PEMBANGUNAN SISTEM INFORMASI MANAJEMEN DATA LEMBAGA PENELITIAN PENGABDIAN MASYARAKAT MENGGUNAKAN PERSONAL EXTREME PROGRAMMING PADA STAI ALI BIN ABI THALIB SURABAYA

DEVELOPMENT OF DATA MANAGEMENT INFORMATION SYSTEM FOR COMMUNITY SERVICE RESEARCH INSTITUTE USING PERSONAL EXTREME PROGRAMMING AT STAI ALI BIN ABI THALIB SURABAYA

Proposal Tugas Akhir

Mata Kuliah Metode Penelitian Dan Penyusunan Karya Ilmiah

(BBK4AAB2)

Disusun oleh:

MUHAMMAD RIZKY QOIRUL HUDA

1204220096

PROGRAM STUDI SARJANA SISTEM INFORMASI

DIREKTORAT KAMPUS SURABAYA

UNIVERSITAS TELKOM

SURABAYA

2025

LEMBAR PENGESAHAN

PROPOSAL TUGAS AKHIR

PEMBANGUNAN SISTEM INFORMASI MANAJEMEN DATA LEMBAGA PENELITIAN PENGABDIAN MASYARAKAT MENGGUNAKAN PERSONAL EXTREME PROGRAMMING PADA STAI ALI BIN ABI THALIB SURABAYA

Proposal ini diajukan sebagai usulan pembuatan Tugas Akhir
pada Program Studi Sarjana Sistem Informasi
Direktorat Kampus Surabaya
Universitas Telkom

Disusun oleh:
MUHAMMAD RIZKY QOIRUL HUDA
1204220096

Surabaya, 12 Desember 2025
Menyetujui,

Pembimbing I,	**Pembimbing II**,
Adzanil Rachmadhi Putra, S.Kom., M.Kom.	**Purnama Anaking, S.Kom., M.Kom.**
NIP. 22940040	NIP. 21870007

Kaprodi Sistem Informasi,

Dr. Berlian Rahmy Lidiawaty, S.ST., M.MT.
NIP. 22940041

LEMBAR PERNYATAAN ORISINALITAS

FOTO FORMAL 4X6 BERWARNA

Nama : Muhammad Rizky Qoirul Huda
NIM : 1204220096
Alamat : Kedungturi Permai 2 CC 22

No. Telp : 089666819842
Email : muhammadrizkyqh@gmail.com

Menyatakan bahwa Tugas Akhir ini merupakan karya orisinal saya sendiri. Atas pernyataan ini, saya siap menanggung risiko atau sanksi yang dijatuhkan kepada saya apabila kemudian ditemukan adanya pelanggaran terhadap kejujuran akademik atau etika keilmuan dalam karya ini, atau ditemukan bukti yang menunjukkan ketidakaslian karya ini.

Surabaya, 12 Desember 2025

Muhammad Rizky Qoirul Huda

ABSTRAK

Lembaga Penelitian dan Pengabdian Masyarakat (LPPM) STAI Ali bin Abi Thalib Surabaya menghadapi tantangan dalam pengelolaan data penelitian dan pengabdian masyarakat karena masih menggunakan sistem manual, yang menyebabkan inefisiensi operasional, kesulitan dalam rekapitulasi data, dan pelaporan yang rentan terhadap kesalahan. Penelitian ini bertujuan untuk mengembangkan sistem informasi manajemen data LPPM berbasis web menggunakan metodologi Personal Extreme Programming (PXP) untuk mengotomatisasi proses pengelolaan penelitian dan pengabdian masyarakat. Sistem ini dirancang untuk mengelola skema penelitian (Dasar, Terapan, Pengembangan, dan Kolaboratif), mendukung pencairan dana bertahap (50%-25%-25%), serta mengintegrasikan data dengan Lembaga Penjamin Mutu (LPM) untuk pelaporan real-time. Dengan pendekatan PXP, sistem dikembangkan melalui iterasi cepat untuk memastikan kualitas dan kesesuaian dengan kebutuhan pengguna. Hasil penelitian ini diharapkan meningkatkan efisiensi operasional, akurasi data, dan mendukung pencapaian target institusi yaitu 25 penelitian, 25 pengabdian masyarakat, dan 40 publikasi per tahun.

Kata Kunci: Sistem Informasi Manajemen, LPPM, Personal Extreme Programming, STAI Ali bin Abi Thalib, Integrasi Data.

ABSTRACT

The Institute of Research and Community Service (LPPM) at STAI Ali bin Abi Thalib Surabaya faces challenges in managing research and community service data due to reliance on manual systems, leading to operational inefficiencies, difficulties in data aggregation, and error-prone reporting. This study aims to develop a web-based LPPM data management information system using the Personal Extreme Programming (PXP) methodology to automate the management processes of research and community service activities. The system is designed to handle various research schemes (Basic, Applied, Development, and Collaborative), support phased funding disbursement (50%-25%-25%), and integrate data with the Quality Assurance Institute (LPM) for real-time reporting. Utilizing the PXP approach, the system is developed through rapid iterations to ensure quality and alignment with user needs. The results of this study are expected to enhance operational efficiency, data accuracy, and support the institution's targets of 25 research projects, 25 community service activities, and 40 publications annually.

Keywords: *Management Information System, LPPM, Personal Extreme Programming, STAI Ali bin Abi Thalib, Data Integration.*

KATA PENGANTAR

Puji syukur ke hadirat Allah SWT atas segala rahmat dan hidayah-Nya sehingga penyusunan proposal tugas akhir ini dapat diselesaikan dengan baik. Proposal ini disusun sebagai bagian dari persyaratan akademik untuk memenuhi Mata Kuliah Metode Penelitian dan Penyusunan Karya Ilmiah pada Program Studi Sarjana Sistem Informasi, Universitas Telkom, Direktorat Kampus Surabaya.

Penelitian ini bertujuan untuk mengembangkan sistem informasi manajemen data Lembaga Penelitian dan Pengabdian Masyarakat (LPPM) di STAI Ali bin Abi Thalib Surabaya menggunakan metodologi Personal Extreme Programming. Sistem ini diharapkan dapat mengatasi permasalahan pengelolaan data penelitian dan pengabdian masyarakat yang masih dilakukan secara manual, sehingga meningkatkan efisiensi operasional dan mendukung pencapaian target institusi.

Ucapan terima kasih disampaikan kepada berbagai pihak yang telah mendukung penyusunan proposal ini, antara lain:

1. Bapak Adzanil Rachmadhi Putra, S.Kom., M.Kom., selaku Pembimbing I, yang telah memberikan bimbingan dan arahan yang sangat berharga.
2. Bapak Purnama Anaking, S.Kom., M.Kom., selaku Pembimbing II, atas masukan dan dukungannya selama proses penyusunan.
3. Ibu Dr. Berlian Rahmy Lidiawaty, S.ST., M.MT. Selaku Kepala Program Studi Sistem Informasi, atas dukungan administratif dan akademik.
4. Seluruh staf LPPM dan LPM STAI Ali bin Abi Thalib Surabaya yang telah memberikan data dan informasi penting untuk penelitian ini.
5.
6. Keluarga, teman, dan rekan-rekan mahasiswa yang senantiasa memberikan semangat dan doa.

Penulis menyadari bahwa proposal ini masih jauh dari sempurna. Oleh karena itu, kritik dan saran yang konstruktif sangat diharapkan untuk penyempurnaan lebih lanjut. Semoga proposal ini dapat memberikan manfaat bagi pengembangan sistem informasi di STAI Ali bin Abi Thalib Surabaya serta menjadi kontribusi nyata bagi dunia akademik dan masyarakat.

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR LAMPIRAN

DAFTAR ISTILAH

Sistem Informasi Manajemen (SIM)	Sistem yang dirancang untuk mengumpulkan, menyaring, memproses, menciptakan, dan mendistribusikan data guna mendukung pengambilan keputusan dalam suatu organisasi.
Lembaga Penelitian dan Pengabdian Masyarakat (LPPM)	Unit strategis di perguruan tinggi yang bertugas mengelola, memfasilitasi, dan mengembangkan kegiatan penelitian dan pengabdian kepada masyarakat.
Lembaga Penjamin Mutu (LPM)	Unit di perguruan tinggi yang bertanggung jawab atas penjaminan mutu akademik dan pelaporan sesuai standar akreditasi.
Personal Extreme Programming (PXP)	Adaptasi metodologi Extreme Programming untuk pengembangan perangkat lunak oleh individu atau tim kecil, dengan fokus pada iterasi cepat, test-driven development, dan fleksibilitas.
Extreme Programming (XP)	Metodologi pengembangan perangkat lunak agile yang menekankan pada iterasi pendek, umpan balik cepat, dan praktik seperti test-driven development dan refactoring.
Test-Driven Development (TDD)	Pendekatan pengembangan perangkat lunak di mana test case ditulis sebelum kode implementasi untuk memastikan fungsionalitas yang sesuai.
Refactoring	Proses perbaikan struktur kode secara berkelanjutan tanpa mengubah fungsionalitas untuk meningkatkan kualitas dan kemudahan perawatan.
User Stories	Deskripsi singkat tentang kebutuhan pengguna dalam format "Sebagai [role], saya ingin [functionality], sehingga [benefit]" untuk mengarahkan pengembangan sistem.
RESTful API	Antarmuka pemrograman aplikasi berbasis Representational State Transfer yang memungkinkan komunikasi antar sistem menggunakan protokol HTTP dengan format JSON.
Three-Tier Architecture	Arsitektur sistem yang memisahkan lapisan presentasi (presentation tier), logika bisnis (application tier), dan penyimpanan data (data tier) untuk meningkatkan skalabilitas dan pemeliharaan.
Laravel Framework	Framework PHP open-source yang digunakan untuk pengembangan aplikasi web dengan pola arsitektur Model-View-Controller (MVC).
Eloquent ORM	Object-Relational Mapping pada Laravel yang memungkinkan interaksi dengan database menggunakan sintaks PHP yang intuitif.
Blade Templating Engine	Mesin templating Laravel yang digunakan untuk membuat antarmuka pengguna yang dinamis dan reusable.

Model-View-Controller (MVC)	Pola arsitektur yang memisahkan logika data (Model), antarmuka pengguna (View), dan logika bisnis (Controller) untuk pengembangan aplikasi yang terstruktur.
Skema Penelitian	Kategori penelitian yang mencakup Penelitian Dasar, Penelitian Terapan, Penelitian Pengembangan, dan Penelitian Kolaboratif, sesuai dengan pedoman institusi.
Pencairan Dana Bertahap	Sistem pengelolaan dana penelitian yang dibagi menjadi tahap 50%-25%-25% sesuai dengan kemajuan proyek.
Audit Trail	Jejak dokumentasi yang mencatat setiap perubahan atau aktivitas dalam sistem untuk memastikan akuntabilitas dan transparansi.
Dashboard Analytics	Antarmuka visual yang menampilkan data kinerja (KPI) seperti jumlah penelitian, pengabdian masyarakat, dan publikasi secara real-time.
Sistem Penjaminan Mutu	Sistem yang digunakan untuk memastikan kualitas proses dan hasil akademik sesuai dengan standar yang ditetapkan, seperti SPMI dari Kemendikbud.
Responsive Design	Pendekatan desain antarmuka pengguna yang memastikan sistem dapat diakses dengan baik di berbagai perangkat dan ukuran layar.

BAB I
PENDAHULUAN

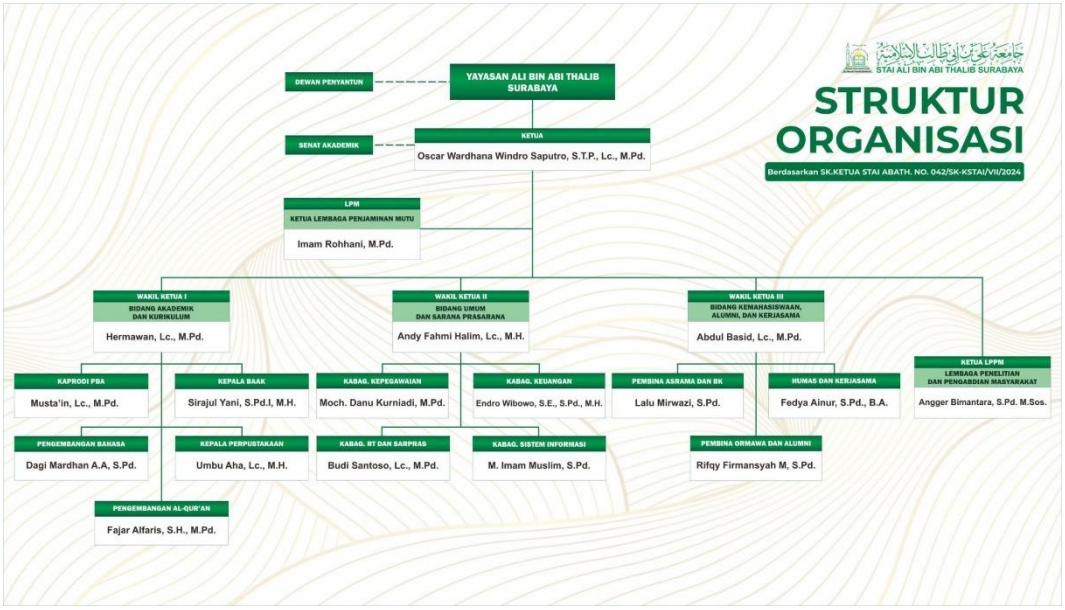
1.1 Latar Belakang

Lembaga Penelitian dan Pengabdian Masyarakat (LPPM) merupakan salah satu unit strategis dalam struktur organisasi perguruan tinggi yang memiliki peran sentral dalam mengelola, memfasilitasi, dan mengembangkan kegiatan penelitian serta pengabdian kepada masyarakat. Berdasarkan Undang-Undang Nomor 12 Tahun 2012 tentang Pendidikan Tinggi, perguruan tinggi wajib menyelenggarakan Tri Dharma Perguruan Tinggi yang mencakup pendidikan, penelitian, dan pengabdian kepada masyarakat. Keberadaan LPPM menjadi krusial dalam mengimplementasikan amanat tersebut, khususnya dalam menghasilkan penelitian berkualitas yang dapat berkontribusi pada pengembangan ilmu pengetahuan dan peningkatan daya saing institusi. Dalam era digital dan persaingan global yang semakin ketat, kebutuhan akan sistem informasi yang terintegrasi dan efisien menjadi sangat penting untuk mendukung kelancaran operasional dan peningkatan mutu layanan LPPM.

Transformasi digital dalam pengelolaan penelitian institusional tidak hanya meningkatkan efisiensi administratif, tetapi juga mendukung transparansi dan akuntabilitas dalam setiap tahapan kegiatan penelitian. Sistem informasi manajemen yang baik memungkinkan LPPM untuk mengotomatisasi proses-proses manual yang selama ini memakan waktu dan rentan terhadap kesalahan, seperti pengajuan proposal, review dan penilaian, monitoring pelaksanaan penelitian, pencairan dana bertahap, hingga pelaporan dan diseminasi hasil penelitian. Penelitian oleh menunjukkan bahwa implementasi sistem informasi pada lembaga penelitian dapat meningkatkan efisiensi operasional hingga 40% dan mengurangi tingkat kesalahan data hingga 30%. Hal ini menegaskan bahwa sistem informasi bukan lagi pilihan, melainkan kebutuhan mendasar bagi perguruan tinggi yang ingin meningkatkan kualitas dan produktivitas penelitian.

