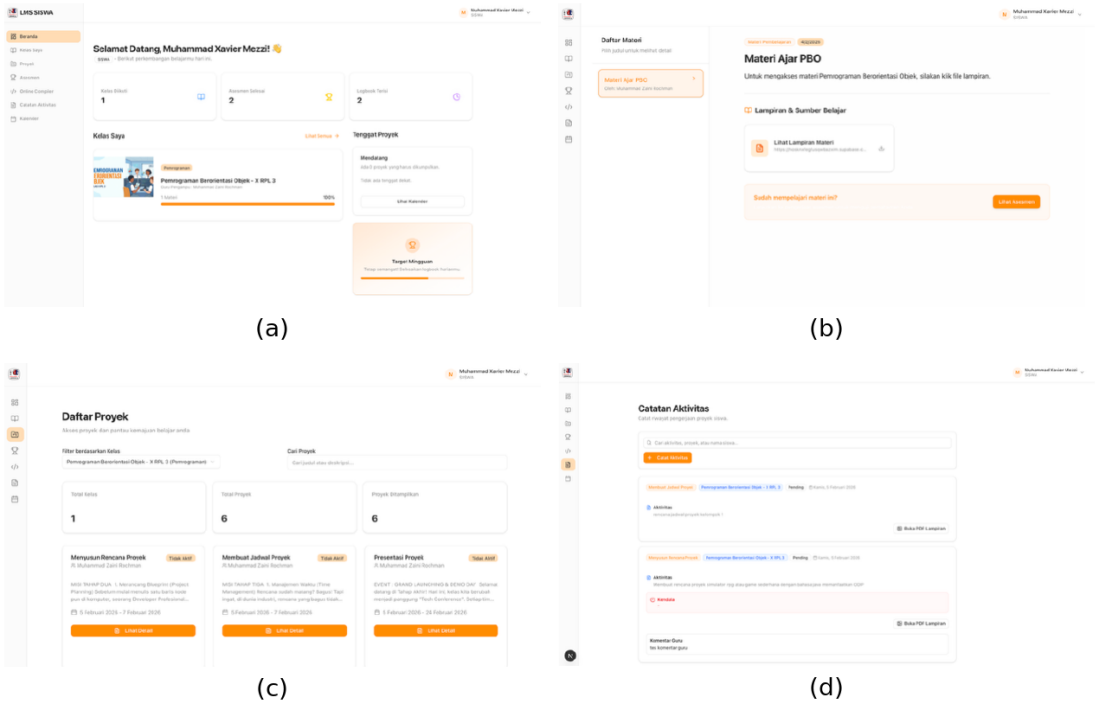
b) Antarmuka Siswa: Gbr 8 menunjukkan implementasi antarmuka siswa. Halaman *dashboard*/beranda (Gbr 8a) menampilkan statistik kelas yang diikuti, jumlah asesmen yang telah dikerjakan, dan *logbook* yang telah terisi. Disediakan juga widget "Target Mingguan" berupa *progress bar* pengisian *logbook*. Halaman materi (Gbr 8b) menampilkan daftar judul materi dalam bentuk akordeon (*accordion*) yang dapat diekspansi untuk melihat lampiran dan sumber belajar. Halaman proyek (Gbr 8c) menampilkan daftar tahapan proyek PjBL beserta label status pengerjaan setiap tahapan (*aktif*/*tidak aktif*) dan rentang tanggal pengerjaan. Halaman Catatan Aktivitas/*logbook* (Gambar 8d) menampilkan riwayat aktivitas yang telah dicatat siswa, beserta kendala dan komentar dari guru.

c) Antarmuka Admin: Gbr 9 menunjukkan implementasi antarmuka admin, yang hanya memuat dua menu navigasi utama, yaitu kelola pengguna dan kelola kelas. Halaman

