

Implementasi Metode PXP dan Prinsip SOLID untuk Integrasi Modul KIA pada SIMPUSWANGI Banyuwangi

Implementation of PXP Method and SOLID Principles for KIA Module Integration in SIMPUSWANGI Banyuwangi

**Wahyu Sahri Rhamadhan¹, Jaki Daniyudin², Daviq Rizqi Zawwa El Sofya³,
Lukman Hakim⁴, Lutfi Hakim⁵**

Jurusan Bisnis dan Informatika, Politeknik Negeri Banyuwangi, Banyuwangi, Indonesia^{2,3}

Email: sahri2827@gmail.com¹, jakidaniyudin90@gmail.com², daviqrizqi0708@gmail.com³, lukmanhakim@poliwangi.ac.id⁴, lutfi@poliwangi.ac.id⁵

Abstract

The Community Health Center Management Information System (SIMPUSWANGI) in Banyuwangi Regency plays a role in improving the quality of health services through patient data management and data reporting. However, this system faces challenges in integrating Maternal and Child Health (MCH) services with the Ministry of Health's SATU SEHAT platform, due to non-standardized services and difficulties in adapting the system to new regulations. To overcome this problem, this study developed the SIMPUSWANGI MCH Service module using the Personal Extreme Programming (PXP) method and SOLID design principles. This study aims to design and implement the integration of the module into the SATU SEHAT platform and ensure that the module built is flexible and easy to maintain. The PXP methodology was chosen to bring developers closer to user needs, so that the developed module is able to respond quickly to changes. The development results show the success of the implementation through 3 iterations with a User Acceptance Testing (UAT) acceptance rate of 94.67%, which is categorized as very good in each iteration. The application of SOLID principles to 192 classes resulted in a system with high flexibility (instability value of 0.97, Distance from Main Sequence 0) and an average Maintainability index of 89.28. These findings confirm that the application of SOLID keeps the architecture modular and maintainable, while PXP supports rapid and adaptive iteration. The combination of both has proven effective in producing a system that is responsive to changing needs and regulations without disrupting running services.

Keywords: SOLID Design Principle, PXP, KIA, SIMPUS, Flexibility, Maintainability

Abstrak

Sistem Informasi Manajemen Puskesmas (SIMPUSWANGI) di Kabupaten Banyuwangi berperan meningkatkan kualitas pelayanan kesehatan melalui pengelolaan data pasien dan pelaporan data. Namun, sistem ini menghadapi tantangan dalam integrasi pelayanan Kesehatan Ibu dan Anak (KIA) dengan platform SATU SEHAT milik Kementerian Kesehatan, yang diakibatkan oleh ketidakstandaran pelayanan dan kesulitan dalam menyesuaikan sistem dengan regulasi baru. Dalam mengatasi permasalahan ini, penelitian ini mengembangkan modul Pelayanan KIA SIMPUSWANGI menggunakan metode *Personal Extreme Programming* (PXP) dan prinsip desain SOLID. Penelitian ini bertujuan untuk merancang dan mengimplementasikan integrasi modul ke dalam platform SATU SEHAT, serta memastikan modul yang dibangun memiliki sifat fleksibel, dan mudah dipelihara. Metodologi PXP dipilih untuk mendekatkan pengembang dengan kebutuhan pengguna, sehingga modul yang dikembangkan mampu merespon perubahan dengan cepat. Hasil pengembangan menunjukkan keberhasilan implementasi melalui 3 iterasi dengan tingkat penerimaan *User Acceptance Testing* (UAT) 94.67% masuk dalam kategori *very good* pada setiap iterasi. Penerapan prinsip SOLID pada 192 kelas menghasilkan sistem dengan fleksibilitas tinggi (nilai *instability* 0.97, *Distance from Main Sequence* 0) dan *Maintainability index* rata-rata 89,28. Temuan ini menegaskan bahwa penerapan SOLID menjaga arsitektur tetap modular dan terpelihara, sementara PXP mendukung iterasi cepat dan adaptif. Kombinasi keduanya terbukti efektif dalam menghasilkan sistem yang responsif terhadap perubahan kebutuhan maupun regulasi tanpa mengganggu layanan yang berjalan.

Kata Kunci: SOLID Design Principle, PXP, KIA, SIMPUS, Flexibility, Maintainability

I. PENDAHULUAN

Sistem Informasi Manajemen Puskesmas Banyuwangi (SIMPUSWANGI) merupakan platform untuk digitalisasi pelayanan kesehatan berbasis web di Kabupaten Banyuwangi [1]. Sistem ini digunakan untuk pengolahan data pasien, pelayanan, dan laporan kesehatan. Namun dalam implementasinya sistem menghadapi tantangan, khususnya pada pengelolaan pelayanan Kesehatan Ibu dan Anak (KIA) dalam proses integrasi dan perubahan regulasi [2].