Sekolah Tinggi Agama Islam (STAI) Ali bin Abi Thalib Surabaya sebagai institusi pendidikan tinggi Islam yang sedang berkembang, memiliki komitmen kuat untuk meningkatkan kualitas penelitian dan pengabdian masyarakat sesuai dengan visi institusi untuk menjadi universitas berlandaskan paham Ahlus Sunnah wal Jamaah yang unggul dalam Agama Islam, Pendidikan, dan Sosial Humaniora di Indonesia pada tahun 2037. Saat ini, STAI Ali bin Abi Thalib memiliki [X] dosen tetap yang tersebar di [X] program studi, dengan capaian penelitian dalam tiga tahun terakhir (2022-2024) mencapai rata-rata [X] penelitian, [X] pengabdian masyarakat, dan [X] publikasi per tahun. Sesuai dengan Buku Panduan Penelitian STAI Ali bin Abi Thalib Surabaya tahun 2024, institusi mengelola tiga skema penelitian yaitu Penelitian Dasar,

Penelitian Terapan, dan Penelitian Pengembangan, dengan sistem pencairan dana bertahap sebesar 50%, 25%, dan 25% sesuai tahap penyelesaian penelitian, serta melibatkan tiga jenis seminar yaitu Seminar Proposal, Seminar Internal, dan Seminar Publik sebagai bagian dari mekanisme quality assurance.

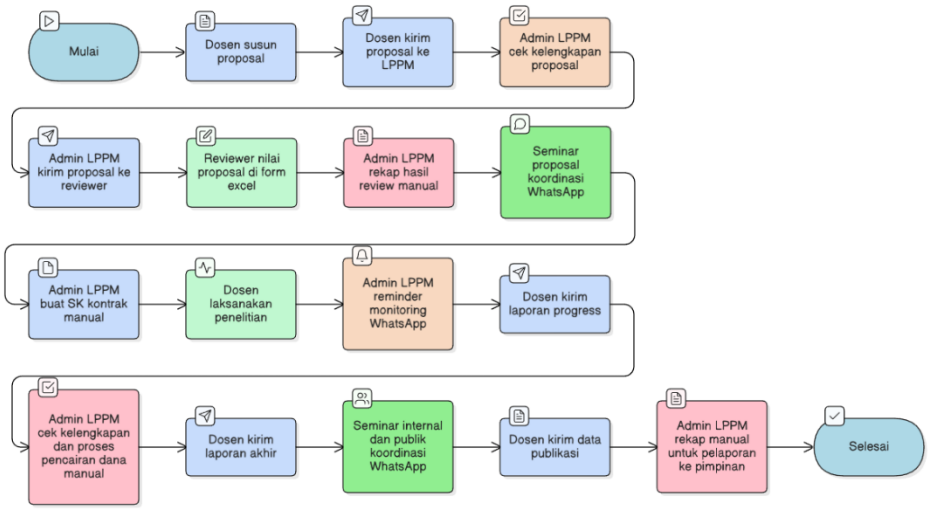


Berdasarkan hasil wawancara mendalam dengan pihak LPPM STAI Ali bin Abi Thalib pada Januari 2025, kondisi aktual menunjukkan bahwa institusi saat ini belum memiliki sistem informasi terintegrasi khusus untuk pengelolaan penelitian dan pengabdian masyarakat. Seluruh proses masih dilakukan secara manual dengan memanfaatkan dokumentasi berbasis file digital yang tersebar dalam berbagai format (Word, Excel, PDF) dan komunikasi melalui email serta aplikasi pesan instan. Kondisi ini menimbulkan beberapa permasalahan operasional yang menghambat produktivitas dan kualitas pengelolaan LPPM. Pertama, dokumentasi kegiatan penelitian dan pengabdian masyarakat tersebar dalam berbagai folder dan platform penyimpanan yang tidak terstruktur, sehingga menyulitkan proses pencarian data dan rekapitulasi. Waktu yang dibutuhkan untuk melakukan rekapitulasi data penelitian secara manual diperkirakan mencapai [2-3] hari kerja dengan tingkat kesalahan mencapai [10-15%] karena duplikasi dan inkonsistensi format dokumen.

Kedua, proses pengajuan proposal penelitian masih dilakukan secara konvensional dengan pengumpulan dokumen melalui email atau penyerahan hardcopy ke sekretariat LPPM. Mekanisme ini tidak hanya memakan waktu, tetapi juga rentan terhadap kehilangan dokumen dan keterlambatan proses approval. Berdasarkan data internal LPPM, terdapat [5-10] kasus per tahun dimana proposal mengalami keterlambatan proses review karena dokumen tidak tersampaikan dengan baik atau terjadi miskomunikasi antara peneliti dan reviewer. Ketiga, monitoring pelaksanaan penelitian yang seharusnya dilakukan

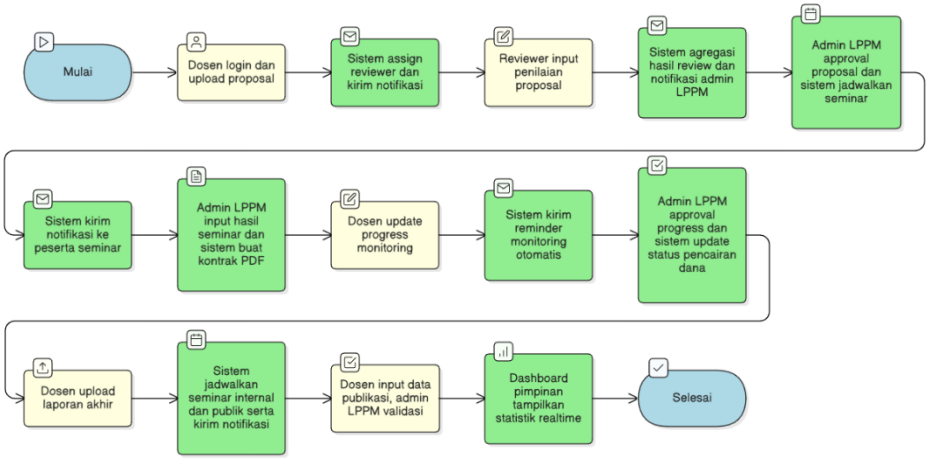
dalam dua tahap (monitoring tahap pertama untuk Bab I-III dan monitoring tahap kedua untuk Bab IV) sulit dilakukan secara real-time karena tidak ada platform terpusat untuk melacak progress penelitian. Admin LPPM harus menghubungi peneliti satu per satu melalui WhatsApp atau email untuk meminta update perkembangan, yang memakan waktu [1-2] minggu untuk satu siklus monitoring lengkap.

Ketiga, sistem pencairan dana bertahap (50%-25%-25%) yang menjadi mekanisme standar sesuai panduan penelitian STAI Ali bin Abi Thalib, belum memiliki sistem tracking yang jelas. Admin LPPM kesulitan memantau penelitian mana yang sudah menerima pencairan tahap pertama, kedua, atau ketiga, serta kapan jadwal pencairan berikutnya. Hal ini berpotensi menyebabkan keterlambatan pencairan yang dapat menghambat pelaksanaan penelitian. Kelima, pelaporan publikasi dan luaran penelitian masih dilakukan secara manual dengan dosen mengirimkan data publikasi melalui email atau mengisi form Excel. Tidak adanya database terpusat menyebabkan kesulitan dalam memonitor capaian publikasi institusi secara akurat dan real-time. Keenam, manajemen tiga jenis seminar (Proposal, Internal, dan Publik) yang menjadi bagian penting dari siklus penelitian di STAI Ali bin Abi Thalib, belum memiliki sistem penjadwalan dan dokumentasi yang terstruktur. Koordinasi seminar masih dilakukan melalui grup WhatsApp dan pengumuman manual, yang rentan terhadap kesalahan jadwal dan ketidakhadiran peserta.



Untuk mengatasi permasalahan tersebut, diperlukan pengembangan sistem informasi manajemen data LPPM yang dapat mengotomatisasi seluruh proses penelitian mulai dari pengajuan proposal, review dan penilaian, penandatanganan kontrak, monitoring pelaksanaan, pencairan dana bertahap, manajemen seminar, hingga pelaporan publikasi dan luaran penelitian. Sistem ini diharapkan dapat menyediakan platform terpusat yang memudahkan Admin

LPPM, Dosen/Peneliti, Reviewer, dan Pimpinan untuk berkolaborasi secara efisien dengan dukungan dashboard monitoring, notifikasi otomatis, dan audit trail untuk memastikan akuntabilitas setiap tahapan penelitian. Pengembangan sistem informasi yang efektif memerlukan pemilihan metodologi pengembangan perangkat lunak yang sesuai dengan konteks dan keterbatasan sumber daya yang ada.



Dalam konteks tugas akhir yang dikerjakan oleh mahasiswa secara individu dengan keterbatasan waktu dan sumber daya, metodologi Personal Extreme Programming (PXP) menjadi pilihan yang tepat. PXP merupakan adaptasi dari metodologi Extreme Programming (XP) yang dirancang khusus untuk pengembangan perangkat lunak oleh individu atau tim yang sangat kecil . PXP mempertahankan nilai-nilai inti XP seperti iterasi pendek, umpan balik cepat, test-driven development (TDD), dan refactoring berkelanjutan, namun disederhanakan untuk konteks pengembang tunggal tanpa mengurangi kualitas hasil akhir . Penelitian oleh menunjukkan bahwa PXP efektif digunakan dalam pengembangan sistem informasi skala menengah dengan kompleksitas sedang-tinggi, dengan tingkat keberhasilan proyek mencapai 85% dibandingkan metodologi tradisional seperti Waterfall yang hanya mencapai 60% untuk konteks individu.

Keunggulan PXP dalam penelitian ini terletak pada kemampuannya untuk melakukan iterasi cepat dengan melibatkan stakeholder (Admin LPPM, Dosen, dan Pimpinan STAI Ali bin Abi Thalib) dalam setiap tahap pengembangan melalui demo dan review berkala. Pendekatan ini memastikan bahwa sistem yang dikembangkan sesuai dengan kebutuhan nyata pengguna dan dapat segera disesuaikan jika terdapat perubahan requirement. Praktik test-driven development (TDD) dalam PXP juga mendukung quality assurance dengan memastikan setiap fungsi sistem diuji secara otomatis sebelum diimplementasikan, sehingga mengurangi bug dan meningkatkan

maintainability kode . Selain itu, PXP mendukung continuous integration melalui tools seperti GitHub Actions yang memungkinkan deployment otomatis ke server produksi setiap kali terdapat perubahan kode yang lolos pengujian, sehingga mempercepat siklus pengembangan dan meningkatkan produktivitas .

Pemilihan teknologi yang tepat juga menjadi faktor krusial dalam keberhasilan pengembangan sistem informasi manajemen LPPM. Dalam penelitian ini, teknologi yang digunakan adalah Next.js sebagai full-stack framework berbasis React, Prisma sebagai Object-Relational Mapping (ORM) untuk manajemen database, TypeScript sebagai bahasa pemrograman yang mendukung type safety, dan MySQL sebagai sistem manajemen basis data relasional. Next.js dipilih karena kemampuannya dalam mendukung Server-Side Rendering (SSR) dan Static Site Generation (SSG) yang meningkatkan performa aplikasi dan Search Engine Optimization (SEO), serta menyediakan API Routes yang memungkinkan pengembangan backend dan frontend dalam satu framework yang kohesif . Penelitian oleh menunjukkan bahwa aplikasi web yang dibangun dengan Next.js memiliki performa 30-40% lebih cepat dibandingkan aplikasi berbasis Client-Side Rendering (CSR) tradisional, serta developer experience yang superior dengan fitur hot reload dan modern tooling.

Prisma dipilih sebagai ORM karena menyediakan type-safe database access yang terintegrasi dengan TypeScript, sehingga mengurangi kesalahan runtime dan meningkatkan produktivitas developer melalui auto-completion dan error detection pada tahap development . Prisma juga menyediakan Prisma Migrate untuk mengelola schema database dengan versioning yang jelas, sehingga memudahkan evolusi struktur database seiring dengan perubahan requirement sistem. TypeScript dipilih sebagai bahasa pemrograman karena kemampuannya dalam mendeteksi kesalahan tipe data pada tahap kompilasi, bukan pada tahap runtime, sehingga meningkatkan code quality dan mengurangi bug pada aplikasi kompleks . Kombinasi Next.js, Prisma, dan TypeScript menciptakan ekosistem full-stack development yang modern, type-safe dari database hingga user interface, dan mendukung best practices seperti separation of concerns dan maintainability code.

Sistem informasi manajemen LPPM yang akan dikembangkan dalam penelitian ini dirancang untuk mengelola seluruh siklus penelitian sesuai dengan Buku Panduan Penelitian STAI Ali bin Abi Thalib Surabaya tahun 2024, mencakup fitur-fitur utama sebagai berikut: (1) Manajemen user dengan role-based access control untuk Admin LPPM, Dosen/Peneliti, Reviewer, dan Pimpinan; (2) Modul pengajuan dan review proposal penelitian dengan workflow approval dan penilaian sesuai kriteria substansi dan administratif; (3) Manajemen kontrak penelitian yang di-generate otomatis setelah proposal disetujui; (4) Modul monitoring pelaksanaan penelitian dalam dua tahap (Bab

I-III dan Bab IV) dengan form progress dan upload dokumen; (5) Sistem pencairan dana bertahap (50%-25%-25%) dengan tracking status dan approval oleh Admin LPPM; (6) Manajemen tiga jenis seminar (Proposal, Internal, dan Publik) dengan penjadwalan, daftar peserta, dan dokumentasi hasil seminar; (7) Modul tracking publikasi dan luaran penelitian dengan input manual oleh dosen dan validasi oleh Admin LPPM; (8) Dashboard monitoring untuk pimpinan yang menampilkan capaian penelitian, pengabdian, publikasi, dan seminar dalam bentuk statistik dan visualisasi chart; (9) Sistem notifikasi otomatis melalui email untuk update status proposal, reminder deadline monitoring, dan jadwal seminar; serta (10) Audit trail untuk mencatat setiap aktivitas user dalam sistem guna memastikan akuntabilitas dan transparansi.

Sistem ini akan di-deploy pada Virtual Private Server (VPS) dengan web server Nginx dan process manager PM2 untuk memastikan availability dan scalability. Evaluasi kelayakan sistem akan dilakukan melalui tiga metode pengujian: Black Box Testing untuk menguji fungsionalitas sistem sesuai requirement, White Box Testing untuk menguji logika kode dan alur program, serta User Acceptance Testing (UAT) dengan melibatkan stakeholder dari STAI Ali bin Abi Thalib untuk memvalidasi bahwa sistem memenuhi kebutuhan pengguna. Penelitian ini diharapkan dapat memberikan kontribusi baik secara teoritis dalam penerapan PXP untuk pengembangan sistem informasi LPPM, maupun secara praktis dalam meningkatkan efisiensi operasional dan kualitas pengelolaan penelitian di STAI Ali bin Abi Thalib Surabaya.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah diuraikan, maka rumusan masalah dalam penelitian ini adalah sebagai berikut:

a. Bagaimana merancang sistem informasi manajemen LPPM yang dapat mengotomatisasi proses pengajuan proposal, monitoring penelitian, pencairan dana, manajemen seminar, dan pelaporan publikasi di STAI Ali bin Abi Thalib Surabaya?
b. Bagaimana menerapkan metodologi Personal Extreme Programming (PXP) dalam pengembangan sistem informasi manajemen LPPM dengan iterasi pendek, test-driven development, dan continuous integration?
c. Bagaimana mengevaluasi kelayakan sistem informasi manajemen LPPM melalui pengujian Black Box, White Box, dan User Acceptance Testing (UAT)?

1.3 Tujuan Penelitian

1.3.1 Tujuan Umum

Tujuan umum dari penelitian ini adalah mengembangkan sistem informasi manajemen data Lembaga Penelitian dan Pengabdian Masyarakat (LPPM)

di STAI Ali bin Abi Thalib Surabaya menggunakan metodologi Personal Extreme Programming (PXP) untuk meningkatkan efisiensi dan efektivitas pengelolaan penelitian dan pengabdian masyarakat.