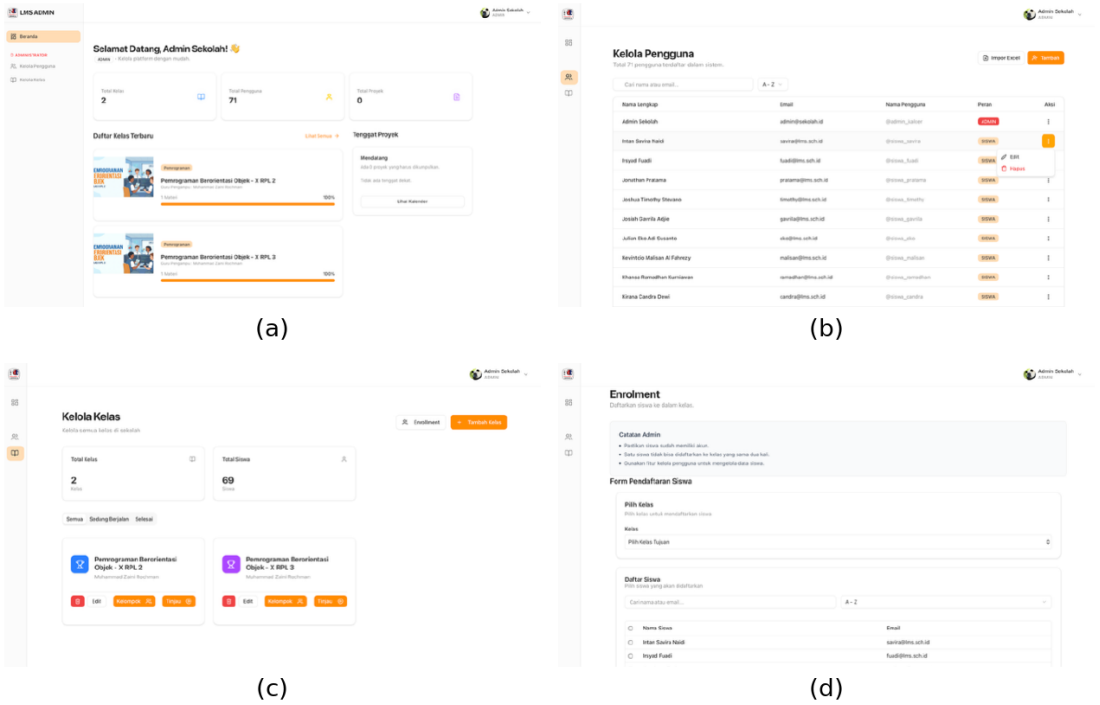
dashboard/beranda (Gbr 9a) menampilkan statistik total kelas, total pengguna, dan total proyek pada sistem. Halaman kelola pengguna (Gbr 9b) menampilkan seluruh pengguna LMS lengkap dengan label warna sesuai peran (admin/guru/siswa); admin dapat mengedit dan menghapus data pengguna, serta mengimpor pengguna secara massal melalui *file* Excel. Halaman kelola kelas (Gbr 9c) memungkinkan admin menambah, mengedit, dan menghapus kelas, menyaring kelas berdasarkan status (sedang berjalan/selesai), serta menentukan guru pengampu. Halaman *enrolment* (Gbr 9d) menyediakan kolom pencarian dan daftar *checkbox* siswa untuk mendaftarkan siswa ke dalam kelas tertentu.