Berdasarkan hasil observasi dan wawancara peneliti mendapatkan bahwa aplikasi SIMPUSWANGI pada pelayanan Kesehatan Ibu dan Anak (KIA) masih belum terstandarisasi dengan platform Satu Sehat milik Kementerian Kesehatan (Kemenkes). Akibatnya, sistem belum dapat dilakukan integrasi data pelayanan kesehatan. Satu Sehat merupakan platform data center integrasi data kesehatan (HIE: *Health Information Exchange*) dinaungi oleh Kementerian Kesehatan. Melalui peraturan Permenkes No.24 Tahun 2022 tentang RME [3], dampak tidak sinkronisasi data baik dari layanan kesehatan daerah dan terpusat menjadikan sulitnya akses RME (Rekam Medis Elektronik) dengan instansi kesehatan diluar maupun didalam wilayah banyuwangi. Tidak hanya itu, SIMPUSWANGI mengalami kesulitan proses adaptasi jika adanya perubahan dan membutuhkan banyak waktu dan pembaruan karena perubahan berdampak pada layanan lain yang seharusnya tidak terpengaruh.

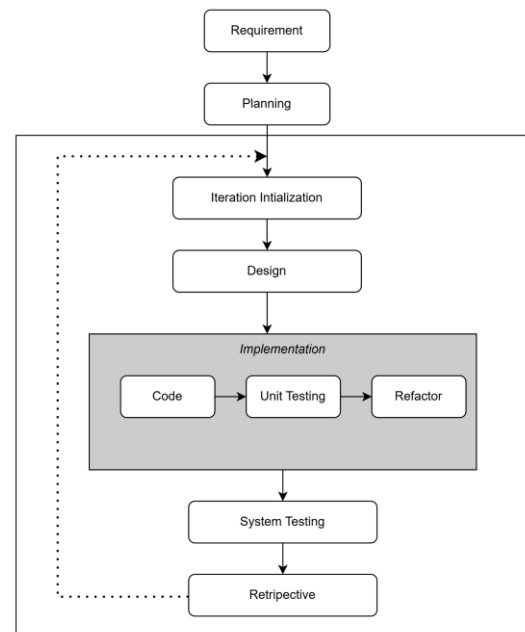
Sebagai solusi, penelitian ini dilakukan pengembangan modul pelayanan KIA (Kesehatan Ibu dan Anak) yang terdiri dari beberapa pelayanan seperti ANC (*Antenatal Care*), INC (*Intranatal Care*), PNC (*Postnatal Care*), *Neonatus*, Tumbuh Kembang, Kematian *Maternal-Perinatal* [4]. Dimana fokus pengembangan ini menciptakan sistem SIMPUSWANGI terintegrasi dan adaptif dengan perubahan regulasi tanpa mempengaruhi sistem yang berjalan. Beberapa desain untuk menciptakan sistem terstruktur dan adaptif seperti DRY (*Don't Repeat Yourself*), GRASP (*General Responsibility Assignment Software Patterns*), dan *SOLID Design Principles*. Menurut pencipta prinsip DRY [5], DRY menekankan penghindaran duplikasi kode untuk meningkatkan efisiensi dan tidak berulang. Selanjutnya untuk prinsip GRASP menurut penelitian [6] menekankan desain perangkat lunak berorientasi objek untuk membimbing penugasan tanggung jawab kepada kelas-kelas, lebih berfokus pada pola alokasi tanggung jawab dan tidak sepenuhnya membahas fleksibilitas pengembangan jangka panjang. Sebaliknya, *SOLID Design Principles* menurut [7] prinsip membangun sistem yang *flexible*, *maintainable*, independent modul tidak saling bergantung satu sama lain. Tidak hanya itu, untuk manajemen dalam proses pengembangan, ada beberapa cara metode pengembangan seperti *Waterfall*, XP, dan PXP. Metode *Waterfall* menurut [8] merupakan model linier siklus hidup teratur berurutan dari atas dan bawah, namun metode ini sulit adanya perubahan sistem *flexible* dan bersifat kaku. Selanjutnya metode XP (*Extreme Programming*) menurut [9], pendekatan pengembangan lincah dan adaptif mampu

mengakomodasi perubahan, namun dibutuhkan sebuah manajemen dan komunikasi tim yang kuat. Sebaliknya, metode PXP (*Personal Extreme Programming*) PXP pengembangan dari XP dikhususkan untuk tim pengembang kecil dan tetap menerapkan prinsip-prinsip XP [10].