1.3.2 Tujuan Khusus

Tujuan khusus dari penelitian ini adalah sebagai berikut:

1. Merancang dan mengimplementasikan sistem informasi manajemen LPPM berbasis web menggunakan teknologi Next.js, Prisma, TypeScript, dan MySQL yang mencakup fitur-fitur sebagai berikut:
   - Manajemen user dengan role-based access control untuk Admin LPPM, Dosen/Peneliti, Reviewer, dan Pimpinan
   - Pengajuan dan review proposal penelitian dengan workflow approval dan penilaian
   - Manajemen kontrak penelitian otomatis
   - Monitoring pelaksanaan penelitian dalam dua tahap (Bab I-III dan Bab IV)
   - Sistem pencairan dana bertahap (50%-25%-25%) dengan tracking dan approval
   - Manajemen tiga jenis seminar (Proposal, Internal, dan Publik)
   - Tracking publikasi dan luaran penelitian
   - Dashboard monitoring untuk pimpinan dengan statistik dan visualisasi data
   - Sistem notifikasi otomatis dan audit trail
2. Menerapkan metodologi Personal Extreme Programming (PXP) melalui empat iterasi pengembangan dengan praktik test-driven development menggunakan Jest, continuous integration menggunakan GitHub Actions, serta demo dan review dengan stakeholder LPPM pada setiap iterasi, dan menghasilkan user manual untuk semua role serta technical documentation untuk maintenance sistem.
3. Mengevaluasi kelayakan sistem informasi manajemen LPPM melalui pengujian Black Box untuk memvalidasi fungsionalitas sistem sesuai requirement, White Box untuk memvalidasi logika kode dan alur program, serta User Acceptance Testing (UAT) dengan melibatkan stakeholder dari STAI Ali bin Abi Thalib Surabaya untuk memvalidasi penerimaan dan kepuasan pengguna terhadap sistem.

1.4 Batasan dan Asumsi Penelitian

1.4.1 Batasan Penelitian

Batasan penelitian ini dibuat untuk memberikan fokus yang jelas pada ruang lingkup pengembangan sistem. Adapun batasan-batasan dalam penelitian ini adalah sebagai berikut:

1. Sistem hanya mengelola penelitian dan pengabdian masyarakat internal STAI Ali bin Abi Thalib Surabaya yang didanai oleh institusi, tidak mencakup penelitian eksternal atau hibah kompetitif nasional (seperti hibah Dikti atau Kemenag).
2. Sistem mencakup tiga skema penelitian yaitu Penelitian Dasar, Penelitian Terapan, dan Penelitian Pengembangan sesuai dengan Buku Panduan Penelitian STAI Ali bin Abi Thalib Surabaya tahun 2024. Penelitian Kolaboratif tidak termasuk dalam scope sistem pada fase pertama pengembangan.
3. Sistem tidak terintegrasi dengan sistem eksternal lainnya, yaitu:
    - Lembaga Penjamin Mutu (LPM) atau Sistem Penjaminan Mutu Internal (SPMI) dari Kemendikbud
    - Sistem keuangan institusi untuk pengelolaan anggaran dan pencairan danaSistem Informasi Akademik (SIAKAD) untuk data dosen dan mahasiswa
    - Repository publikasi eksternal seperti SINTA, Google Scholar, Scopus, atau database jurnal internasional
4. Sistem hanya menyediakan fitur tracking publikasi secara manual, dimana dosen/peneliti menginput data publikasi sendiri dan divalidasi oleh Admin LPPM. Tidak ada fitur auto-sync atau web scraping untuk mengambil data publikasi dari repository eksternal.
5. Sistem mencatat capaian penelitian, pengabdian masyarakat, publikasi, dan seminar dalam bentuk statistik dan visualisasi pada dashboard. Sistem tidak memiliki fitur target pencapaian atau warning jika target belum tercapai, karena fokus sistem adalah monitoring capaian aktual, bukan enforcement terhadap target institusional.
6. Evaluasi sistem fokus pada kelayakan sistem dari aspek fungsionalitas (Black Box Testing), logika program (White Box Testing), dan penerimaan pengguna (User Acceptance Testing). Penelitian ini tidak mengukur dampak operasional sistem terhadap efisiensi waktu atau pengurangan tingkat kesalahan data secara kuantitatif (tidak ada studi before-after).

1.4.2 Asumsi Penelitian

Asumsi penelitian merupakan kondisi-kondisi yang diasumsikan akan terpenuhi selama pelaksanaan penelitian. Adapun asumsi-asumsi dalam penelitian ini adalah sebagai berikut:

1. Infrastruktur dan Teknologi:
    - STAI Ali bin Abi Thalib Surabaya memiliki infrastruktur jaringan internet yang stabil dan memadai untuk mendukung akses sistem berbasis web.

- Virtual Private Server (VPS) yang digunakan untuk deployment sistem memiliki spesifikasi yang cukup (minimal 2 CPU cores, 4GB RAM, dan 40GB storage) dan uptime yang baik.
- Admin LPPM, Dosen/Peneliti, Reviewer, dan Pimpinan memiliki perangkat (laptop/PC/smartphone) dengan browser modern yang mendukung teknologi web terkini.

2. Data dan Konten:
   - Data penelitian, pengabdian masyarakat, dan publikasi yang akan di-input ke dalam sistem tersedia dan lengkap dari pihak LPPM STAI Ali bin Abi Thalib.
   - Dokumen-dokumen pendukung seperti proposal, laporan hasil penelitian, dan bukti publikasi tersedia dalam format digital (PDF, Word, Excel) yang dapat di-upload ke sistem.
3. Stakeholder dan Pengguna:
   - Admin LPPM, Dosen/Peneliti, Reviewer, dan Pimpinan STAI Ali bin Abi Thalib bersedia untuk berpartisipasi dalam proses demo dan review pada setiap iterasi pengembangan sesuai metodologi PXP.
   - Stakeholder bersedia untuk terlibat dalam User Acceptance Testing (UAT) dan memberikan feedback terhadap sistem yang dikembangkan.
   - Pengguna sistem memiliki kemampuan dasar dalam mengoperasikan aplikasi web dan bersedia untuk mengikuti pelatihan atau membaca user manual yang disediakan.
4. Regulasi dan Kebijakan:
   - Buku Panduan Penelitian STAI Ali bin Abi Thalib Surabaya tahun 2024 yang menjadi acuan dalam perancangan sistem tidak mengalami perubahan signifikan selama masa pengembangan sistem.
   - Tidak ada perubahan kebijakan institusional terkait proses penelitian, pengabdian masyarakat, dan publikasi yang dapat mempengaruhi requirement sistem secara fundamental.
5. Keamanan dan Privasi:
   - Data penelitian, publikasi, dan informasi pengguna yang disimpan dalam sistem dianggap bersifat internal institusi dan tidak mengandung data pribadi yang bersifat sangat sensitif (seperti data keuangan pribadi atau data kesehatan).
   - Implementasi protokol keamanan dasar seperti HTTPS, JWT authentication, password hashing, dan role-based access control dianggap cukup untuk melindungi data dalam konteks institusi pendidikan.

1.5 Manfaat Penelitian

1.5.1 Manfaat Teoritis

Penelitian ini diharapkan dapat memberikan kontribusi teoritis dalam beberapa aspek, yaitu:

1. Memberikan kontribusi terhadap pengembangan ilmu pengetahuan di bidang sistem informasi, khususnya dalam perancangan dan implementasi sistem informasi manajemen untuk lembaga penelitian dan pengabdian masyarakat di perguruan tinggi.
2. Memberikan bukti empiris mengenai penerapan metodologi Personal Extreme Programming (PXP) dalam konteks pengembangan sistem informasi oleh individu (mahasiswa tugas akhir) dengan kompleksitas sedang-tinggi, sehingga dapat menjadi referensi bagi penelitian selanjutnya yang menggunakan metodologi serupa.
3. Memberikan insight mengenai implementasi teknologi modern (Next.js, Prisma, TypeScript) dalam pengembangan sistem informasi manajemen di institusi pendidikan, serta evaluasi kelebihan dan tantangan yang dihadapi dalam penerapan teknologi tersebut.
4. Memperkaya literatur mengenai integrasi praktik-praktik modern software engineering seperti test-driven development (TDD), continuous integration/continuous deployment (CI/CD), dan iterative development dalam konteks tugas akhir mahasiswa.

1.5.2 Manfaat Praktis

Penelitian ini diharapkan dapat memberikan manfaat praktis bagi berbagai pihak, yaitu:

1. Bagi STAI Ali bin Abi Thalib Surabaya:
    - Meningkatkan efisiensi operasional dalam pengelolaan penelitian dan pengabdian masyarakat melalui otomasi proses-proses manual yang selama ini memakan waktu.
    - Meningkatkan transparansi dan akuntabilitas dalam pengelolaan dana penelitian dengan sistem tracking pencairan bertahap dan audit trail.
    - Menyediakan dashboard monitoring yang memudahkan pimpinan dalam mengambil keputusan berbasis data terkait capaian penelitian, publikasi, dan seminar.
    - Mendukung pencapaian visi institusi untuk menjadi universitas yang unggul dalam penelitian di bidang Agama Islam, Pendidikan, dan Sosial Humaniora.

2. Bagi Lembaga Penelitian dan Pengabdian Masyarakat (LPPM):
    - Memudahkan Admin LPPM dalam mengelola seluruh siklus penelitian mulai dari pengajuan proposal, review, kontrak, monitoring, hingga pelaporan dalam satu platform terpusat.
    - Mengurangi beban administratif dengan otomasi notifikasi, reminder, dan generate dokumen kontrak secara otomatis.
    - Meningkatkan akurasi data dan mengurangi kesalahan akibat duplikasi atau inkonsistensi format dokumen yang sebelumnya terjadi pada sistem manual.
    - Memudahkan dalam melakukan rekapitulasi data penelitian dan publikasi untuk keperluan pelaporan ke pimpinan atau akreditasi institusi.
3. Bagi Dosen/Peneliti:
    - Memudahkan proses pengajuan proposal penelitian dan pengabdian masyarakat dengan interface yang user-friendly dan workflow yang jelas.
    - Mempercepat proses review dan approval proposal dengan sistem notifikasi otomatis yang menginformasikan status proposal secara real-time.
    - Memudahkan dalam melaporkan progress penelitian dan publikasi tanpa harus mengirim email atau mengisi form Excel secara manual.
    - Meningkatkan transparansi terkait status pencairan dana penelitian sehingga dapat merencanakan pelaksanaan penelitian dengan lebih baik.
4. Bagi Mahasiswa dan Peneliti Lain:
    - Menjadi referensi atau studi kasus dalam pengembangan sistem informasi manajemen untuk lembaga penelitian di perguruan tinggi dengan menggunakan teknologi modern.
    - Memberikan contoh implementasi metodologi Personal Extreme Programming (PXP) secara praktis dalam konteks tugas akhir individu.
    - Menyediakan technical documentation yang dapat membantu developer lain dalam melakukan maintenance atau pengembangan lebih lanjut terhadap sistem.
5. Bagi Masyarakat Umum:
    - Meningkatkan kualitas dan kuantitas penelitian yang dihasilkan oleh STAI Ali bin Abi Thalib Surabaya yang dapat memberikan manfaat bagi pengembangan ilmu pengetahuan dan solusi terhadap permasalahan masyarakat.

- Mendukung diseminasi hasil penelitian melalui publikasi yang lebih terkelola dengan baik, sehingga hasil penelitian dapat diakses dan dimanfaatkan oleh masyarakat luas

1.6 Sistematika Penulisan

Sistematika penulisan laporan tugas akhir ini disusun untuk memberikan gambaran yang jelas dan terstruktur mengenai isi dari setiap bab. Laporan tugas akhir ini terdiri dari lima bab dengan sistematika sebagai berikut:

- **BAB I PENDAHULUAN**

  Bab ini menguraikan latar belakang penelitian yang menjelaskan konteks masalah, kondisi aktual LPPM STAI Ali bin Abi Thalib, dan urgensi pengembangan sistem informasi manajemen. Selain itu, bab ini juga memuat rumusan masalah yang akan dijawab, tujuan penelitian baik umum maupun khusus, batasan dan asumsi penelitian untuk memberikan fokus yang jelas, manfaat penelitian baik secara teoritis maupun praktis, serta sistematika penulisan laporan tugas akhir.
- **BAB II TINJAUAN PUSTAKA**

  Bab ini membahas landasan teori dan konsep-konsep yang relevan dengan penelitian, yang mencakup teori sistem informasi manajemen, konsep Lembaga Penelitian dan Pengabdian Masyarakat (LPPM), metodologi Personal Extreme Programming (PXP) beserta perbandingan dengan metodologi lain, teknologi yang digunakan (Next.js, Prisma, TypeScript, MySQL), Unified Modeling Language (UML) meliputi Use Case Diagram, Activity Diagram, Sequence Diagram, Robust Diagram, dan Class Diagram, Entity Relationship Diagram (ERD), serta metode pengujian perangkat lunak (Black Box, White Box, dan User Acceptance Testing). Bab ini juga memuat penelitian terdahulu yang relevan dan analisis gap untuk menunjukkan posisi penelitian ini dalam konteks keilmuan yang lebih luas.
- **BAB III METODOLOGI PENELITIAN**

  Bab ini menjelaskan secara rinci tahapan dan metode yang digunakan dalam penelitian, yang mencakup kerangka penelitian, metode pengumpulan data (wawancara, observasi, dan studi literatur), analisis kebutuhan sistem (fungsional dan non-fungsional), perancangan sistem menggunakan UML dan ERD, implementasi Personal Extreme Programming (PXP) dalam empat iterasi dengan detail aktivitas pada setiap iterasi, metode pengujian sistem (Black Box, White Box, dan UAT), serta rencana deployment sistem ke Virtual Private Server (VPS).
- **BAB IV HASIL DAN PEMBAHASAN**

  Bab ini menyajikan hasil dari implementasi sistem informasi manajemen LPPM, yang mencakup hasil analisis kebutuhan sistem, hasil

perancangan sistem dalam bentuk diagram UML dan ERD, hasil implementasi sistem dengan screenshot interface untuk setiap modul (pengajuan proposal, review, monitoring, pencairan dana, seminar, publikasi, dashboard), hasil pengujian Black Box dengan skenario test case dan hasil pengujian, hasil pengujian White Box dengan analisis coverage, hasil User Acceptance Testing (UAT) dengan responden dari stakeholder STAI Ali bin Abi Thalib, serta pembahasan mengenai kelebihan, kekurangan, dan tantangan yang dihadapi selama pengembangan sistem.

- **BAB V PENUTUP**

  Bab ini merupakan bab penutup yang berisi kesimpulan dari hasil penelitian yang menjawab rumusan masalah dan mencapai tujuan penelitian, serta saran untuk pengembangan lebih lanjut terhadap sistem maupun penelitian sejenis di masa depan.