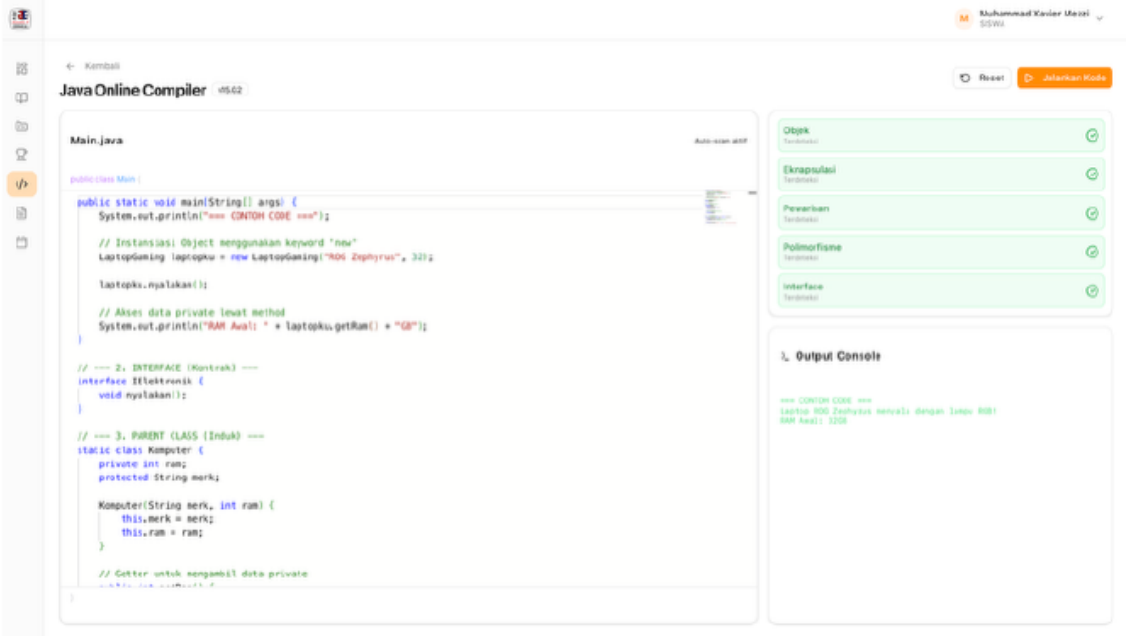
Gbr. 7 Implementasi Antarmuka Guru: (a) Dashboard, (b) Kelola Materi, (c) Kelola Tahapan Proyek, (d) Monitoring Aktivitas Proyek



Gbr. 8 Implementasi Antarmuka Siswa: (a) Dashboard, (b) Materi, (c) Proyek PjBL, (d) Catatan Aktivitas (Logbook) Proyek



Gbr. 9 Implementasi Antarmuka Admin: (a) Dashboard, (b) Kelola Pengguna, (c) Kelola Kelas, (d) Enrolment Siswa



Gbr. 10 Antarmuka Java Online Compiler dengan Panel OOP Analyzer dan Output Console

3) Fitur Unggulan: *Java Online Compiler* dan *OOP Analyzer*: fitur ini dirancang agar pengguna dapat menulis, mengompilasi, dan menjalankan kode program Java secara langsung melalui LMS di *browser*, sebagaimana ditunjukkan pada Gbr 10. Siswa tidak perlu menginstal *Integrated Development Environment* (IDE) tambahan seperti NetBeans atau Visual Studio Code. Antarmuka penulisan kode memanfaatkan *library* monaco-editor/react. *Library* ini merupakan mesin yang sama dengan yang digunakan oleh Visual Studio Code, lengkap dengan fitur *syntax highlighting* untuk bahasa Java, *auto-indentation*, dan *minimap*.

Untuk menyederhanakan proses belajar siswa tingkat pemula, sistem menggunakan pendekatan *wrapper* pada kode. Baris kode deklarasi kelas utama (public class Main {) disembunyikan secara statis sebagai *prefix* dan *suffix* di balik layar sebelum kode dikirim ke server. Hal ini memungkinkan siswa untuk langsung fokus menulis logika program dan pembuatan *class* tanpa perlu berulang kali menulis struktur wajib program.