Berdasarkan masalah dan pertimbangan tersebut, Peneliti mengembangkan Aplikasi KIA menggunakan SOLID sebagai desain code dan PXP sebagai manajemen project pengembangan [11]. Dimana penerapan prinsip SOLID dapat mengurangi resiko ketergantungan modul, dan memungkinkan perubahan tanpa mempengaruhi bagian lain dari sistem [12]. penerapan PXP menghasilkan sistem yang efisien, adaptif, dan feedback yang cepat terhadap perubahan kebutuhan pengguna dengan tim kecil [12]. Kombinasi, SOLID mengedepankan desain code *independent*, *flexible* dan *maintainable* dengan PXP bersifat mampu mengakomodasi perubahan dengan cepat kebutuhan cepat tanpa mempengaruhi sistem yang lain.

Tujuan dari penelitian ini untuk mengembangkan sistem terintegrasi sesuai dengan kebutuhan user/regulasi dan *flexible* dengan adanya perubahan serta *maintainable* untuk masa depan. Dengan hal itu, diharapkan hasil penelitian dapat mengintegrasikan platform SIMPUSWANGI dengan sistem yang dapat beradaptasi dan mudah dirawat dengan adanya perubahan regulasi.

II. METODE PENELITIAN



Gambar 1. Tahapan Penelitian

Penelitian ini dilaksanakan menggunakan metode PXP (*Personal Extreme Programming*), dilaksanakan beberapa tahap seperti yang ditunjukkan pada **Gambar 1** Tahapan yang dilakukan terdiri dari:

A. Requirements

Peneliti menggunakan pendekatan kualitatif dengan

metode observasi dan wawancara. Lokasi penelitian di Dinas Kesehatan Banyuwangi, Puskesmas. Subjek penelitian meliputi tenaga kesehatan yang terlibat pada pelayanan KIA dengan jumlah 5 orang (1 Dokter, 2 Bidan, 1 Simpuher, 1 Admin SIMPUS). Instrumen penelitian digunakan yaitu pedoman observasi dan pertanyaan wawancara semi terstruktur. Prosedur penelitian dilakukan beberapa tahap: (1) studi literatur terkait standar KIA dan integrasi Satu Sehat; (2) observasi bagaimana *flow* bisnis pelayanan KIA pada SIMPUSWANGI; (3) wawancara dengan responden untuk menggali kebutuhan dan kendala; (4) melakukan analisis data dengan penarikan kesimpulan dengan format user story (sebagai siapa?, kebutuhannya?, harapannya?). Penyusunan *user story* dilakukan validasi kembali kepada pengguna untuk memastikan kesesuaian dengan kebutuhan nyata pengguna.

B. Planning

Peneliti merumuskan fungsionalitas berdasarkan hasil user story diperoleh pada tahap *requirement*. Setiap user story dilakukan identifikasi pada aspek prioritas dengan mempertimbangkan perspektif *Product Owner (PO)* dan *Developer Team (DT)*. Dari sisi PO, penentuan skala prioritas menggunakan metode MoSCoW menentukan prioritas berdasarkan nilai bisnis [13]. Dimana metode ini cocok dengan metodologi PXP untuk client persyaratan belum jelas dan membutuhkan pengembangan cepat dengan membagi 4 kriteria bobot penilaian [14], yaitu

Tabel 1. Klasifikasi bobot MoSCoW

Kategori MoSCoW	Bobot
<i>Must Have</i>	4
<i>Should Have</i>	3
<i>Could Have</i>	2
<i>Won't Have</i>	0

Dari sisi DT mengukur dengan menggunakan *Story point* dimana memperhatikan 4 dimensi penilaian seperti pada Tabel 1, yaitu upaya pengerjaan, estimasi waktu, tingkat kompleksitas, serta potensi resiko, yaitu.