BAB II
LANDASAN TEORI

2.1 Penelitian Terdahulu

Penelitian terdahulu merupakan kajian terhadap penelitian-penelitian yang telah dilakukan sebelumnya yang memiliki relevansi dengan topik penelitian ini. Tinjauan terhadap penelitian terdahulu bertujuan untuk mengidentifikasi gap penelitian, memahami metodologi yang telah digunakan, serta memposisikan penelitian ini dalam konteks pengembangan keilmuan yang lebih luas. Berikut adalah beberapa penelitian terdahulu yang relevan dengan pengembangan sistem informasi manajemen LPPM menggunakan Personal Extreme Programming:

No	Penulis & Tahun	Judul Penelitian	Metodologi	Teknologi	Hasil	Relevansi	Perbedaan
1	[Nama Peneliti], [Tahun]	Sistem Informasi Manajemen Penelitian pada LPPM Universitas X	Waterfall	PHP, MySQL, Bootstrap	Sistem dapat mengelola proposal, monitoring, dan pelaporan penelitian	Sama-sama mengembangkan sistem informasi LPPM dengan fitur utama pengelolaan penelitian	Penelitian ini menggunakan metodologi PXP dengan teknologi modern (Next.js, Prisma, TypeScript), mencakup manajemen seminar dan publikasi, serta menggunakan TDD dan CI/CD
2	[Nama Penel	Pengembangan E-Research	Scrum	Laravel, React,	Sistem meningkatkan	Menggunakan metodolog	Penelitian ini menggu

	iti], [Tahun]	System menggunakan Agile Scrum		Postgre SQL	efisiensi penelitian sebesar 35%	i agile untuk pengembangan sistem penelitian	nakan PXP (individu) bukan Scrum (tim), teknologi full-stack Next.js, dan konteks STAI dengan skema penelitian spesifik
3	[Nama Peneliti], [Tahun]	Implementasi Personal Extreme Programming untuk Aplikasi E-Commerce	PXP	Node.js, Express, Mongo DB	PXP efektif untuk pengembangan individu dengan kualitas kode baik	Sama-sama menggunakan metodologi PXP	Penelitian ini fokus pada sistem LPPM (domain berbeda), menggunakan Next.js + Prisma (lebih modern), dan konteks institusi pendidikan
4	[Nama Peneliti], [Tahun]	Sistem Monitoring Penelitian Berbasis Web pada Perguruan Tinggi Y	RAD	CodeIgniter, MySQL	Sistem mempercepat monitoring penelitian	Sama-sama mengembangkan sistem monitoring penelitian	Penelitian ini mencakup full lifecycle (proposal–seminar–publikasi),

							menggunakan PXP dengan TDD, dan teknologi modern Next.js
5	[Nama Peneliti], [Tahun]	Pengembangan Sistem Informasi LPPM dengan Integrasi SINTA	Prototyping	Laravel, Vue.js, API SINTA	Sistem terintegrasi dengan SINTA untuk tracking publikasi otomatis	Sama-sama mengelola publikasi penelitian	Penelitian ini fokus pada tracking manual (tanpa integrasi eksternal), menggunakan PXP, dan mencakup manajemen seminar

6	[Nama Peneliti], [Tahun]	Aplikasi Manajemen Hibah Penelitian menggunakan Next.js	Iterative	Next.js, Firebase	Aplikasi dapat mengelola hibah penelitian dengan performa baik	Sama-sama mengguna kan Next.js	Penelitian ini fokus pada penelitian internal (bukan hibah eksternal), menggunakan Prisma + MySQL, metodologi PXP dengan TDD, dan mencakup seminar
7	[Nama Peneliti], [Tahun]	Penerapan Test-Driven Development pada Sistem Akademik	XP	Django, PostgreSQL	TDD meningkatkan kualitas kode dan mengurangi bug sebesar 40%	Sama-sama menerapkan TDD dalam pengembangan sistem	Penelitian ini menggunakan PXP (individu), fokus pada sistem LPPM, dan teknologi Next.js + Prisma dengan CI/CD

2.2 Dasar Teori

2.2.1 Sistem Informasi Manajemen (SIM)

Sistem Informasi Manajemen (SIM) adalah sistem yang dirancang untuk mengumpulkan, menyimpan, memproses, dan mendistribusikan informasi yang dibutuhkan untuk mendukung pengambilan keputusan dan aktivitas manajerial dalam suatu organisasi [ref:X]. Menurut [ref:X], SIM

merupakan kumpulan dari subsistem yang saling berhubungan, berkumpul bersama-sama dan membentuk satu kesatuan, saling berinteraksi dan bekerja sama antara bagian satu dengan lainnya dengan cara-cara tertentu untuk melakukan fungsi pengolahan data, menerima masukan (input) berupa data, kemudian mengolahnya (processing), dan menghasilkan keluaran (output) berupa informasi.

Menurut [ref:X], SIM terdiri dari lima komponen utama yang saling terintegrasi:

1. Hardware (Perangkat Keras): Komponen fisik yang digunakan untuk menjalankan sistem, seperti server, komputer, perangkat jaringan, dan perangkat penyimpanan data.
2. Software (Perangkat Lunak): Program aplikasi dan sistem operasi yang mengolah data menjadi informasi, termasuk database management system, application software, dan user interface.
3. Data: Fakta-fakta mentah yang dikumpulkan, disimpan, dan diolah oleh sistem untuk menghasilkan informasi yang berguna bagi pengambilan keputusan.
4. Prosedur: Aturan dan kebijakan yang mengatur bagaimana sistem dioperasikan, termasuk Standard Operating Procedure (SOP) untuk input data, proses pengolahan, dan output informasi.
5. Manusia (People): Pengguna sistem yang terdiri dari end-user yang menggunakan sistem untuk keperluan operasional, dan administrator yang mengelola dan memelihara sistem.

SIM memiliki beberapa fungsi utama dalam organisasi [ref:X]:

1. Mengumpulkan Data: SIM mengumpulkan data dari berbagai sumber baik internal maupun eksternal organisasi secara sistematis.
2. Menyimpan Data: Data yang dikumpulkan disimpan dalam database yang terstruktur untuk memudahkan akses dan retrieval.
3. Memproses Data: Data mentah diolah menjadi informasi yang meaningful melalui proses kalkulasi, klasifikasi, dan summarization.
4. Mendistribusikan Informasi: Informasi yang dihasilkan didistribusikan kepada user yang memerlukan sesuai dengan hak akses dan kebutuhan masing-masing.
5. Mendukung Pengambilan Keputusan: SIM menyediakan informasi yang akurat dan tepat waktu untuk mendukung manajemen dalam pengambilan keputusan strategis, taktis, dan operasional.

Dalam konteks Lembaga Penelitian dan Pengabdian Masyarakat (LPPM) di perguruan tinggi, SIM memiliki peran khusus dalam

mengelola seluruh siklus penelitian mulai dari perencanaan, pelaksanaan, monitoring, hingga pelaporan dan diseminasi hasil penelitian [ref:X]. SIM LPPM harus mampu mengintegrasikan data dari berbagai aktivitas penelitian, menyediakan dashboard monitoring untuk pimpinan, mengotomatisasi workflow approval dan notifikasi, serta memastikan transparansi dan akuntabilitas dalam pengelolaan dana penelitian. Penelitian oleh [ref:X] menunjukkan bahwa implementasi SIM pada LPPM dapat meningkatkan produktivitas peneliti sebesar 30-40% dengan mengurangi beban administratif dan mempercepat proses approval proposal penelitian.

Penelitian ini mengembangkan SIM untuk LPPM STAI Ali bin Abi Thalib Surabaya yang mencakup seluruh komponen SIM (hardware berupa VPS deployment, software berupa aplikasi Next.js, data berupa database MySQL penelitian dan publikasi, prosedur berupa workflow sesuai Buku Panduan Penelitian, dan people berupa Admin LPPM, Dosen, Reviewer, dan Pimpinan). Sistem ini dirancang untuk mengotomatisasi fungsi-fungsi SIM dalam konteks pengelolaan penelitian, sehingga meningkatkan efisiensi operasional dan mendukung pengambilan keputusan berbasis data.

2.2.2 Lembaga Penelitian dan Pengabdian Masyarakat (LPPM)

Lembaga Penelitian dan Pengabdian Masyarakat (LPPM) adalah unit organisasi di perguruan tinggi yang bertanggung jawab untuk merencanakan, melaksanakan, mengendalikan, dan mengembangkan kegiatan penelitian dan pengabdian kepada masyarakat [ref:X]. Berdasarkan Peraturan Menteri Riset, Teknologi, dan Pendidikan Tinggi Nomor 44 Tahun 2015 tentang Standar Nasional Pendidikan Tinggi, penelitian merupakan kegiatan yang dilakukan menurut kaidah dan metode ilmiah secara sistematis untuk memperoleh informasi, data, dan keterangan yang berkaitan dengan pemahaman dan/atau pengujian suatu cabang pengetahuan dan teknologi [ref:X].

Menurut [ref:X], LPPM memiliki fungsi dan tanggung jawab utama sebagai berikut:

1. Perencanaan Penelitian: Menyusun roadmap penelitian institusi, mengidentifikasi tema-tema penelitian prioritas, dan mengalokasikan anggaran penelitian sesuai dengan kebutuhan dan prioritas institusi.
2. Fasilitasi Penelitian: Memberikan dukungan administratif dan teknis kepada dosen/peneliti dalam menyusun proposal, melaksanakan penelitian, dan mempublikasikan hasil penelitian.

3. Monitoring dan Evaluasi: Memantau pelaksanaan penelitian untuk memastikan bahwa penelitian berjalan sesuai dengan rencana, timeline, dan anggaran yang telah ditetapkan.
4. Pengelolaan Dana Penelitian: Mengelola alokasi dan pencairan dana penelitian secara transparan dan akuntabel sesuai dengan regulasi yang berlaku.
5. Diseminasi Hasil Penelitian: Memfasilitasi publikasi hasil penelitian dalam jurnal ilmiah, prosiding seminar, buku, atau media lainnya untuk menyebarluaskan hasil penelitian kepada masyarakat akademik dan masyarakat luas.
6. Penjaminan Mutu Penelitian: Memastikan bahwa penelitian yang dilakukan memenuhi standar kualitas melalui peer review, seminar proposal, dan evaluasi hasil penelitian.

Proses bisnis LPPM secara umum mencakup beberapa tahapan berikut [ref:X]:

1. Pengajuan Proposal: Dosen/peneliti menyusun dan mengajukan proposal penelitian kepada LPPM sesuai dengan format dan timeline yang ditetapkan.
2. Review dan Penilaian: Proposal dinilai oleh tim reviewer berdasarkan kriteria substansi (relevansi, metodologi, orisinalitas) dan administratif (kelengkapan dokumen, kesesuaian format).
3. Seminar Proposal: Proposal yang lolos review dipresentasikan dalam seminar proposal untuk mendapatkan masukan dan perbaikan dari tim penilai dan peserta seminar.
4. Persetujuan dan Kontrak: Proposal yang dinyatakan layak didanai diterbitkan Surat Keputusan (SK) dan kontrak penelitian yang mengikat antara peneliti dan institusi.
5. Pelaksanaan Penelitian: Peneliti melaksanakan penelitian sesuai dengan proposal yang telah disetujui dengan dukungan fasilitasi dari LPPM.
6. Monitoring: LPPM melakukan monitoring pelaksanaan penelitian secara berkala untuk memastikan penelitian berjalan sesuai rencana dan memberikan solusi jika terdapat kendala.
7. Pencairan Dana: Dana penelitian dicairkan secara bertahap sesuai dengan progress pelaksanaan penelitian dan kepatuhan terhadap timeline yang ditetapkan.
8. Pelaporan: Peneliti menyusun laporan hasil penelitian yang menjelaskan proses, temuan, dan kesimpulan dari penelitian yang telah dilakukan.
9. Seminar Hasil: Hasil penelitian dipresentasikan dalam seminar internal dan/atau seminar publik untuk mendapatkan feedback dan validasi dari komunitas akademik.

10. Publikasi dan Luaran: Peneliti wajib mempublikasikan hasil penelitian dalam bentuk artikel jurnal, buku, paten, atau luaran lainnya sesuai dengan komitmen dalam proposal.

Berdasarkan literatur [ref:X] dan [ref:X], pengelolaan data LPPM menghadapi beberapa tantangan, antara lain:

1. Dokumentasi Tidak Terstruktur: Dokumen proposal, laporan, dan publikasi tersimpan dalam berbagai format dan lokasi yang berbeda, menyulitkan pencarian dan akses.
2. Duplikasi dan Inkonsistensi Data: Tidak adanya sistem terpusat menyebabkan duplikasi data dan inkonsistensi informasi antara berbagai dokumen.
3. Kesulitan Monitoring Real-time: Proses monitoring yang manual memakan waktu dan tidak memberikan informasi progress yang real-time.
4. Pelaporan Manual: Rekapitulasi data untuk pelaporan ke pimpinan atau akreditasi memerlukan waktu lama karena harus mengumpulkan dan mengolah data dari berbagai sumber.
5. Kurangnya Transparansi: Tidak adanya platform terpusat membuat peneliti kesulitan untuk mengetahui status proposal, progress review, atau status pencairan dana secara transparan.

Penelitian ini mengembangkan sistem informasi manajemen khusus untuk LPPM STAI Ali bin Abi Thalib Surabaya yang dirancang sesuai dengan proses bisnis LPPM berdasarkan Buku Panduan Penelitian STAI Ali bin Abi Thalib 2024. Sistem mencakup seluruh tahapan dari pengajuan proposal, review, seminar proposal, kontrak, pelaksanaan, monitoring 2 tahap, pencairan dana bertahap (50-25-25%), seminar internal dan publik, hingga tracking publikasi. Sistem ini diharapkan dapat mengatasi tantangan-tantangan pengelolaan data LPPM dengan menyediakan platform terpusat, otomasi workflow, dan dashboard monitoring real-time.

2.2.3 Software Development Life Cycle SDLC

Software Development Life Cycle (SDLC) adalah proses sistematis yang digunakan untuk mengembangkan perangkat lunak berkualitas tinggi yang memenuhi kebutuhan pengguna dalam waktu dan biaya yang efisien [ref:X].

SDLC mencakup beberapa fase utama: Requirements Analysis, Design, Implementation, Testing, Deployment, dan Maintenance [ref:X].

Metodologi pengembangan perangkat lunak dapat diklasifikasikan menjadi dua kategori utama [ref:X]:

1. Predictive/Plan-Driven (Waterfall, V-Model): Metodologi yang menekankan pada perencanaan detail di awal proyek dengan asumsi bahwa requirement tidak akan berubah secara signifikan.
2. Adaptive/Agile (Scrum, XP, Kanban, PXP): Metodologi yang menekankan pada adaptabilitas terhadap perubahan requirement dengan iterasi pendek dan feedback berkelanjutan.

Perbandingan Metodologi Pengembangan

Kriteria	**Waterfall**	**RAD**	**Scrum**	**XP**	**PXP**
Paradigma	Predictive	Iterative	Agile	Agile	Agile (Personal)
Ukuran Tim	Flexible (5–50+)	Small–Medium (3–10)	5–9 orang	4–12 orang	1–3 orang (individu)
Durasi Iterasi	Linear (6–12 bulan+)	60–90 hari	2–4 minggu (Sprint)	1–3 minggu	1–2 minggu
Fleksibilitas Perubahan	Sangat rendah	Sedang	Tinggi	Sangat tinggi	Sangat tinggi
Dokumentasi	Sangat lengkap	Minimal	Moderate	Minimal	Minimal (fokus code quality)
Customer Involvement	Di awal dan akhir	Moderate	Sprint Review	Continuous (on-site)	Berkala (demo per iterasi)
Testing Approach	End-phase testing	Iterative testing	Sprint-based	TDD (Test-Driven Development)	TDD + Continuous Testing

Pair Programming	Tidak	Tidak	Opsional	Wajib	Code Review (pengganti pair programming)
Continuous Integration	Tidak	Tidak	Recommended	Wajib	Wajib (CI/CD)
Cocok untuk Requirement	Stabil, jelas	Moderate changes	Berubah-ubah	Sangat dinamis	Dinamis dengan feedback cepat
Kesesuaian untuk TA Individu	✗ (terlalu kaku)	△ (butuh tim)	✗ (team 5–9)	✗ (team 4–12)	✓✓✓ (dirancang untuk individu)
Cocok untuk Sistem LPPM	✗ (butuh adaptasi cepat)	△	✓	✓	✓✓✓ (individu + quality assurance)

Berdasarkan tabel perbandingan di atas, PXP dipilih sebagai metodologi pengembangan dalam penelitian ini karena beberapa alasan:

1. Konteks Individu: Penelitian ini merupakan tugas akhir yang dikerjakan oleh satu mahasiswa, sehingga metodologi yang dirancang untuk tim (Scrum, XP) tidak sesuai. PXP dirancang khusus untuk pengembangan individu atau tim sangat kecil.
2. Iterasi Cepat dengan Feedback: PXP memungkinkan iterasi 1-2 minggu dengan demo/review berkala kepada stakeholder LPPM, sehingga requirement dapat disesuaikan dengan cepat berdasarkan feedback.
3. Quality Assurance: PXP menekankan TDD dan continuous integration yang memastikan kualitas kode tetap terjaga meskipun dikerjakan oleh individu. Praktik ini mengurangi bug dan meningkatkan maintainability sistem.