Nilai tambah utama dari *compiler* ini adalah kehadiran fitur *OOP Analyzer*. Fitur ini berfungsi sebagai asisten otomatis yang memindai struktur kode siswa secara *real-time* setiap kali ada perubahan karakter di dalam *editor* (useEffect). Pemindaian ini dilakukan menggunakan metode *Regular Expression* pada fungsi analyzeCode(). Sistem akan mendeteksi penerapan lima konsep utama Pemrograman Berorientasi Objek (OOP), yaitu:

1) *Objek*, dideteksi dari penggunaan kata kunci new.
2) *Enkapsulasi*, dideteksi dari penggunaan *access modifier* seperti private atau protected.
3) *Pewarisan (Inheritance)*, dideteksi dari penggunaan kata kunci extends.
4) *Polimorfisme*, dideteksi dari keberadaan anotasi @Override atau pemanggilan metode induk melalui super.
5) *Interface*, dideteksi dari penggunaan kata kunci implements atau deklarasi *interface*.

Hasil deteksi ini divisualisasikan secara langsung pada panel di sebelah kanan *editor*, sebagaimana ditunjukkan pada gambar 10. Jika siswa berhasil menerapkan suatu konsep, indikator pada antarmuka akan langsung berubah warna menjadi hijau.

Untuk mengeksekusi kode Java yang telah ditulis, sistem memanfaatkan *Application Programming Interface* (API) pihak ketiga, yaitu *Piston API*. *Piston API* bertindak sebagai eksekutor kode program yang telah ditulis. Ketika pengguna menekan tombol "Jalankan Kode", fungsi asinkronus runCode() akan dipanggil. Sistem merakit kode utuh secara lengkap, lalu mengirimkannya dalam format JSON yang berisi parameter bahasa (Java) beserta versi kompilatornya ke *endpoint Piston API*.

Sistem kemudian menunggu respon dari server dan menampilkannya ke bentuk *Output Console* pada panel kanan bawah. Logika program telah dirancang untuk membedakan jenis respon yang diterima. Jika API mengembalikan kesalahan sintaks atau kegagalan kompilasi, teks akan ditampilkan dengan warna merah untuk menandakan pesan *error*. Sebaliknya, jika eksekusi berhasil, teks ditampilkan dengan warna hijau. Pendekatan ini melatih kemandirian siswa dalam melakukan *debugging* layaknya menggunakan *software* IDE profesional, tanpa memerlukan instalasi *compiler* tambahan pada perangkat siswa.

4) Modul Catatan Aktivitas (*Logbook*) Proyek: modul ini dirancang untuk mendukung pemantauan proyek pada model PjBL secara sistematis. Setiap kelompok siswa dapat mencatat aktivitas pengerjaan proyek pada tahapan sintaks PjBL yang sedang berjalan, meliputi tanggal, deskripsi aktivitas, kendala yang dihadapi, serta lampiran bukti pendukung berupa foto atau *file*. Guru dapat mengakses seluruh catatan aktivitas pada kelas yang diampu melalui halaman *monitoring* (Gambar 7d), memberikan komentar sebagai umpan balik, dan menandai catatan sebagai diterima. Modul ini menjawab permasalahan minimnya sistem pemantauan proyek yang ditemukan pada tahap analisis.

E. Tahap Evaluasi (evaluation)

Tahap evaluasi difokuskan pada pengujian fungsional dan non-fungsional sistem menggunakan metode *Blackbox Testing* yang dilakukan oleh sepuluh pengguna (admin, guru, dan siswa) sebagai calon pengguna. Pengujian dilakukan terhadap skenario *input* dan *output* sistem, dengan ringkasan hasil pengujian fungsional dan non-fungsional ditunjukkan pada Tabel 4 dan Tabel 5.