Tabel 2. Klasifikasi Story Point

Story Point	Upaya	Waktu	Kompleksitas	Risiko
1	<i>Minimum</i>	<i>Beberapa menit</i>	<i>Rendah</i>	<i>Tidak ada</i>
2	<i>Minimum</i>	<i>Beberapa Jam</i>	<i>Rendah</i>	<i>Tidak ada</i>
3	<i>Sedang</i>	<i>Satu hari</i>	<i>Rendah</i>	<i>Rendah</i>
5	<i>Sedang</i>	<i>Beberapa hari</i>	<i>Sedang</i>	<i>Sedang</i>
8	<i>Besar</i>	<i>Satu Minggu</i>	<i>Sedang</i>	<i>Sedang</i>

Prosedur penilaian dilakukan secara sistematis melalui diskusi kelompok. Peneliti bersama PO mengklasifikasikan setiap user story ke dalam kategori MoSCoW serta secara kolektif dimana DT memberikan nilai *Story Point* berdasarkan tabel 2. Dalam menentukan Total Urgensi (TU) menggunakan formula, yaitu

$$TU = \text{Bobot MoSCoW} \times \text{Story Point} \quad (2.1)$$

Hasil perhitungan TU disusun dalam tabel prioritas yang menampilkan urutan kebutuhan dari tingkat urgensi tinggi ke rendah. Dimana ya prioritas utama akan untuk dilakukan diawal, sedangkan kebutuhan dengan nilai rendah dapat ditunda atau dieliminasi.

C. Interaction Initialization

Tahap ini menentukan setiap iterasi dimulai, dimana menentukan aspek *user story* yang sudah di list pada tahap *planning*. Proses ini mencakup pemilihan kebutuhan berdasarkan nilai urgensi, penyusunan rencana teknis, serta persiapan eksekusi. Dengan begitu proses pengembangan berjalan terarah dan terukur.

D. Design

Design dilakukan berdasarkan daftar *user story* yang terpilih pada setiap iterasi. Proses desain dilaksanakan melalui tiga langkah utama. Pertama, penyusunan Use Case Diagram menggunakan pendekatan pemodelan UML untuk menggambarkan interaksi setiap aktor (petugas kesehatan). Kedua, penyusunan Entity Relationship Diagram (ERD) untuk merancang struktur data, relasi antar entitas, dan alur penyimpanan informasi sesuai kebutuhan modul. Ketiga pembuatan desain antarmuka dalam bentuk mockup menggunakan tools desain salah satunya Figma, untuk memvisualisasikan tata letak tampilan dan alur interaksi pengguna. Hasil desain divalidasi bersama pengguna untuk memastikan kesesuaian dengan perencanaan yang ditetapkan.

E. Implementation

E.1 Coding

Pada tahap coding, peneliti mengimplementasikan hasil desain ke dalam kode aplikasi dengan menerapkan prinsip SOLID agar struktur program terjaga dan mudah dikembangkan. Prinsip Single Responsibility Principle (SRP) dimana memastikan setiap kelas hanya memiliki satu tanggung jawab spesifik. Merancang Principle Open/Closed Principle (OCP) dengan sistem terbuka untuk pengembangan namun tetap stabil terhadap perubahan inti. Menjaga Liskov Substitution Principle (LSP) dengan memastikan kelas turunan dapat menggantikan kelas induk tanpa mengubah perilaku sistem. Mewujudkan Interface Segregation Principle (ISP) dengan memisahkan antarmuka agar hanya berisi fungsi relevan sesuai kebutuhan. Menerapkan konsep Dependency Inversion Principles (DIP) dengan membangun ketergantungan modul tingkat tinggi pada abstraksi, bukan pada detail implementasi.

E.2 Unit Testing

Peneliti melakukan pengujian unit menggunakan PHPUnit untuk memastikan modul berjalan sesuai dengan alur bisnis yang diharapkan dan meminimalisir bug sistem. Proses dilakukan dengan menentukan modul kritis yang diuji (input data pasien, validasi form, laporan pengujian), menyusun test case sesuai skenario valid dan tidak valid.

E.3 Code Refactor

Tahap ini dilakukan setelah unit testing menemukan kegagalan pada modul. Peneliti meninjau ulang bagian kode yang bermasalah, memperbaiki logika tidak sesuai, kemudian menjalankan kembali pengujian hingga perbaikan dipastikan berhasil tanpa menimbulkan bug baru. Proses ini dilakukan secara iteratif hingga seluruh tes case yang disusun sesuai kebutuhan sistem.

F. System testing

Tahap ini dilakukan untuk melakukan pengujian terhadap alur sistem utama. Dimana peneliti menggunakan pendekatan UAT (*User Acceptance Testing*) untuk mengukur penerimaan dari user terkait aplikasi yang dibuat [15]. Proses dilakukan dengan menyusun instrumen kuesioner berbasis skala Likert (1-5), membagikan kuesioner kepada tenaga kesehatan dan staf SIMPUS yang menjadi responden dari setiap iterasi yang sudah dirancang pada tahap planning untuk validasi kesesuaian hasil dengan expetasi user. Untuk menentukan nilai klasifikasi UAT, peneliti menggunakan skala likert seperti pada Tabel 3 berikut:

Tabel 3. Interpretasi Score UAT	
Score	Interpretasi
5	Sangat Sesuai
4	Sesuai
3	Cukup Sesuai
2	Tidak Sesuai
1	Sangat Tidak Sesuai

Adapun range skor kriteria dari skala *likert* untuk menentukan klasifikasi skala UAT yang dihasilkan dengan presentasi penerimaan user. Pada tabel 4 berikut merupakan klasifikasi skoring kriteria.