4. Sustainable Pace: PXP tidak memaksakan overtime seperti beberapa metodologi agile lainnya, sehingga cocok untuk mahasiswa yang juga memiliki kewajiban akademik lainnya.
5. Dokumentasi Code-Centric: PXP fokus pada clean code, self-documenting code, dan user manual/technical documentation, sehingga sistem mudah dipahami dan di-maintain di masa depan.

Penelitian ini menerapkan PXP dengan 4 iterasi, masing-masing berdurasi 4-5 minggu, dengan aktivitas planning (user stories), design (UML), coding (TDD dengan Jest), testing (Black Box, White Box), continuous integration (GitHub Actions), dan demo/review dengan stakeholder LPPM setiap akhir iterasi. Pemilihan PXP memastikan bahwa sistem dikembangkan dengan kualitas tinggi meskipun dikerjakan secara individu dalam waktu terbatas (1 semester).

2.2.4 Personal Extreme Programming (PXP)

Extreme Programming (XP) adalah metodologi pengembangan perangkat lunak yang termasuk dalam kategori agile development, yang menekankan pada adaptabilitas terhadap perubahan requirement, iterasi pendek, kolaborasi intensif dengan customer, dan praktik-praktik teknis untuk menghasilkan software berkualitas tinggi [ref:X]. XP dikembangkan oleh Kent Beck pada akhir 1990-an sebagai respons terhadap kelemahan metodologi tradisional (Waterfall) yang kaku dan lambat dalam merespons perubahan [ref:X].

XP memiliki lima nilai inti [ref:X]:

1. Communication (Komunikasi): Komunikasi yang efektif antara developer, customer, dan stakeholder lainnya.
2. Simplicity (Kesederhanaan): Fokus pada solusi yang paling sederhana yang dapat bekerja (do the simplest thing that could possibly work).
3. Feedback (Umpan Balik): Mendapatkan feedback secepat mungkin dari customer dan testing untuk mengarahkan development.
4. Courage (Keberanian): Keberanian untuk refactor kode, mengubah desain, dan menghadapi masalah secara langsung.
5. Respect (Rasa Hormat): Menghormati kontribusi setiap anggota tim dan menghargai perbedaan pendapat.

XP menerapkan 12 praktik utama [ref:X]:

1. Planning Game: Perencanaan iterasi dengan melibatkan customer untuk menentukan prioritas fitur.
2. Small Releases: Rilis versi kecil secara berkala untuk mendapatkan feedback cepat.

3. Metaphor: Menggunakan metafora atau analogi untuk menjelaskan sistem kepada semua pihak.
4. Simple Design: Desain yang sederhana, tidak over-engineering.
5. Test-Driven Development (TDD): Menulis test sebelum menulis kode implementasi.
6. Refactoring: Memperbaiki struktur kode secara berkelanjutan tanpa mengubah fungsionalitas.
7. Pair Programming: Dua programmer bekerja bersama pada satu komputer untuk meningkatkan kualitas kode.
8. Collective Code Ownership: Semua anggota tim bertanggung jawab terhadap seluruh kode.
9. Continuous Integration: Integrasi kode ke repository utama sesering mungkin (minimal sehari sekali).
10. 40-Hour Week: Menjaga work-life balance dengan tidak bekerja overtime secara berlebihan.
11. On-Site Customer: Customer tersedia setiap saat untuk menjawab pertanyaan dan memberikan feedback.
12. Coding Standards: Mengikuti standar penulisan kode yang konsisten.

Personal Extreme Programming (PXP) adalah adaptasi dari metodologi XP yang dirancang untuk pengembangan perangkat lunak oleh individu atau tim yang sangat kecil (1-3 orang) [ref:X]. PXP dikembangkan oleh Nawrocki dan Wojciechowski pada tahun 2001 untuk mengatasi keterbatasan XP yang memerlukan tim dengan ukuran minimal 4-12 orang [ref:X]. PXP mempertahankan nilai-nilai inti dan sebagian besar praktik XP, namun dengan modifikasi dan simplifikasi untuk konteks individu.

Menurut [ref:X], PXP melakukan adaptasi sebagai berikut:

1. Pair Programming → Code Review dan TDD: Karena tidak ada pasangan, PXP mengganti pair programming dengan self-code review dan penekanan lebih kuat pada TDD untuk menjaga kualitas kode.
2. On-Site Customer → Regular Demo/Review: Customer tidak perlu tersedia setiap saat, tetapi dijadwalkan demo/review secara berkala (misalnya setiap akhir iterasi).
3. Collective Code Ownership → Personal Ownership dengan Dokumentasi: Individu bertanggung jawab penuh terhadap kode, tetapi harus membuat dokumentasi yang baik untuk transfer knowledge.
4. Planning Game → User Stories dan Iterasi Pendek: Perencanaan dilakukan dengan menulis user stories dan membagi pekerjaan ke dalam iterasi pendek (1-2 minggu).
5. 40-Hour Week → Sustainable Pace: Tetap menjaga work-life balance dengan tidak memaksakan diri bekerja secara berlebihan.

PXP memiliki tahapan yang iteratif dan incremental [ref:X]:

1. Planning: Mengidentifikasi user stories, memprioritaskan fitur, dan merencanakan iterasi.
2. Design: Merancang arsitektur sistem dan desain detail untuk fitur yang akan dikembangkan dalam iterasi.
3. Coding dengan TDD: Menulis test terlebih dahulu (unit test), kemudian menulis kode implementasi untuk membuat test pass, dan melakukan refactoring.
4. Testing: Melakukan testing pada berbagai level (unit, integration, system) untuk memastikan kualitas.
5. Integration: Mengintegrasikan kode ke repository utama secara continuous dengan bantuan CI/CD tools.
6. Demo/Review: Mendemonstrasikan hasil iterasi kepada stakeholder dan mendapatkan feedback untuk iterasi berikutnya.

Menurut [ref:X], PXP memiliki beberapa kelebihan untuk konteks pengembangan individu:

1. Fleksibilitas Tinggi: Dapat beradaptasi dengan cepat terhadap perubahan requirement tanpa overhead koordinasi tim besar.
2. Quality Assurance melalui TDD: Praktik TDD memastikan setiap fungsi diuji secara otomatis, mengurangi bug dan meningkatkan confidence dalam kode.
3. Iterasi Cepat: Iterasi pendek (1-2 minggu) memungkinkan feedback cepat dan penyesuaian arah development sesuai kebutuhan.
4. Sustainable Development: Menjaga kualitas kode melalui refactoring berkelanjutan dan simple design, sehingga sistem mudah dimaintain dan dikembangkan lebih lanjut.
5. Continuous Integration: CI/CD memastikan setiap perubahan kode langsung diuji dan di-deploy, mempercepat siklus development.
6. Cocok untuk Tugas Akhir: PXP sangat sesuai untuk konteks tugas akhir mahasiswa yang dikerjakan secara individu dengan keterbatasan waktu (1 semester) dan sumber daya.

Penelitian ini menerapkan PXP sebagai metodologi pengembangan sistem informasi manajemen LPPM dengan 4 iterasi, masing-masing iterasi berdurasi 4-5 minggu. Setiap iterasi mencakup planning (menulis user stories), design (membuat UML), coding dengan TDD menggunakan Jest, testing (Black Box dan White Box), continuous integration dengan GitHub Actions, dan demo/review dengan stakeholder LPPM STAI Ali bin Abi Thalib. Penerapan PXP memastikan sistem dikembangkan secara iteratif dengan quality assurance yang baik meskipun dikerjakan oleh individu.

2.2.5 Unified Modeling Language (UML)

Unified Modeling Language (UML) adalah bahasa pemodelan visual standar untuk mendokumentasikan, menspesifikasikan, dan memvisualisasikan artifact dari sistem perangkat lunak [ref:X]. UML dikembangkan oleh Object Management Group (OMG) dan pertama kali dirilis pada tahun 1997 sebagai standar untuk object-oriented modeling [ref:X]. UML menyediakan berbagai jenis diagram untuk memodelkan aspek statis (struktur) dan dinamis (behavior) dari sistem.

UML memiliki 14 jenis diagram yang diklasifikasikan menjadi dua kategori [ref:X]:

1. Structure Diagrams (Diagram Struktur): Menggambarkan elemen statis dari sistem, seperti Class Diagram, Object Diagram, Component Diagram, dan Deployment Diagram.
2. Behavior Diagrams (Diagram Perilaku): Menggambarkan aspek dinamis dari sistem, seperti Use Case Diagram, Activity Diagram, Sequence Diagram, State Machine Diagram, dan Collaboration Diagram.

   Dalam penelitian ini, diagram UML yang digunakan adalah Use Case Diagram, Activity Diagram, Sequence Diagram, Robust Diagram (Collaboration Diagram), dan Class Diagram untuk memodelkan sistem informasi manajemen LPPM.

2.2.6 Use Case Diagram

Use Case Diagram adalah diagram UML yang menggambarkan interaksi antara actor (pengguna atau sistem eksternal) dengan sistem untuk mencapai tujuan tertentu (use case) [ref:X]. Use Case Diagram digunakan pada tahap analisis kebutuhan untuk memahami fungsionalitas sistem dari perspektif pengguna [ref:X].

Use Case Diagram terdiri dari beberapa komponen utama [ref:X]:

Notasi	Nama	Deskripsi	Simbol
Actor	Aktor	Entitas eksternal (user atau sistem lain) yang berinteraksi dengan sistem	Stick figure atau kotak berlabel `<<actor>>`
Use Case	Use Case	Fungsionalitas atau layanan yang disediakan sistem untuk aktor	Bentuk oval / ellipse

Association	Asosiasi	Hubungan komunikasi antara aktor dan use case	Garis solid
Include	Include	Relasi dimana satu use case selalu memanggil use case lain (mandatory)	Garis putus-putus dengan label `<<include>>`
Extend	Extend	Relasi di mana satu use case dapat memanggil use case lain secara opsional	Garis putus-putus dengan label `<<extend>>`
Generalization	Generalisasi	Relasi turunan (inheritance) antar aktor atau antar use case	Garis solid dengan panah segitiga terbuka
System Boundary	Batasan Sistem	Kotak yang membungkus seluruh use case yang termasuk dalam cakupan sistem	Kotak persegi panjang

Use Case Diagram digunakan dalam penelitian ini untuk:

1. Mengidentifikasi Actor: Admin LPPM, Dosen/Peneliti, Reviewer, dan Pimpinan sebagai pengguna sistem dengan use case yang berbeda.
2. Mendefinisikan Fungsionalitas: Mengidentifikasi use case utama seperti "Mengajukan Proposal", "Review Proposal", "Monitoring Penelitian", "Approve Pencairan Dana", "Lihat Dashboard", dll.
3. Memahami Interaksi: Menggambarkan bagaimana setiap actor berinteraksi dengan sistem untuk mencapai tujuan mereka.
4. Validasi Requirement: Use Case Diagram menjadi dasar diskusi dengan stakeholder untuk memvalidasi bahwa semua kebutuhan fungsional telah tercakup.

2.2.7 Activity Diagram

Activity Diagram adalah diagram UML yang menggambarkan alur kerja (workflow) atau proses bisnis dalam sistem dengan fokus pada urutan aktivitas dan kondisi decision [ref:X]. Activity Diagram mirip dengan flowchart tetapi lebih powerful karena mendukung concurrency (parallel activities) dan swimlane untuk menunjukkan responsibility [ref:X].

Notasi	Nama	Deskripsi	Simbol
Initial Node	Node Awal	Titik mulai dari activity diagram	Lingkaran solid hitam

Activity	Aktivitas	Aksi atau proses yang dilakukan dalam sistem	Persegi panjang dengan sudut membulat
Decision Node	Node Keputusan	Percabangan berdasarkan kondisi (if–else)	Diamond (belah ketupat)
Merge Node	Node Penggabungan	Menggabungkan beberapa alur menjadi satu	Diamond (belah ketupat)
Fork Node	Node Fork	Memulai aktivitas paralel (concurrent)	Garis tebal horizontal atau vertikal
Join Node	Node Join	Mengakhiri aktivitas paralel dan menunggu semua selesai	Garis tebal horizontal atau vertikal
Final Node	Node Akhir	Titik akhir dari activity diagram	Lingkaran dengan lingkaran solid di dalamnya
Swimlane	Swimlane	Membagi diagram berdasarkan actor/role yang bertanggung jawab	Kolom vertikal atau horizontal
Object Node	Node Objek	Data atau objek yang mengalir antar aktivitas	Persegi panjang

Activity Diagram digunakan dalam penelitian ini untuk:

1. Memodelkan Proses Bisnis: Menggambarkan alur proses pengajuan proposal, review, monitoring, pencairan dana, dan seminar sesuai Buku Panduan Penelitian STAI Ali bin Abi Thalib.
2. Menunjukkan Responsibility: Menggunakan swimlane untuk menunjukkan aktivitas mana yang dilakukan oleh Dosen, Admin LPPM, Reviewer, atau Sistem.
3. Mengidentifikasi Decision Point: Menunjukkan kondisi seperti "Jika proposal approved → lanjut kontrak, jika rejected → kembali ke Dosen untuk revisi".
4. Validasi Workflow: Activity Diagram menjadi dokumentasi workflow yang dapat divalidasi dengan stakeholder sebelum implementasi.

2.2.8 Sequence Diagram

Sequence Diagram adalah diagram UML yang menggambarkan interaksi antar objek dalam sistem berdasarkan urutan waktu (time sequence) dengan fokus pada message yang dipertukarkan [ref:X]. Sequence Diagram digunakan untuk memodelkan skenario spesifik dari use

case dengan menunjukkan bagaimana objek-objek berkolaborasi untuk menyelesaikan tugas [ref:X].

Notasi	Nama	Deskripsi	Simbol
Actor	Aktor	Pengguna yang memulai interaksi	Stick figure
Object	Objek	Instance dari class yang berpartisipasi dalam interaksi	Kotak persegi panjang dengan format *nama:Class*
Lifeline	Lifeline	Garis vertikal putus-putus yang menunjukkan keberadaan objek sepanjang waktu	Garis vertikal putus-putus
Activation	Aktivasi	Periode waktu di mana objek aktif melakukan proses	Persegi panjang tipis di atas lifeline
Synchronous Message	Pesan Sinkron	Pesan yang membutuhkan response sebelum eksekusi berlanjut	Garis solid dengan panah penuh
Asynchronous Message	Pesan Asinkron	Pesan yang tidak menunggu response (event/notification)	Garis solid dengan panah terbuka
Return Message	Pesan Kembali	Response dari pesan sinkron	Garis putus-putus dengan panah terbuka
Self Message	Pesan Diri	Objek memanggil method miliknya sendiri	Panah melengkung kembali ke lifeline
Destruction	Destruksi	Akhir dari keberadaan objek	Tanda X pada ujung lifeline

Sequence Diagram digunakan dalam penelitian ini untuk:

1. Memodelkan Interaksi Detail: Menggambarkan interaksi antara UI, Controller, Service, dan Database untuk setiap use case.
2. Menunjukkan Flow Data: Menggambarkan bagaimana data mengalir dari user input hingga response yang ditampilkan kembali ke user.
3. Validasi Desain Teknis: Sequence Diagram membantu memahami apakah desain sistem sudah benar dari perspektif technical implementation.
4. Dokumentasi untuk Developer: Sequence Diagram menjadi dokumentasi teknis yang memudahkan developer lain memahami cara kerja sistem.