TABEL 4
HASIL UJI FUNGSIONAL BLACKBOX TESTING

Modul / Fitur	Skenario Pengujian (Test Case)	Hasil yang Diharapkan	Hasil
Auth	Login dengan kredensial benar sebagai admin, guru, dan siswa.	Sistem mengarahkan ke dashboard sesuai peran; menu admin tidak terlihat/tidak dapat diakses oleh guru maupun siswa.	Valid
Auth	Login dengan kredensial yang salah.	Sistem menolak akses dan menampilkan pesan error.	Valid *
Kelola kelas dan enrolmen (admin)	Admin menambah, mengedit, dan menghapus kelas, serta mendaftarkan (enrolment) siswa ke kelas.	Sistem menyimpan perubahan kelas; siswa yang didaftarkan dapat mengakses kelas tersebut.	Valid
Kelola pengguna (admin)	Admin menambahkan pengguna satu per satu maupun secara massal melalui file Excel, mengedit, dan menghapus pengguna.	Sistem menyimpan/menghapus data pengguna sesuai aksi yang dilakukan.	Valid
Proyek PjBL	Guru membuat tahapan proyek dan membentuk kelompok; siswa mengakses tahapan dan mengunggah pengumpulan proyek.	Data proyek tersimpan dan tampil pada halaman siswa; data pengumpulan proyek (tautan/file) tersimpan.	Valid
Proyek PjBL	Guru memberikan nilai pada proyek yang telah dikumpulkan.	Sistem menyimpan nilai dan siswa dapat melihat nilai yang diberikan oleh guru.	Valid
Catatan Aktivitas	Siswa mengisi form catat aktivitas (tanggal,	Data tersimpan dan langsung tampil pada	Valid

D. Tahap Implementasi (implementation)

Sistem yang telah selesai dikembangkan kemudian dilakukan *deployment* ke Vercel sehingga dapat diakses secara publik melalui *browser* dari berbagai perangkat dan sistem operasi. Uji coba terbatas dilakukan oleh peneliti bersama pengguna (admin, guru, dan siswa kelas X RPL SMK Antartika 2 Sidoarjo) untuk memastikan integrasi basis data Supabase berjalan stabil. Pengujian ini mencakup pengisian *logbook* proyek dan eksekusi kode pada fitur *Online Compiler* oleh banyak pengguna dalam waktu yang berdekatan. Hasil uji coba menunjukkan sistem dapat diakses dan menyimpan data tanpa kendala teknis yang berarti.

(Logbook) Proyek	aktivitas, kendala, lampiran) dan menyimpannya.	riwayat aktivitas proyek.	
Catatan Aktivitas (Logbook) Proyek	Guru mengakses catatan aktivitas siswa dan memberikan komentar.	Sistem menampilkan seluruh catatan aktivitas dan menyimpan komentar guru.	Valid
Online Compiler	Siswa menulis kode Java valid (contoh: System.out.println("Test");) dan menekan tombol Jalankan Kode.	Compiler memproses kode dan menampilkan output "Test" tanpa refresh halaman.	Valid
Online Compiler	Siswa menulis kode dengan sintaks error (kurang tanda titik koma).	Sistem menampilkan pesan error pada Output Console.	Valid
Materi	Guru mengunggah materi (file PDF) pada lampiran; siswa mengakses dan mengunduh materi.	File berhasil terunggah ke storage, tautan unduhan berfungsi dan file dapat dibuka.	Valid
Asesmen	Guru membuat soal asesmen dan menambahkan gambar pada beberapa soal; siswa mengerjakan dan menekan tombol submit.	Sistem menyimpan soal dan gambar, serta menghitung skor secara otomatis berdasarkan kunci jawaban.	Valid
Asesmen	Siswa mengeluarkan pointer mouse dari area asesmen saat mengerjakan.	Sistem menampilkan peringatan dan jumlah pelanggaran yang telah dilakukan.	Valid

** Sistem menolak akses sesuai rancangan; satu catatan perbaikan pada penyajian pesan kesalahan dibahas pada bagian Pembahasan.*

TABEL 5
HASIL UJI NON-FUNGSIONAL BLACKBOX TESTING

Modul / Fitur	Skenario Pengujian (Test Case)	Hasil yang Diharapkan	Hasil
Kinerja (performance)	Mengakses halaman utama LMS dan mengukur waktu muat (load time).	Halaman terbuka sempurna dalam waktu < 1000 ms (1 detik).	Valid
Kinerja (performance)	Mengunggah file dengan ukuran di atas batas (misal > 5 MB).	Sistem menolak unggahan dan memberikan peringatan.	Valid
Kompatibilitas	Membuka LMS menggunakan browser Google Chrome dan Microsoft Edge.	Tampilan (layout) tidak pecah dan fitur Online Compiler berjalan lancar di kedua browser.	Valid
Keamanan (security)	Mencoba mengakses halaman URL khusus admin (misal /admin/dashboard) tanpa login (sebagai guest).	Sistem secara otomatis redirect paksa ke halaman login (middleware protection).	Valid
Ketersediaan (availability)	Mengakses LMS pada jam sibuk maupun jam malam.	Server merespons dan sistem dapat diakses tanpa downtime.	Valid