Tabel 4. Interpretasi Klasifikasi Kriteria	
% Total Score	Interpretasi
$0\% \leq x \leq 36\%$	<i>Bad</i>
$36\% \leq x \leq 52\%$	<i>Little Bad</i>
$52\% \leq x \leq 68\%$	<i>Medium</i>
$68\% \leq x \leq 84\%$	<i>Good</i>
$84\% \leq x \leq 100\%$	<i>Very Good</i>

G. Retrospective

Sebagai tahap penutup setiap iterasi. Peneliti membandingkan waktu penyelesaian aktual dengan estimasi yang telah direncanakan pada tahap planning, mencatat selisih yang terjadi, serta menganalisis penyebab perbedaan. Hasil analisis digunakan sebagai umpan balik untuk memperbaiki perencanaan dan pelaksanaan pada iterasi berikutnya.

H. Flexibility dan Maintainability

Pengujian *flexibility* dan *Maintainability* untuk menentukan seberapa *flexibility* sistem yang dibuat dengan melakukan pengukuran kualitas *Software Quality Metrics* dan menilai seberapa *Maintainability Software* dimana menentukan tingkat kesulitan dalam melakukan proses perawatan dan pengembangan dimasa akan datang. Proses

pengujian ini menggunakan *tools* yaitu *PHPMetrics* dimana aplikasi ini menilai *metric*, dimana mengukur pada tingkat kompleksitas aplikasi, ketergantungan *code*.

III. HASIL DAN PEMBAHASAN

A. Hasil Penelitian

A.1 Requirements

Berdasarkan hasil studi literatur, peneliti mengidentifikasi standar pelayanan KIA dibagi menjadi modul pengiriman (*Fast Healthcare Interoperability Resources*) FHIR. Namun, hasil observasi di lapangan memperlihatkan belum tersedianya kebutuhan pemenuhan standar FHIR pada SIMPUSWANGI. Hal ini diperkuat dari proses wawancara semi-terstruktur dengan enam responden (3 Staff KIA, 2 Staff Puskesmas, dan 1 Staf pengembang) sehingga diperoleh sejumlah kebutuhan pengguna yang dianalisis dengan pendekatan user story, Rekap kebutuhan dirangkum pada tabel 5 sebagai dasar pengembangan sistem.

Tabel 5. User Story

Kode	Pengguna	Kebutuhan
US-001	Staff dan Kepala KIA	Dukungan untuk layanan ANC, INC, PNC, Neonatus, Tumbuh Kembang, Laporan Kematian
US-002	Staff dan Kepala KIA	Klasifikasi pelayanan pemeriksaan (obstetri, subjektif, objektif)
US-003	Tim IT	Sistem modular dan fleksibel terhadap regulasi
US-004	Tim IT	Service Integrasi Rest API FHIR
US-005	Petugas KIA	Pencatatan Sub Pelayanan record layanan
US-006	Tim IT	Validasi otomatis pada setiap <i>form</i>
US-007	Semua Pengguna	Notifikasi status kirim data (berhasil/gagal)
US-008	Staf dan Kepala KIA	Input USG otomatis berdasarkan trimester
US-009	Staf dan Kepala KIA	Target BB otomatis berdasarkan IMT
US-010	Staf dan Kepala KIA	Status lila otomatis 10
US-011	Staf dan Kepala KIA	Status berat badan bayi otomatis

A.2 Planning

Tabel 6 berikut merupakan hasil planning dilakukan peneliti dapat membuat daftar iterasi tingkat prioritas menggunakan pendekatan *MosCow*, dan *Story Point*:

Tabel 6. Klasifikasi Story Point

Ite-rasi	Kategori Fitur	Jumlah User Story	Average Mos Cow	Average Story Point	Average rag e TU	Average Estim asi
1	Pengembangan Fitur KIA (ANC, INC, Kematian/Perinatal, PNC, Neonatus, Tumbuh Kembang	20	4	5	20	4 hari