2.2.9 Robust Diagram

Robust Diagram, juga dikenal sebagai Analysis Object Model atau Robustness Analysis, adalah diagram yang digunakan sebagai jembatan antara Use Case dan Sequence Diagram [ref:X]. Robust Diagram mengidentifikasi tiga jenis objek: Boundary (interface), Control (logic), dan Entity (data), sehingga membantu dalam transisi dari analisis ke desain [ref:X].

Komponen Robust Diagram

Notasi	Nama	Deskripsi	Simbol
Boundary Object	Objek Boundary	Interface yang berinteraksi langsung dengan actor (UI, form, layar)	Lingkaran dengan garis vertikal di samping
Control Object	Objek Control	Mengelola logika bisnis dan alur proses dalam sistem	Lingkaran dengan panah melingkar di atas
Entity Object	Objek Entity	Menyimpan data dan merepresentasikan entitas (tabel database, model)	Lingkaran dengan garis horizontal di bawah
Actor	Aktor	Pengguna atau sistem eksternal yang berinteraksi dengan sistem	Stick figure
Association	Asosiasi	Hubungan komunikasi atau aliran informasi antar objek	Garis dengan panah

Menurut [ref:X], Robust Diagram memiliki aturan komunikasi:

1. Actor hanya berkomunikasi dengan Boundary
2. Boundary berkomunikasi dengan Control dan Entity (tidak langsung dengan Boundary lain)
3. Control berkomunikasi dengan Boundary dan Entity
4. Entity hanya berkomunikasi dengan Control (tidak langsung dengan Boundary atau Actor)

Robust Diagram digunakan dalam penelitian ini untuk:

1. Bridge Analysis dan Design: Menghubungkan Use Case Diagram (analisis) dengan Sequence Diagram (desain) dengan mengidentifikasi objek-objek yang diperlukan.
2. Validasi Use Case: Memastikan bahwa setiap use case memiliki Boundary (UI), Control (logic), dan Entity (data) yang lengkap.

3. Desain Arsitektur: Robust Diagram membantu dalam merancang arsitektur MVC (Model-View-Controller) yang sesuai untuk sistem.

2.2.10 Class Diagram

Class Diagram adalah diagram UML yang menggambarkan struktur statis dari sistem dengan menunjukkan class, atribut, method, dan relasi antar class [ref:X]. Class Diagram merupakan diagram yang paling banyak digunakan dalam object-oriented design dan menjadi blueprint untuk implementasi kode [ref:X].

Notasi	Nama	Deskripsi	Simbol
Class	Kelas	Blueprint dari objek yang memiliki atribut dan method	Kotak 3 bagian (nama, atribut, method)
Attribute	Atribut	Properti atau data yang dimiliki oleh class	`visibility name: type`
Method	Method	Fungsi atau operasi yang dapat dilakukan oleh class	`visibility name(params): returnType`
Association	Asosiasi	Hubungan antar class yang menunjukkan relasi	Garis solid dengan nama relasi
Aggregation	Agregasi	Relasi *has-a* lemah; bagian dapat ada tanpa keseluruhan	Garis dengan *diamond* kosong
Composition	Komposisi	Relasi *has-a* kuat; bagian tidak dapat eksis tanpa keseluruhan	Garis dengan *diamond* solid
Inheritance	Pewarisan	Relasi *is-a*; subclass mewarisi atribut dan method superclass	Garis dengan panah segitiga terbuka
Dependency	Ketergantungan	Relasi dimana suatu class menggunakan class lain	Garis putus-putus dengan panah
Multiplicity	Multiplisitas	Menyatakan kardinalitas relasi (1, 0..1, 1.., *0..*)	Angka atau range di ujung garis relasi

Visibility Modifier:

- + Public: dapat diakses dari mana saja
- - Private: hanya dapat diakses dari dalam class
- # Protected: dapat diakses dari class dan subclass
- ~ Package: dapat diakses dari package yang sama

Class Diagram digunakan dalam penelitian ini untuk:

1. Memodelkan Struktur Data: Menggambarkan entity class seperti User, Proposal, Review, Seminar, Disbursement, Publication dengan atribut dan method.
2. Menunjukkan Relasi: Menggambarkan relasi one-to-many, many-to-many antara entity (misalnya: User has many Proposals, Proposal has many Reviews).
3. Blueprint Implementasi: Class Diagram menjadi panduan langsung untuk menulis kode Prisma Schema dan TypeScript interfaces.
4. Dokumentasi Struktur: Class Diagram menjadi dokumentasi teknis yang menjelaskan struktur data sistem kepada developer lain.

2.2.11 Entity Relationship Diagram (ERD)

Entity Relationship Diagram (ERD) adalah diagram yang digunakan untuk memodelkan struktur database relasional dengan menunjukkan entity (tabel), atribut (kolom), dan relasi antar entity [ref:X]. ERD dikembangkan oleh Peter Chen pada tahun 1976 dan menjadi standar untuk database design [ref:X].

Notasi	Nama	Deskripsi	Simbol
Entity	Entitas	Objek atau konsep yang datanya disimpan dalam database	Persegi panjang
Attribute	Atribut	Properti atau karakteristik dari sebuah entitas	Oval / ellipse
Primary Key	Kunci Primer	Atribut unik sebagai identitas tiap record	Oval dengan garis bawah
Multi-valued Attribute	Atribut Multi-nilai	Atribut yang dapat memiliki lebih dari satu nilai	Oval ganda
Derived Attribute	Atribut Turunan	Atribut yang diperoleh dari atribut lain (hasil perhitungan)	Oval putus-putus
Relationship	Relasi	Hubungan antar dua atau lebih entitas	Diamond (belah ketupat)
Cardinality	Kardinalitas	Jumlah instance dalam relasi (1:1, 1:N, M:N)	Notasi angka atau crow's foot

Weak Entity	Entitas Lemah	Entitas yang identitasnya bergantung pada entitas lain	Persegi panjang ganda

Kardinalitas Relasi:

- One-to-One (1:1): Satu instance entity A berelasi dengan satu instance entity B
- One-to-Many (1:N): Satu instance entity A berelasi dengan banyak instance entity B
- Many-to-Many (M:N): Banyak instance entity A berelasi dengan banyak instance entity B

ERD digunakan dalam penelitian ini untuk:

1. Desain Database: Merancang struktur tabel database MySQL untuk sistem LPPM dengan relasi yang jelas.
2. Normalisasi: Memastikan database dinormalisasi untuk mengurangi redundansi dan meningkatkan integritas data.
3. Blueprint Prisma Schema: ERD menjadi panduan untuk menulis Prisma Schema dengan relasi yang tepat.
4. Dokumentasi Struktur Data: ERD menjadi dokumentasi database yang mudah dipahami oleh database administrator atau developer lain.

2.2.12 Arsitektur Sistem Web Modern

Three-Tier Architecture adalah pola arsitektur perangkat lunak yang memisahkan aplikasi menjadi tiga lapisan (tier) yang berbeda untuk meningkatkan modularitas, scalability, dan maintainability [ref:X]. Ketiga lapisan tersebut adalah:

1. Presentation Tier (Lapisan Presentasi): Lapisan yang bertanggung jawab untuk user interface dan interaksi dengan pengguna. Dalam sistem LPPM ini, presentation tier berupa React components di Next.js yang di-render di browser.
2. Application Tier (Lapisan Aplikasi/Logic): Lapisan yang berisi business logic dan processing data. Dalam Next.js, ini berupa API Routes dan Server Components yang menangani validasi, authorization, dan orchestration.
3. Data Tier (Lapisan Data): Lapisan yang bertanggung jawab untuk penyimpanan dan retrieval data. Dalam sistem ini, data tier berupa MySQL database yang diakses melalui Prisma ORM.

Keuntungan Three-Tier Architecture:

- Separation of Concerns: Setiap tier memiliki tanggung jawab yang jelas, memudahkan development dan testing.
- Scalability: Setiap tier dapat di-scale secara independen sesuai kebutuhan (misalnya menambah server database tanpa mengubah application tier).
- Maintainability: Perubahan pada satu tier tidak mempengaruhi tier lainnya selama interface/contract tetap sama.
- Reusability: Business logic di application tier dapat digunakan oleh berbagai presentation tier (web, mobile, API).

Model-View-Controller (MVC) Pattern

MVC adalah pola desain arsitektur yang memisahkan aplikasi menjadi tiga komponen utama [ref:X]:

1. Model: Mewakili data dan business logic aplikasi (dalam sistem ini: Prisma models dan service layer).
2. View: Mewakili user interface yang menampilkan data kepada pengguna (dalam sistem ini: React components).
3. Controller: Menangani input dari user dan mengatur komunikasi antara Model dan View (dalam sistem ini: API Routes dan event handlers).

Next.js secara natural mendukung MVC pattern dengan struktur yang jelas antara React components (View), API Routes (Controller), dan Prisma models (Model).

RESTful API Principles

REST (Representational State Transfer) adalah arsitektur untuk membangun web services yang scalable dan maintainable [ref:X]. RESTful API memiliki prinsip-prinsip berikut:

1. Stateless: Setiap request dari client harus mengandung semua informasi yang diperlukan server, server tidak menyimpan state client.
2. Resource-Based: API diorganisir berdasarkan resource (misalnya /api/proposals, /api/users) bukan action.
3. HTTP Methods: Menggunakan HTTP methods standar (GET untuk read, POST untuk create, PUT/PATCH untuk update, DELETE untuk delete).
4. JSON Format: Data dipertukarkan dalam format JSON yang lightweight dan mudah di-parse.
5. Status Codes: Menggunakan HTTP status codes untuk mengindikasikan hasil request (200 OK, 201 Created, 400 Bad Request, 401 Unauthorized, 404 Not Found, 500 Internal Server Error).

Sistem LPPM dirancang dengan Three-Tier Architecture yang jelas:

- Presentation: Next.js React components dengan responsive design
- Application: API Routes untuk business logic (authentication, validation, notification)
- Data: MySQL database dengan Prisma ORM

API dirancang mengikuti RESTful principles untuk memudahkan integrasi di masa depan dan memastikan konsistensi dalam pengelolaan resource (proposals, reviews, seminars, publications).

2.2.13 Next.js Framework

Next.js adalah framework full-stack berbasis React yang dikembangkan oleh Vercel untuk membangun aplikasi web modern dengan performa tinggi [ref:X]. Next.js menyediakan fitur-fitur seperti Server-Side Rendering (SSR), Static Site Generation (SSG), API Routes, dan optimasi otomatis yang menjadikannya pilihan populer untuk pengembangan aplikasi web kompleks [ref:X]. Next.js pertama kali dirilis pada tahun 2016 dan telah menjadi salah satu framework React paling banyak digunakan dengan lebih dari 1 juta website yang dibangun menggunakan teknologi ini [ref:X].

Next.js menggunakan arsitektur yang fleksibel dengan beberapa komponen utama [ref:X]:

1. App Router (Next.js 13+): Sistem routing berbasis file system yang mendukung React Server Components, nested layouts, dan streaming SSR untuk performa optimal.
2. API Routes: Memungkinkan pembuatan backend API endpoint langsung dalam project Next.js tanpa perlu server terpisah, sehingga mendukung full-stack development dalam satu codebase.
3. Server Components vs Client Components: Next.js membedakan komponen yang dirender di server (default) dan komponen yang memerlukan interaktivitas client-side, sehingga mengurangi JavaScript bundle size.
4. Middleware: Memungkinkan eksekusi kode sebelum request selesai diproses, berguna untuk autentikasi, logging, dan redirect.

Next.js mendukung beberapa strategi rendering yang dapat dipilih sesuai kebutuhan [ref:X]:

1. Server-Side Rendering (SSR): Halaman di-render di server setiap kali ada request, cocok untuk konten yang dinamis dan memerlukan data real-time. SSR meningkatkan SEO karena content sudah tersedia saat crawler search engine mengakses halaman.
2. Static Site Generation (SSG): Halaman di-render pada saat build time dan disajikan sebagai file HTML statis, cocok untuk konten yang jarang berubah. SSG memberikan performa terbaik karena tidak perlu rendering saat runtime.

3. Incremental Static Regeneration (ISR): Kombinasi SSG dengan kemampuan regenerasi halaman secara berkala tanpa perlu rebuild seluruh aplikasi, cocok untuk konten yang berubah periodik.
4. Client-Side Rendering (CSR): Halaman di-render di browser menggunakan JavaScript, cocok untuk dashboard atau aplikasi interaktif yang memerlukan data real-time dari API.

Menurut [ref:X], Next.js memiliki beberapa keunggulan dibandingkan framework lain:

1. Performa Tinggi: Optimasi otomatis seperti code splitting, image optimization, dan font optimization menghasilkan aplikasi yang sangat cepat. Penelitian menunjukkan aplikasi Next.js 30-40% lebih cepat dibanding React tradisional [ref:X].
2. Developer Experience: Hot reload yang cepat, error reporting yang jelas, dan TypeScript support out-of-the-box meningkatkan produktivitas developer.
3. SEO-Friendly: SSR dan SSG membuat konten dapat di-crawl oleh search engine dengan baik, meningkatkan ranking SEO.
4. Full-Stack Capability: API Routes memungkinkan pengembangan backend dan frontend dalam satu project dengan bahasa yang sama (JavaScript/TypeScript).
5. Scalability: Built-in support untuk deployment ke platform serverless (Vercel, AWS Lambda) maupun traditional server (VPS, Docker).

Dalam penelitian ini, Next.js dipilih sebagai framework utama karena:

1. Full-Stack Development: API Routes memungkinkan pengembangan backend (pengelolaan database, authentication, business logic) dan frontend (UI, dashboard) dalam satu codebase, cocok untuk pengembang individu.
2. SSR untuk Dashboard: Dashboard monitoring untuk pimpinan menggunakan SSR untuk memastikan data selalu up-to-date dan performa rendering yang cepat.
3. Type Safety dengan TypeScript: Next.js terintegrasi sempurna dengan TypeScript, sehingga mendukung type safety end-to-end dari API hingga UI.
4. Performance: Sistem LPPM memerlukan performa yang baik untuk menangani upload dokumen, rendering tabel data penelitian, dan visualisasi chart pada dashboard.

5. Developer Experience: Hot reload dan error reporting yang baik mempercepat development cycle dalam metodologi PXP yang menekankan iterasi cepat.

2.2.14 Prisma ORM

Object-Relational Mapping (ORM) adalah teknik pemrograman yang memungkinkan developer untuk berinteraksi dengan database menggunakan paradigma object-oriented tanpa perlu menulis raw SQL query [ref:X]. ORM memetakan tabel database menjadi class/object dalam kode, sehingga operasi database (CRUD) dapat dilakukan dengan sintaks yang lebih intuitif dan type-safe [ref:X].

Prisma adalah ORM generasi baru (next-generation ORM) yang dirancang khusus untuk TypeScript dan Node.js [ref:X]. Berbeda dengan ORM tradisional (seperti Sequelize atau TypeORM), Prisma menyediakan pengalaman developer yang superior dengan pendekatan declarative schema, type-safe query builder, dan migration system yang powerful [ref:X].

Prisma terdiri dari tiga komponen utama [ref:X]:

1. Prisma Schema: File deklaratif (schema.prisma) yang mendefinisikan model database, relasi antar tabel, dan konfigurasi koneksi database. Schema ini menjadi single source of truth untuk struktur database.
2. Prisma Client: Auto-generated query builder yang type-safe berdasarkan Prisma Schema. Setiap kali schema berubah, Prisma Client di-regenerate untuk menyediakan autocomplete dan type checking yang akurat.
3. Prisma Migrate: Tool untuk mengelola database schema migration dengan versioning yang jelas. Migrate memastikan perubahan schema dapat di-track dan di-apply secara konsisten di berbagai environment (development, staging, production).