F. Pembahasan

Hasil pengembangan menunjukkan bahwa arsitektur *full-stack* berbasis Next.js memungkinkan sisi *frontend* dan *backend* dibangun dalam satu basis kode (*codebase*). Hal ini mempercepat proses pengembangan dan mengurangi kompleksitas konfigurasi *server* yang terpisah. Selain itu, pemanfaatan Supabase sebagai *Backend-as-a-Service* menyediakan otentikasi, *Row Level Security*, dan *storage* tanpa perlu membangun layanan *backend* dari awal. Kebutuhan non-fungsional terkait keamanan (Tabel 2) terbukti valid pada pengujian *blackbox testing* (Tabel 5).

Pemanfaatan *Piston API* sebagai mesin eksekusi kode menjawab permasalahan keterbatasan perangkat siswa yang ditemukan pada tahap analisis. Proses kompilasi dipindahkan ke sisi *server*, sehingga siswa dapat menjalankan program Java hanya melalui *browser* tanpa instalasi *Integrated Development Environment* (IDE) seperti NetBeans atau Eclipse. Hal ini berbeda dengan LMS berbasis Moodle pada penelitian terdahulu yang telah mengintegrasikan PjBL namun tidak menyediakan fitur *compiler* bawaan [4], sehingga siswa pada penelitian tersebut tetap bergantung pada aplikasi eksternal untuk praktik pemrograman.

Algoritma *OOP Analyzer* dikembangkan menggunakan pendekatan *Regular Expression* karena ringan secara komputasi dan dapat dijalankan secara *real-time* di sisi *client* tanpa membebani *server*. Namun, pendekatan ini hanya mendeteksi pola sintaksis (leksikal) dan belum memvalidasi struktur kode secara semantik. Akibatnya, deteksi dapat menjadi kurang akurat pada penulisan kode dengan format tidak standar.

Secara keseluruhan, hasil *blackbox testing* pada Tabel 4 dan Tabel 5 yang mencatatkan tingkat kevalidan sebesar 99% menunjukkan bahwa arsitektur *full-stack* Next.js, integrasi Supabase, serta fitur *Online Compiler* dan *OOP Analyzer* yang dirancang pada tahap *design* dan *development* telah berfungsi sesuai rancangan.

Hasil pengujian *blackbox testing* pada Tabel 4 juga menunjukkan satu skenario pada modul otentikasi, yaitu pengujian *login* dengan kredensial yang salah, masih memerlukan penyesuaian pada penyajian pesan kesalahan. Meskipun demikian, sistem berhasil menolak akses tidak sah sesuai rancangan keamanan. Temuan ini menjadi catatan perbaikan pada pengembangan lanjutan dan tidak mengurangi tingkat kevalidan fungsionalitas sistem secara keseluruhan (99%).

IV. KESIMPULAN

Berdasarkan hasil dan pembahasan pada penelitian ini yang berjudul "Rancang Bangun Sistem LMS Berbasis Web dengan Online Compiler dan OOP Analyzer", dapat ditarik kesimpulan sebagai berikut.