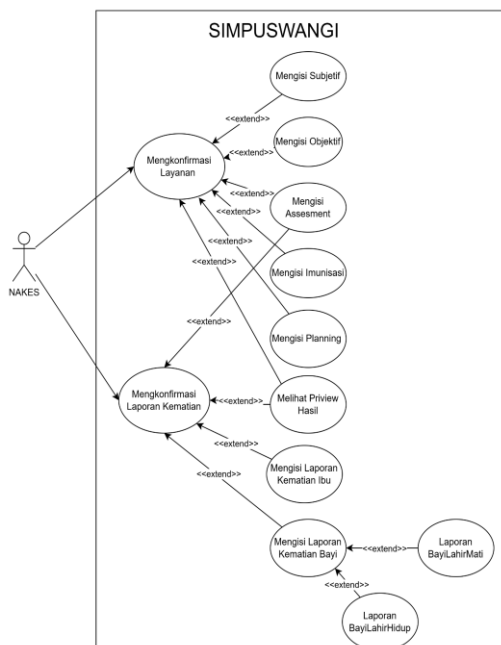
2.	Integrasi Resource Satu Sehat (ANC, INC, Kematian Maternal/Perinatal, PNC, Neonatus, Tumbuh Kembang)	9	4	5	20	3 hari
3	Validasi, Notifikasi, dan Konfirmasi Pengiriman	4	3	3	9	3 hari

A.3 Interaction Initialization

Proses pengembangan dilakukan secara interaktif yang mencakup desain, implementasi, pengujian, dan retrospective. Berdasarkan tabel 2.4 tahap planning, dimana peneliti berfokus pada pengembangan fitur layanan KIA (ANC, INC, PNC, Neonatus, Tumbuh Kembang, serta pencatatan Kematian Maternal dan Perinatal) dengan jumlah user story 20 dan estimasi rata penyelesaian 4 hari. Selanjutnya pada iterasi ke dua menitikberatkan pada integrasi resource Satu Sehat dengan 9 user story dan estimasi rata 3 hari. Iterasi ketiga difokuskan pada pengembangan validasi otomatis, notifikasi, dan konfirmasi pengiriman resource dengan 4 user story dan estimasi rata 3 hari. Dengan demikian, setiap iterasi dijalankan secara terstruktur dengan pembagian jelas.

A.3 Design

A.3.2 Use Case



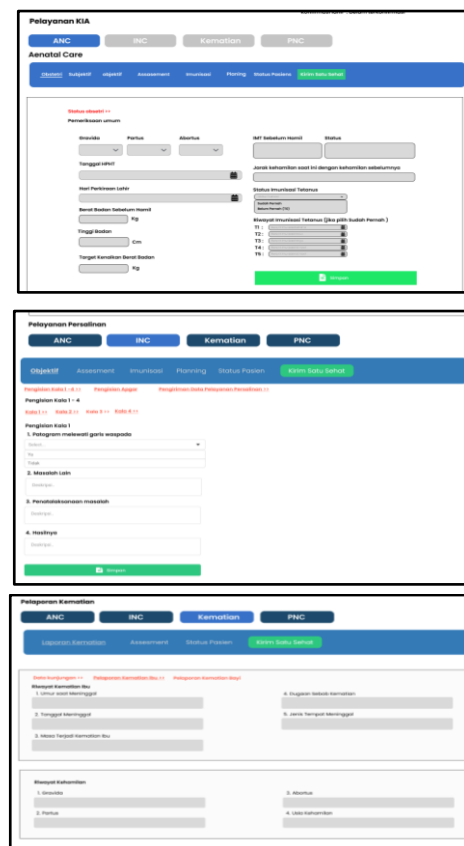
Gambar 2. Use Case

Pada pengembangan aplikasi kesehatan ibu dan anak pada SIMPUSWANGI akan mendeskripsikan hubungan pada aktivitas pelayanan pada modul ANC, INC, Pelaporan Kematian Maternal dan Perinatal, dimana actor Nakes /

Petugas Kesehatan melakukan pelayanan, mulai dari pemeriksaan Objektif, Subjektif, Assessment. Dengan hal itu peneliti dapat menentukan alur bisnis dan kebutuhan berdasarkan diagram *Use Case*. Berikut diagram *use case* dapat dilihat pada **Gambar 2**.

A.3.1 MockUp

Dalam proses mendesain peneliti membagi layanan menjadi 6 fitur utama, yaitu ANC (Ibu hamil), INC (Ibu bersalin), PNC (Ibu nifas), *Neonatus* (Bayi 1 bulan), Tumbuh Kembang (Bayi umur 2 bulan sampai umur 5 tahun). Menghasilkan beberapa tampilan yang sudah disesuaikan dengan kebutuhan pengguna proses desain melibatkan user dalam setiap iterasi.