Menurut [ref:X], Prisma memiliki beberapa keunggulan:

1. Type Safety End-to-End: Setiap query yang ditulis menggunakan Prisma Client akan di-type-check oleh TypeScript, sehingga error dapat terdeteksi pada tahap development, bukan runtime.
2. Auto-Completion: IDE (VSCode) memberikan autocomplete untuk semua field, relasi, dan method Prisma Client, meningkatkan produktivitas dan mengurangi typo.
3. Declarative Schema: Schema database ditulis dalam format yang mudah dibaca dan dipahami, tanpa perlu menulis migration manual yang kompleks.

4. Database Agnostic: Prisma mendukung berbagai database (PostgreSQL, MySQL, SQLite, SQL Server, MongoDB) dengan API yang konsisten, memudahkan migrasi antar database.
5. Performance: Prisma mengoptimasi query secara otomatis dengan batching, caching, dan lazy loading untuk mengurangi jumlah query ke database.
6. Developer Experience: Error message yang jelas, documentation yang lengkap, dan Prisma Studio (GUI untuk explore database) meningkatkan pengalaman developer.

Dalam penelitian ini, Prisma dipilih sebagai ORM karena:

1. Type Safety: Integrasi dengan TypeScript memastikan setiap operasi database di-type-check, mengurangi runtime error pada sistem yang kompleks seperti LPPM.
2. Complex Relations: Sistem LPPM memiliki relasi kompleks (User-Proposal-Review-Seminar-Disbursement-Publication), dan Prisma menyediakan API yang intuitif untuk query relasi tersebut.
3. Migration Management: Prisma Migrate memudahkan evolusi schema database seiring dengan perubahan requirement di setiap iterasi PXP.
4. Developer Productivity: Autocomplete dan error detection meningkatkan kecepatan development, cocok untuk metodologi PXP yang menekankan iterasi cepat.
5. Prisma Studio: GUI untuk explore dan edit data database memudahkan debugging dan testing selama development.

2.2.15 TypeScript

TypeScript adalah superset dari JavaScript yang menambahkan static typing dan fitur-fitur modern lainnya ke dalam bahasa JavaScript [ref:X]. TypeScript dikembangkan oleh Microsoft dan dirilis pertama kali pada tahun 2012, sejak itu telah menjadi standar de facto untuk pengembangan aplikasi JavaScript skala besar [ref:X]. TypeScript di-compile menjadi JavaScript biasa, sehingga dapat berjalan di semua environment yang mendukung JavaScript (browser, Node.js, dll).

TypeScript menyediakan type system yang powerful dengan fitur-fitur berikut [ref:X]:

1. Static Typing: Setiap variabel, parameter function, dan return value dapat diberi tipe data, yang akan di-check pada tahap kompilasi.
2. Type Inference: TypeScript dapat menginfer (menebak) tipe data secara otomatis berdasarkan nilai yang diberikan, sehingga tidak selalu perlu eksplisit menulis tipe.

3. Union dan Intersection Types: Memungkinkan tipe data yang kompleks dengan kombinasi beberapa tipe (contoh: string | number).
4. Generics: Membuat function atau class yang dapat bekerja dengan berbagai tipe data dengan tetap menjaga type safety.
5. Interface dan Type Aliases: Mendefinisikan struktur object yang kompleks dengan dokumentasi tipe yang jelas.

Menurut [ref:X], TypeScript memberikan beberapa manfaat signifikan:

1. Error Prevention: Sebagian besar error (typo, wrong type, undefined property) dapat terdeteksi pada tahap development, bukan saat runtime di production.
2. IDE Support: Autocomplete, refactoring tools, dan inline documentation bekerja dengan sangat baik karena IDE memahami tipe data setiap variabel.
3. Maintainability: Kode dengan tipe eksplisit lebih mudah dipahami dan di-maintain, terutama untuk project besar dengan banyak file.
4. Refactoring Confidence: Refactoring kode menjadi lebih aman karena TypeScript akan memberikan error jika ada breaking change yang mempengaruhi bagian lain dari kode.
5. Documentation: Tipe data berfungsi sebagai dokumentasi yang selalu up-to-date dan tidak bisa "bohong" seperti komentar.

Dalam konteks full-stack development dengan Next.js dan Prisma, TypeScript memberikan keuntungan tambahan [ref:X]:

1. End-to-End Type Safety: Tipe data dari database (Prisma) dapat mengalir hingga ke frontend (React components) tanpa kehilangan informasi tipe, sehingga seluruh aplikasi type-safe.
2. API Contract: Tipe data untuk request dan response API dapat didefinisikan dengan jelas, memastikan frontend dan backend selalu sinkron.
3. Shared Types: Tipe data yang sama dapat digunakan di frontend dan backend (karena keduanya menggunakan TypeScript), mengurangi duplikasi dan inkonsistensi.

Dalam penelitian ini, TypeScript dipilih karena:

1. Quality Assurance: Type safety mengurangi bug, mendukung praktik TDD dalam metodologi PXP dengan mendeteksi error lebih awal.
2. Complex Domain Model: Sistem LPPM memiliki banyak entity (User, Proposal, Review, Seminar, Disbursement, Publication) dengan relasi kompleks, TypeScript memastikan konsistensi tipe data.

3. Maintainability: Kode TypeScript lebih mudah dipahami dan di-maintain oleh developer lain (untuk technical documentation), mendukung tujuan penelitian untuk menghasilkan sistem yang dapat di-maintain.
4. Integration dengan Next.js dan Prisma: Next.js dan Prisma keduanya mendukung TypeScript secara native, sehingga memberikan pengalaman development yang seamless.

2.2.16 MySQL

MySQL adalah sistem manajemen basis data relasional (RDBMS) open-source yang paling populer di dunia [ref:X]. MySQL dikembangkan oleh MySQL AB (sekarang bagian dari Oracle Corporation) dan pertama kali dirilis pada tahun 1995 [ref:X]. MySQL menggunakan SQL (Structured Query Language) sebagai bahasa untuk mengelola dan query data, serta mendukung berbagai storage engine seperti InnoDB dan MyISAM.

MySQL memiliki beberapa karakteristik utama [ref:X]:

1. Relational Database: Data disimpan dalam tabel-tabel yang saling berelasi melalui foreign key, mendukung normalisasi untuk mengurangi redundansi data.
2. ACID Compliance: MySQL dengan InnoDB storage engine mendukung properti ACID (Atomicity, Consistency, Isolation, Durability) untuk memastikan integritas transaksi.
3. Scalability: MySQL dapat menangani database dengan ukuran sangat besar (terabytes) dan jumlah query yang tinggi dengan optimasi yang tepat.
4. Performance: MySQL terkenal dengan performa read/write yang cepat, terutama untuk aplikasi web dengan traffic tinggi.
5. Open Source: MySQL tersedia secara gratis dengan lisensi GPL, sehingga cocok untuk project dengan budget terbatas seperti institusi pendidikan.

Menurut [ref:X], MySQL memiliki kelebihan untuk aplikasi sistem informasi:

1. Mature dan Reliable: MySQL telah digunakan oleh jutaan aplikasi selama lebih dari 25 tahun dengan track record reliability yang baik.
2. Community Support: Dokumentasi lengkap dan komunitas yang besar memudahkan troubleshooting dan learning.
3. Integration: MySQL mudah diintegrasikan dengan berbagai bahasa pemrograman dan framework, termasuk Node.js dan Prisma.
4. Backup dan Recovery: MySQL menyediakan tools untuk backup (mysqldump) dan recovery yang mudah digunakan.

Normalisasi adalah proses mengorganisir data dalam database untuk mengurangi redundansi dan meningkatkan integritas data [ref:X]. Dalam sistem LPPM, normalisasi diterapkan untuk memastikan data penelitian, user, dan publikasi tersimpan secara efisien tanpa duplikasi.

Dalam penelitian ini, MySQL dipilih sebagai DBMS karena:

1. Compatibility dengan Prisma: Prisma mendukung MySQL dengan baik, termasuk migration dan type generation.
2. Relational Model: Data LPPM (User, Proposal, Review, Seminar, Disbursement, Publication) memiliki relasi yang kompleks, sehingga cocok dengan relational database.
3. Performance: MySQL dapat menangani query kompleks (join multiple tables, aggregation) dengan performa yang baik untuk jumlah data yang dikelola LPPM STAI Ali bin Abi Thalib.
4. Availability: MySQL mudah di-install dan dikonfigurasi di VPS untuk deployment sistem.
5. Backup Strategy: MySQL menyediakan tools backup yang mudah digunakan untuk memastikan data penelitian tidak hilang.

2.2.17 Git Dan Github

Git adalah distributed version control system yang memungkinkan developer untuk tracking perubahan kode, berkolaborasi, dan mengelola versi software [ref:X]. Git dikembangkan oleh Linus Torvalds pada tahun 2005 dan menjadi standar industri untuk version control [ref:X].

Fitur Utama Git:

1. Branching dan Merging: Membuat branch untuk fitur baru dan merge kembali ke main branch
2. Commit History: Tracking setiap perubahan kode dengan commit message
3. Revert dan Reset: Kemampuan untuk kembali ke versi sebelumnya jika terjadi error
4. Distributed: Setiap developer memiliki copy lengkap repository di lokal

GitHub adalah platform hosting untuk Git repository yang menyediakan fitur kolaborasi, issue tracking, dan CI/CD melalui GitHub Actions [ref:X].

- Continuous Integration (CI) adalah praktik untuk merge perubahan kode ke repository utama secara frequent dengan automated testing untuk mendeteksi bug sedini mungkin [ref:X].

- Continuous Deployment (CD) adalah praktik untuk deploy setiap perubahan kode yang lolos testing ke production secara otomatis [ref:X].

GitHub Actions adalah platform CI/CD yang terintegrasi dengan GitHub untuk automate workflow [ref:X]. Dalam penelitian ini, GitHub Actions digunakan untuk:

1. Automated Testing: Setiap kali ada push atau pull request, GitHub Actions akan run unit test (Jest) dan integration test secara otomatis.
2. Code Quality Check: Menjalankan linter (ESLint), type checker (TypeScript), dan code formatter (Prettier).
3. Build Verification: Memastikan aplikasi dapat di-build tanpa error sebelum merge ke main branch.
4. Automated Deployment: Setelah semua test pass, GitHub Actions akan deploy aplikasi ke VPS secara otomatis.

Git dan CI/CD digunakan dalam penelitian ini untuk:

1. Version Control: Tracking setiap perubahan kode selama 4 iterasi PXP dengan commit message yang jelas.
2. Quality Assurance: Automated testing via GitHub Actions memastikan setiap perubahan kode tidak break existing functionality (regression testing).
3. Rapid Iteration: CI/CD mempercepat deployment setiap akhir iterasi, sehingga stakeholder dapat langsung testing versi terbaru.
4. Dokumentasi Development: Git history menjadi dokumentasi proses development yang menunjukkan evolusi sistem dari iterasi 1 hingga 4.
5. Praktik Modern Engineering: Menerapkan CI/CD sesuai dengan best practice PXP untuk continuous integration.

2.2.18 Black Box Testing

Black Box Testing adalah metode pengujian perangkat lunak yang fokus pada fungsionalitas sistem tanpa memperhatikan struktur internal atau logika kode [ref:X]. Tester melakukan testing berdasarkan spesifikasi requirement dengan memberikan input dan memvalidasi output yang dihasilkan [ref:X].

Karakteristik Black Box Testing:

- Tester tidak perlu mengetahui source code
- Fokus pada "what the system does" bukan "how it does it"
- Menguji dari perspektif end-user
- Dapat dilakukan oleh non-programmer

Teknik Black Box Testing

1. Equivalence Partitioning: Membagi input menjadi kelompok-kelompok (partisi) yang dianggap equivalent, dan testing dilakukan dengan memilih satu representatif dari setiap partisi [ref:X].
   Contoh untuk field "Judul Proposal":
   - Partisi valid: string 10-200 karakter
   - Partisi invalid: string kosong, string < 10 karakter, string > 200 karakter
2. Boundary Value Analysis: Menguji nilai-nilai pada batas (boundary) dari partisi, karena bug sering terjadi di edge cases [ref:X].
   Contoh untuk field "Judul Proposal" (10-200 karakter):
   - Test: 9 karakter (invalid), 10 karakter (valid), 11 karakter (valid)
   - Test: 199 karakter (valid), 200 karakter (valid), 201 karakter (invalid)
3. Decision Table Testing: Membuat tabel yang menggambarkan kombinasi kondisi dan aksi yang sesuai [ref:X].
   Contoh untuk "Submit Proposal":
   - Kondisi: User role, Proposal status, Dokumen lengkap
   - Aksi: Submit berhasil / Submit gagal dengan error message

Black Box Testing digunakan dalam penelitian ini untuk:

1. Validasi Fungsionalitas: Memastikan setiap fitur (pengajuan proposal, review, monitoring, pencairan dana, seminar, publikasi) berfungsi sesuai requirement.
2. Test Case Design: Membuat test case dengan equivalence partitioning dan boundary value analysis untuk semua form input (judul proposal, upload file, penilaian review, dll).
3. User Perspective: Menguji sistem dari perspektif setiap role (Admin LPPM, Dosen, Reviewer, Pimpinan) untuk memastikan workflow sesuai dengan Buku Panduan Penelitian.
4. Acceptance Criteria: Black Box test cases menjadi acceptance criteria untuk setiap user story dalam iterasi PXP.

2.2.19 White Box Testing

White Box Testing adalah metode pengujian perangkat lunak yang fokus pada struktur internal, logika kode, dan path execution [ref:X]. Tester harus memahami source code dan menulis test case untuk memvalidasi bahwa setiap path, branch, dan condition dalam kode berfungsi dengan benar [ref:X].

Karakteristik White Box Testing:

- Tester harus mengetahui dan memahami source code
- Fokus pada "how the system works"
- Menguji logika internal program
- Memerlukan pengetahuan programming

Teknik White Box Testing

1. Statement Coverage: Memastikan setiap statement (baris kode) dieksekusi minimal satu kali dalam testing [ref:X].
   Coverage = (Number of statements executed / Total statements) × 100%
2. Branch Coverage (Decision Coverage): Memastikan setiap branch dari decision point (if-else, switch) dieksekusi minimal satu kali [ref:X].
   Coverage = (Number of branches executed / Total branches) × 100%
3. Path Coverage: Memastikan setiap possible path dari entry point hingga exit point dieksekusi [ref:X].
4. Coverage: Memastikan setiap boolean sub-expression dievaluasi menjadi true dan false minimal satu kali [ref:X].

Test Case untuk 100% Branch Coverage:

1. Test dengan proposalId tidak ditemukan → Branch 1 true
2. Test dengan user role bukan ADMIN_LPPM → Branch 2 true
3. Test dengan proposal status bukan UNDER_REVIEW → Branch 3 true
4. Test dengan semua kondisi valid → Semua branch false, eksekusi statement 6-9

White Box Testing digunakan dalam penelitian ini untuk:

1. Code Quality Assurance: Memastikan setiap logika business (validation, authorization, state transition) diuji dengan coverage yang tinggi.
2. Test-Driven Development (TDD): Dalam PXP, unit test ditulis sebelum implementasi kode (TDD), sehingga white box testing menjadi bagian integral dari development.
3. Critical Path Testing: Memastikan path kritis (seperti approval workflow, pencairan dana, notifikasi) diuji dengan thorough.
4. Regression Testing: White box test yang otomatis dapat dijalankan setiap kali ada perubahan kode (via CI/CD) untuk memastikan tidak ada bug baru yang muncul.