1. Rancang bangun sistem LMS dilaksanakan melalui lima tahapan model pengembangan ADDIE, yaitu:
   a. Analisis (*Analyze*): menggali permasalahan pembelajaran Pemrograman Berorientasi Objek (OOP) pada kelas X RPL SMK Antartika 2 Sidoarjo, meliputi identifikasi pengguna serta kebutuhan fungsional dan non-fungsional sistem.
   b. Desain (*Design*): merancang arsitektur sistem *full-stack* berbasis Next.js. Perancangan disajikan dalam bentuk UML berupa *use case diagram* dan *activity diagram*, sedangkan perancangan basis data berupa *class diagram*.
   c. Pengembangan (*Development*): merealisasikan rancangan menjadi LMS menggunakan Next.js, Tailwind CSS, dan Supabase (PostgreSQL dengan *Row Level Security*), dilengkapi integrasi *Piston API* untuk eksekusi kode dan algoritma *OOP Analyzer* berbasis *Regex*.
   d. Implementasi (*Implementation*): melakukan *deployment* sistem ke Vercel dan uji coba terbatas bersama pengguna (admin, guru, dan siswa) kelas X RPL SMK Antartika 2 Sidoarjo.
   e. Evaluasi (*Evaluation*): melakukan pengujian fungsional dan non-fungsional melalui *blackbox testing* oleh sepuluh pengguna yang menunjukkan sistem berjalan valid (99%).
2. Fitur Online Compiler yang dilengkapi Monaco Code Editor dan OOP Analyzer berbasis Regex berhasil mendeteksi penerapan lima pilar OOP (objek, enkapsulasi, pewarisan, polimorfisme, dan interface) secara real-time. Hal ini ditunjukkan oleh hasil blackbox testing pada Tabel 4 yang menunjukkan modul Online Compiler dan Catatan Aktivitas (logbook) Proyek berjalan valid sesuai rancangan.

Dengan demikian, LMS yang dikembangkan dinyatakan valid dan layak digunakan sebagai media pembelajaran praktik pemrograman berbasis *website* pada pendidikan vokasi, khususnya pada mata pelajaran Pemrograman Berorientasi Objek.

V. SARAN

Berdasarkan hasil penelitian yang telah dilakukan, berikut saran yang dapat diberikan kepada pihak sekolah, peneliti selanjutnya, serta pengembang LMS.

a. *Penyempurnaan Algoritma OOP Analyzer*
   Bagi peneliti selanjutnya, disarankan untuk menyempurnakan algoritma *OOP Analyzer* menggunakan teknologi *Abstract Syntax Tree* (AST) *parser*. Fitur pendeteksi konsep OOP pada penelitian ini masih dibangun menggunakan algoritma *Regular Expression* (Regex) yang memiliki keterbatasan apabila kode ditulis dengan format tidak standar. Dengan *AST parser*, deteksi konsep OOP menjadi lebih presisi dan dapat memvalidasi struktur kode secara semantik.

B. *Peningkatan Kesiapan Infrastruktur*
   Bagi pihak sekolah, disarankan untuk terus mengoptimalkan stabilitas jaringan internet di lingkungan sekolah. LMS ini beroperasi secara *cloud-based*, khususnya pada eksekusi *Online Compiler* melalui *Piston API* yang membutuhkan koneksi internet stabil secara *real-time*, terutama saat digunakan secara serentak oleh banyak pengguna.

C. *Ekspansi Bahasa Pemrograman*
   Bagi peneliti selanjutnya, disarankan untuk mengekspansi LMS tidak hanya untuk bahasa pemrograman Java, tetapi juga mata pelajaran kejuruan lain yang membutuhkan eksekusi kode, seperti Pemrograman *Web* (PHP/JavaScript) atau Dasar-Dasar Pemrograman (C++/Python). *Piston API* mendukung lebih dari satu bahasa pemrograman pada *endpoint* yang sama, sehingga ekspansi ini tidak memerlukan perubahan arsitektur secara mendasar.

D. *Optimasi Antarmuka untuk Perangkat Mobile*
   Bagi peneliti selanjutnya, disarankan untuk mengoptimalkan antarmuka *Online Compiler* untuk perangkat layar sentuh, mengingat sebagian besar siswa mengakses LMS melalui *smartphone*. Misalnya, dengan menyediakan tombol pintas untuk karakter simbol pemrograman yang sering digunakan (seperti kurung kurawal, titik koma, dan tanda kutip) guna meningkatkan efisiensi penulisan kode pada perangkat *mobile*.