Gambar 3. Mockup Pelayanan KIA

A.3.3 PDM (Physical Data Model)

Dimana peneliti mendesain entitas seperti *master_layanan* sebagai list fitur utama aplikasi seperti ANC, INC, Kematian, Tumbuh Kembang, PNC, Neonatus, *master_sub_layanan* sebagai layanan dibawah fitur utama, kemudian ada *record_pemeriksaan* digunakan untuk record dalam 1 periode pasien layanan terdiri dari banyak loket kunjungan. kemudian *record_pemeriksaan_detail* yaitu record pasien setiap kali kunjungan di puskesmas terdiri dari berbagai macam pelayanan dilakukan dengan loket yang sama. *hasil_pemeriksaan* yaitu sebagai tempat untuk menaruh semua hasil pemeriksaan dari semua layanan.

Berikut gambar 4 menunjukkan gambaran dari PDM pengembangan aplikasi SIMPUSWANGI.



Gambar 4. Tampilan Pelayanan KIA

A.4 Implementation

Pada iterasi pertama peneliti berfokus pada pengembangan aplikasi untuk menyiapkan sistem yang sesuai dengan kebutuhan proses integrasi Satu Sehat. Dimana membangun sistem ANC, INC, Kematian *Maternal* dan *Perinatal*, PNC, *Neonatus*, Tumbuh Kembang. Gambar 5 berikut tampilan hasil pengembangan.

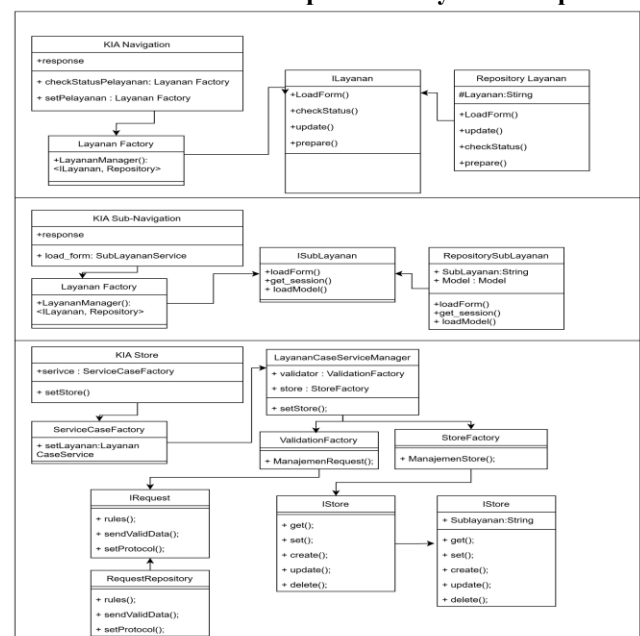
Gambar 5. Tampilan Pelayanan KIA

Selanjutnya pada iterasi kedua berfokus pada proses integrasi Satu Sehat, dengan mengimplementasikan resource FHIR seperti *Encounter*, *EpisodeOfCare*, *Condition*. Pada iterasi ketiga berfokus pada pengembangan sistem notifikasi, dan validasi untuk menghindari adanya human error, dan sebagai response interaktif untuk pengguna mengetahui status pengiriman berhasil atau masih ada kendala. Berikut tampilan pada iterasi ketiga pengembangan validation dan notifikasi satu sehat sistem.