Tool untuk White Box Testing:

- Jest: JavaScript testing framework untuk unit testing dengan coverage report
- Istanbul/nyc: Code coverage tool yang terintegrasi dengan Jest
- SonarQube: Static code analysis untuk mendeteksi bug dan code smell

2.2.20 User Acceptance Testing (UAT)

User Acceptance Testing (UAT) adalah fase terakhir dari software testing dimana sistem diuji oleh end-user atau stakeholder untuk memvalidasi bahwa sistem memenuhi kebutuhan bisnis dan siap untuk deployment ke production [ref:X]. UAT fokus pada validasi bahwa sistem dapat digunakan dalam skenario nyata dan memenuhi ekspektasi pengguna [ref:X].

Tujuan UAT:

1. Memvalidasi bahwa sistem memenuhi business requirement
2. Memastikan sistem dapat digunakan (usable) oleh end-user
3. Mengidentifikasi bug atau issue yang terlewat di testing sebelumnya
4. Mendapatkan feedback untuk improvement
5. Mendapatkan sign-off dari stakeholder sebelum go-live

Jenis UAT

1. Alpha Testing: UAT yang dilakukan di environment developer dengan kehadiran developer untuk observasi dan immediate fix.
2. Beta Testing: UAT yang dilakukan di environment production atau staging tanpa kehadiran developer, user melaporkan bug jika ditemukan.
3. Contract Acceptance Testing: UAT berdasarkan kriteria yang didefinisikan dalam kontrak antara developer dan client.

Proses UAT

1. Persiapan: Menyiapkan UAT plan, test scenarios, dan test data
2. Pelaksanaan: User melakukan testing berdasarkan skenario nyata (real-world usage)
3. Dokumentasi: Mencatat bug, issue, dan feedback dari user
4. Evaluasi: Menganalisis hasil UAT dan menentukan apakah sistem acceptable
5. Sign-off: Stakeholder memberikan approval jika sistem memenuhi kriteria acceptance

Metode Penilaian UAT

1. System Usability Scale (SUS): Kuesioner standar dengan 10 pertanyaan menggunakan skala Likert 1-5 untuk mengukur usability sistem [ref:X].

Skor SUS dihitung dengan formula tertentu dan menghasilkan nilai 0-100:

- Skor < 50: Poor
- Skor 50-70: OK/Fair
- Skor 70-85: Good
- Skor > 85: Excellent

2. Likert Scale Custom Questionnaire: Kuesioner custom dengan pertanyaan spesifik untuk sistem menggunakan skala 1-5 (Sangat Tidak Setuju - Sangat Setuju).

UAT digunakan dalam penelitian ini untuk:

1. Validasi Penerimaan Pengguna: Melibatkan Admin LPPM, Dosen/Peneliti, Reviewer, dan Pimpinan untuk memvalidasi bahwa sistem memenuhi kebutuhan masing-masing role.
2. Real-World Testing: Menguji sistem dengan skenario nyata sesuai Buku Panduan Penelitian STAI Ali bin Abi Thalib (pengajuan proposal, seminar, monitoring, dll).
3. Usability Evaluation: Mengukur usability sistem menggunakan kuesioner (SUS atau Likert scale) dengan target skor minimal 70% untuk dinyatakan layak.
4. Feedback untuk Improvement: Mengumpulkan feedback dari stakeholder untuk perbaikan sebelum deployment final.

Responden UAT dalam Penelitian Ini:

- 1 orang Admin LPPM (pengguna utama semua fitur)
- 2 orang Dosen/Peneliti (pengajuan, monitoring, publikasi)
- 1 orang Reviewer (review proposal)
- 1 orang Pimpinan (dashboard monitoring)
  Total: 5 responden

2.3 Alasan Pemilihan Metode dan Teknologi

2.3.1 Pemilihan Personal Extreme Programming (PXP)

Berdasarkan analisis terhadap berbagai metodologi pengembangan perangkat lunak (Tabel 2.2), PXP dipilih sebagai metodologi dalam penelitian ini karena beberapa alasan:

Kriteria	Waterfall	RAD	Scrum	XP	PXP (Personal XP)	Keterangan

Konteks Pengembang	Tim 5–50	Tim 3–10	Tim 5–9	Tim 4–12	Individu 1–3	✓ Tugas akhir dikerjakan individu
Fleksibilitas Requirement	Sangat rendah	Sedang	Tinggi	Sangat tinggi	Sangat tinggi	✓ Requirement dapat berubah berdasarkan feedback stakeholder
Quality Assurance	End-phase (akhir)	Iterative	Sprint-based	TDD + CI	TDD + CI/CD	✓ Perlu quality assurance meskipun individu
Learning Curve	Sedang	Sedang	Tinggi	Tinggi	Rendah –Sedang	✓ Cocok untuk mahasiswa dengan waktu terbatas
Documentation	Sangat lengkap	Minimal	Moderate	Minimal	Code + User/Tech Docs	✓ Perlu dokumentasi untuk TA + maintenance
Stakeholder Involvement	Awal & akhir	Moderate	Tiap sprint	Continuous	Demo/review per iterasi	✓ Stakeholder LPPM dapat

						terlibat berkala
Sustainable Pace	Tidak enforce	Tidak enforce	Recommended	Recommended	**Wajib**	✓ Mahasiswa perlu work-life balance

2.3.2 Pemilihan Next.js + Prisma + TypeScript

Aspek	Next.js + Prisma + TypeScript	Laravel + React	MERN Stack	Django + Vue
Full-Stack dalam 1 Codebase	✓ (Next.js API Routes)	✗ (Backend – Frontend terpisah)	✗ (Express terpisah)	✗ (Terpisah)
Type Safety End-to-End	✓✓✓ (TypeScript + Prisma)	△ (Partial, mostly frontend)	△ (TypeScript optional)	△ (Python typed opsional)
Developer Experience	✓✓✓ (Hot reload, autocomplete)	✓✓ (Baik)	✓✓ (Baik)	✓✓ (Baik)
Performance	✓✓✓ (SSR/SSG, optimasi otomatis)	✓✓ (Server-side)	✓ (Client-side rendering)	✓✓ (Server-side)
Learning Curve	Sedang (JavaScript/TS)	Sedang–Tinggi (PHP + JS)	Rendah–Sedang (JS Full)	Tinggi (Python + JS)
Community & Ecosystem	✓✓✓ (Sangat besar)	✓✓✓ (Sangat besar)	✓✓✓ (Sangat besar)	✓✓ (Besar)
Database Migration	✓✓✓ (Prisma Migrate)	✓✓✓ (Laravel Migration)	△ (Manual/Mongoose)	✓✓✓ (Django Migration)
Testing Support	✓✓✓ (Jest, RTL)	✓✓✓ (PHPUnit, Jest)	✓✓✓ (Jest, Mocha)	✓✓✓ (Pytest, Jest)

Deployment	✓✓✓ (Vercel, VPS, Docker)	✓✓ (VPS, shared hosting)	✓✓ (VPS, cloud)	✓✓ (VPS, cloud)
Cocok untuk PXP (Individu)	✓✓✓	✓✓	✓✓	✓✓
Cocok untuk Sistem LPPM	✓✓✓	✓✓	✓✓	✓✓

Justifikasi Pemilihan Next.js + Prisma + TypeScript:

1. Full-Stack dalam 1 Codebase: Next.js API Routes memungkinkan pengembangan backend dan frontend dalam satu project dengan bahasa yang sama (TypeScript), mengurangi complexity untuk pengembang individu.
2. Type Safety End-to-End: TypeScript + Prisma menyediakan type safety dari database hingga UI, mengurangi runtime error dan meningkatkan confidence dalam kode. Sangat penting untuk quality assurance dalam PXP.
3. Developer Productivity: Autocomplete, type checking, hot reload, dan error detection meningkatkan produktivitas, cocok untuk iterasi cepat dalam PXP (1-2 minggu per iterasi).
4. Performance: SSR dan SSG di Next.js meningkatkan performa aplikasi, terutama untuk dashboard monitoring yang memerlukan data real-time.
5. Modern Best Practices: Stack ini mendukung best practices modern seperti TDD (Jest), CI/CD (GitHub Actions), dan code quality (ESLint, Prettier).
6. Prisma Advantage: Prisma Migrate memudahkan evolusi schema database di setiap iterasi PXP tanpa perlu manual migration yang error-prone.
7. Community Support: Next.js, Prisma, dan TypeScript memiliki dokumentasi lengkap dan komunitas yang besar, memudahkan learning dan troubleshooting.

2.3.3 Pemilihan MySQL

MySQL dipilih sebagai database management system karena:

1. Compatibility: Prisma mendukung MySQL dengan baik, termasuk semua fitur relational (foreign key, constraint, transaction).

2. Relational Model: Data LPPM (User, Proposal, Review, Seminar, Disbursement, Publication) memiliki relasi kompleks yang cocok dengan relational database.
3. Performance: MySQL dapat menangani query join kompleks dan aggregation dengan performa baik untuk skala data LPPM STAI Ali bin Abi Thalib.
4. Mature dan Stable: MySQL telah teruji selama 25+ tahun dengan reliability yang sangat baik.
5. VPS Friendly: MySQL mudah di-install dan dikonfigurasi di VPS dengan resource management yang efisien.
6. Backup Strategy: MySQL menyediakan tools backup (mysqldump) yang mudah untuk disaster recovery.

Kriteria	MySQL	PostgreSQL	MongoDB	SQLite
Paradigm	Relational	Relational	NoSQL (Document)	Relational
ACID Compliance	✓ (InnoDB)	✓	△ (Tunable, tergantung config)	✓
Performance (Read/Write)	Excellent	Excellent	Excellent (horizontal scaling)	Good
Complex Queries (Join)	✓✓✓	✓✓✓	△ (Tidak mendukung join native)	✓✓
Scalability	High	Very High	Very High	Low
Prisma Support	✓✓✓	✓✓✓	✓✓	✓✓✓
VPS Deployment	Easy	Easy	Easy	Embedded (tidak perlu server)
Data Integrity	✓✓✓ (Foreign key, constraint)	✓✓✓ (Strong constraints)	△ (Tidak ada foreign key)	✓✓✓
Cocok untuk Sistem LPPM	✓✓✓	✓✓✓	✗ (Butuh relasi yang kuat)	✗ (Tidak cocok production)

MySQL dan PostgreSQL keduanya cocok untuk sistem LPPM, tetapi MySQL dipilih karena lebih familiar, dokumentasi Indonesia lebih banyak, dan performa read-heavy yang sedikit lebih baik (cocok untuk dashboard monitoring.

BAB III
METODOLOGI PENELITIAN

DAFTAR PUSTAKA

A Sumarudin, Amirrudin, & Suheryadi, A. (2021). Penerapan Sistem Informasi Penelitian Internal Di Politeknik Negeri Indramayu Menggunakan Metode Kanban. *JITSI : Jurnal Ilmiah Teknologi Sistem Informasi*, *2*(4), 103–107. https://doi.org/10.30630/jitsi.2.4.42

Admaja, M. H., & Pramono, A. (2025). *PENGEMBANGAN LEARNING*

MANAGEMENT SYSTEM (LMS) MENGGUNAKAN PERSONAL EXTREME PROGRAMMING (PXP) PADA SMP NEGERI 1 KEDUNGADEM. *9*(2), 3484–3492.

Firmansyah, M. A., Ramsari, N., & Rachmanto, A. D. (2022). Rancang Bangun Sistem Informasi Penjualan Pada Toko Buku Kita Tasikmalaya Berbasis Web Menggunakan Framework Laravel 8. *Jurnal Teknologi Informasi Dan Komunikasi*, *12*(1). https://doi.org/10.56244/fiki.v12i1.498

Gede Endra Bratha, W. (2022). Literature Review Komponen Sistem Informasi Manajemen: Software, Database Dan Brainware. *Jurnal Ekonomi Manajemen Sistem Informasi*, *3*(3), 344–360. https://doi.org/10.31933/jemsi.v3i3.824

Informasi, S., Penelitian, L., & Masyarakat, K. (2014). Sistem Informasi Penelitian Lppm Di Universitas Dirgantara Marsekal Suryadarma Berbasis Web. *Jurnal Sistem Informasi Universitas Suryadarma*, *9*(1), 119–128. https://doi.org/10.35968/jsi.v9i1.848

Katarina Sianturi, S., Satriani Fansuri, D., & Najmiatul Aini, W. (2023). Algoritma Apriori untuk Mengetahui Pola Beli Konsumen Pada Sistem Informasi Market Basket Analysis Berbasis Andriod. *Jurnal Insan Unggul*, *11*(1), 35–58. https://doi.org/10.47926/jiu.2023.11.1.35-58

Melinda, V., & Zein, A. (2023). Perancangan Sistem Informasi Tour Dan Travel Berbasis Web Menggunakan Metode Personal Extreme Programming (Pxp) Pada Today Trip. *Jurnal Ilmu Komputer JIK*, *VI*(01), 25–32.

Nur, M., & Sabur, F. (2020). Pengembangan Sistem Informasi Penelitian dan Pengabdian Masyarakat (PPM) di ATKP Makassar. *Airman: Jurnal Teknik Dan Keselamatan Transportasi*, *2*(1), 76–89. https://doi.org/10.46509/ajtkt.v1i2.21

Nuryana, M. L., Ibrahim, T., & Arifudin, O. (2024). *IMPLEMENTASI DAN TRANSFORMASI SISTEM INFORMASI*. *5*(9), 1325–1337.

Putuhu, V. (2022). Sistem Informasi Manajemen Penelitian Dan Pengabdian Pnpb Pada Politeknik Negeri Ambon. *Jurnal Simetrik*, *12*(1), 553–560. https://doi.org/10.31959/js.v12i1.1068

Rafadi, S., & Industri, F. T. (2021). Aplikasi Pengajuan Proposal Penelitian

Berbasis Web Pada LPPM Universitas Taman Siswa Palembang. *Prosiding Seminar Nasional Multidisiplin Ilmu*, *2*(1), 543–550.

Rika Widianita, D. (2023). No 主観的健康感を中心とした在宅高齢者における 健康関連指標に関する共分散構造分析Title. *AT-TAWASSUTH: Jurnal Ekonomi Islam*, *VIII*(I), 1–19.

Sandika, I. K. B., & Hamid, H. (2021). Evaluasi Kesuksesan Penerapan Sistem Informasi di LPPM STMIK STIKOM Indonesia. *POSITIF : Jurnal Sistem Dan Teknologi Informasi*, *7*(2), 92–101. https://doi.org/10.31961/positif.v7i2.900

Shaputra, L., & Suwartika, R. (2024). *PERANCANGAN SISTEM INFORMASI LPPM POLITEKNIK PIKSI GANESHA BERBASIS WEB MENGGUNAKAN VS CODE FRAMEWORK LARAVEL*. *12*.

Sita Eriana, E., & Zein, A. (2021). Penerapan Metode Personal Extreme Programming Dalam Perancangan Aplikasi Pemilihan Ketua Hmsi Dengan Weighted Product. *Jurnal Ilmu Komputer JIK*, *IV*(02), 26–32.

Wahono, P. S., Safuan, S., & Alhabshy, M. A. (2023). Penggunaan Aplikasi E-Audit Dalam Sistem Informasi Manajemen Inspektorat Polri. *Jurnal Ilmiah Global Education*, *4*(2), 1122–1130. https://doi.org/10.55681/jige.v4i2.869

Winarsih, E., & Syam, E. (2022). Sistem Informasi Penelitian Dan Pengabdian Kepada Masyarakat (PKM) Dosen Berbasis Web (Studi Kasus: Lppm Universitas Islam Kuantan Singingi). *Jurnal Perencanaan Sains, Teknologi, Dan Komputer*, *5*(1), 6–13.

Yustika, W., Tusa, N., Siregar, diah, Aprinilova Barus, V., Abiyyu Alwansyah Hasibuan, M., & Nurbaiti. (2023). Peranan Sistem Database Di Dalam SistemInformasi Manajemen Pada UINSU(Universitas Islam Negeri Sumatera Utara). *SURPLUS: Jurnal Ekonomi Dan Bisnis*, *1*(2), 188–196.

LAMPIRAN