E. *Pengujian Usability Menggunakan System Usability Scale (SUS)*
   Bagi peneliti selanjutnya, disarankan untuk melakukan pengujian *usability* menggunakan *System Usability Scale* (SUS) yang melibatkan guru dan siswa sebagai responden, sebagaimana diterapkan pada penelitian LMS sejenis [7]. Penelitian ini masih berfokus pada *blackbox testing* dan belum mengukur *usability* dari sisi pengguna akhir. Pengujian SUS dapat melengkapi hasil tersebut dengan ukuran kepuasan dan kemudahan penggunaan sistem secara kuantitatif.

REFERENSI

[1] A. R. Ahmetya, I. Setyaningrum, and O. Tanaya, "Era Baru Ketenagakerjaan: Fleksibilitas Pekerja Digital Pada Era Revolusi Industri 4.0," *Jurnal Ilmiah Universitas Muhammadiyah Buton*, vol. 9, no. 4, pp. 1001–1015, Nov. 2023, doi: 10.35326/pencerah.v8i4.4495.

[2] G. Gayatri, I. G. N. M. Jaya, and V. M. Rumata, "The Indonesian Digital Workforce Gaps in 2021–2025," *Sustainability (Switzerland)*, vol. 15, no. 1, Jan. 2023, doi: 10.3390/su15010754.

[3] D. J. Jaya *et al.*, "Komparasi Keterserapan Kerja Lulusan SMK Kompetensi Keahlian TGB di DIY," *Indonesian Journal of Civil Engineering Education*, vol. 10, no. 1, p. 38, Oct. 2024, doi: 10.20961/ijcee.v10i1.94394.

[4] R. S. Munir and I. Yuliana, "Implementasi Learning Management System Berbasis Moodle dengan Project Based Learning pada Mata Pelajaran TIK Siswa Sekolah Menengah Kejuruan," *JIIP - Jurnal Ilmiah Ilmu Pendidikan*, vol. 8, no. 3, pp. 2542–2551, Mar. 2025, doi: 10.54371/jiip.v8i3.7212.

[5] N. N. Wardah, A. M. Yunita, and A. Yusta, "Pemanfaatan Aplikasi Learning Management System untuk Kebutuhan Pembelajaran Daring Melalui Kegiatan Mini Workshop bagi Guru Tingkat Sekolah Menengah," *Jurnal Dhamabakti Nagri*, vol. 1, no. 1, Mar. 2023, doi: 10.32524/jusitik.v2i2.551.

[6] Y. Nuraeni, W. A. Kurnia, V. N. Handayani, K. C. Kirana, S. N. Syifa, and Z. P. Dinata, "Strategi Implementasi PJBL untuk Meningkatkan Motivasi dan Hasil Belajar Siswa," *Jurnal Nakula: Pusat Ilmu Pendidikan,*

Bahasa dan Ilmu Sosial, vol. 3, no. 6, pp. 193–199, Jul. 2025, doi: 10.61132/nakula.v3i6.2330.

[7] R. Sugiarto and A. Musyafa, "Learning Management System (LMS) pada SMK 1 Barunawati Jakarta," *Jurnal Teknologi Informatika dan Komputer*, vol. 10, no. 2, pp. 768–789, Sep. 2024, doi: 10.37012/jtik.v10i2.2422.

[8] M. I. K. Fannan and M. D. E. Susanti, "Rancang Bangun Jismart Berbasis Website dengan Model PjBL untuk Meningkatkan Kompetensi Dasar Pemrograman dan Algoritma," *IT-Edu: Jurnal Information Technology & Education*, vol. 11, no. 1, pp. 24–35, Jan. 2026, doi: 10.26740/it-edu.v11i01.72133.

[9] R. M. Branch, *Instructional Design: The ADDIE Approach*. New York, NY: Springer, 2009.