Gambar 6. Validation Form Input

Gambar 7. Notifikasi Satu Sehat

B.1.1 Arsitektur SOLID Aplikasi Pelayanan Simpus

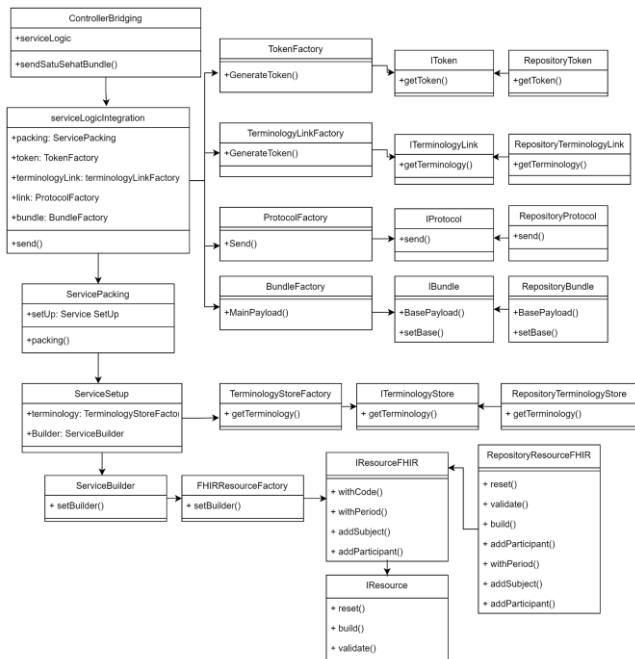


Gambar 8. Service KIA

Penerapan arsitektur SOLID pada service KIA akan dibagi menjadi 5 aturan desain. Pertama SRP dimana dapat kita lihat setiap arsitektur yang dikembangkan terfokus pada satu tugas, seperti *KIANavigation* berfokus pada proses navigasi pelayanan utama, *KIASubNavigation* berfokus pada proses sub-pelayanan utama. *KIAStore* berfokus pada proses validasi dan penyimpanan data pada database. Kedua OCP dimana sistem dikembangkan bisa dikembangkan tanpa harus merubah code yang ada, dengan menambahkan sebuah repository baru dan menerapkan sistem bongkar pasang, tanpa mempengaruhi code yang lain. Ketiga LSP *Interface* akan menjadi parent dapat diturunkan pada child yaitu *repository* dan *repository* dapat menggantikan parent *Interface* untuk eksekusi lebih detail seperti *Ilayanan* dapat digantikan oleh *RepositoryLayanan* sebagai spesifikasi detail logiknya. Keempat ISP dimana peneliti tidak memaksakan method yang tidak diperlukan oleh *repository* sehingga dapat kita lihat ada banyak interface digunakan, seperti *Ilayanan*, *ISubLayanan*, *IStore*, *IRequest* menyesuaikan dengan tugas yang diemban.

Kelima ada DIP dimana penerapan modul *high level* tidak boleh bergantung pada modul *low level* namun pada abstraksi, dapat kita lihat *KIANavigation*, *KIASubNavigation*, *KIAStore* bergantung pada abstraksi seperti *INavigation*, *ISubNavigation*, *IStore*.

B.1.2 Arsitektur SOLID Service Satu Sehat



Gambar 9. Satu Sehat Service

Penerapan service akan dibagi menjadi beberapa tugas, konsep solid SRP dapat kita lihat diatas *ControllerBridging*, *serviceLogicIntegration*, *ServicePacking*, *ServiceSetup* dimana class program yang mengatur lalu lintas logic data. Konsep OCP terlihat pada *TokenRepository*, *RepositoryTerminologyLink*, *RepositoryProtocol*,

RepositoryBundle, *RepositoryTerminologyStore*, *RepositoryResourceFHIR* yang dapat dikembangkan lagi tanpa mempengaruhi code antar repository. Penerapan LSP dapat dilihat hubungan antara interface sebagai class utama dan kelas turunannya *repository* dapat menggantikan interface dalam penerapan lebih detail. Penerapan ISP dapat dilihat dari *Interface* yang terkoordinir berdasarkan aspek yang difokuskan *InterfaceToken*, *InterfaceTerminologyLink*, *InterfaceProtocol*, *InterfaceBundle*, *InterfaceTerminologyStore*, dan *InterfaceTerminologyStore*. Penerapan DIP kelas *high level* modul tidak bergantung pada *low level* modul, seperti halnya modul *serviceLogicIntegration* bergantung pada abstraksi yaitu *IToken*.

B.1.4 Unit Testing

Dalam memastikan setiap unit dapat berjalan dengan baik sesuai *flow* bisnis yang diharapkan, serta mencegah dan mendeteksi *bug*, dengan hal itu peneliti melakukan pengujian *unit testing*, berikut rekap pengujian *unit testing* pada iterasi pertama, kedua, dan ketiga.

Tabel 7. Rekap Pengujian Unit Testing

Iterasi	Fokus Pengujian	Total Skenario Uji	Status Pengujian
1	KIA (ANC, INC, Kematian Maternal dan Perinatal, PNC, Neonatus, Tumbuh Kembang)	33	Lulus (100%)
2	Integrasi Satu Sehat (Payload Generation dan Bundling)	11	Lulus (100%)
3	Validasi Form dan Notifikasi Pengiriman Data	21	Lulus (100%)

Pengembangan sistem Kesehatan Ibu dan Anak (KIA) dilaksanakan melalui 3 iterasi. Dimana setiap fase mencapai 100% kelulusan pengujian unit. Pada iterasi pertama, ada 33 skenario pengujian pada fitur inti KIA (ANC, INC, Kematian Maternal/Perinatal, PNC, Tumbuh Kembang, Neonatus). Pada iterasi kedua ada 11 skenario pengujian untuk memverifikasi payload dan *bundling resource* pada integrasi platform Satu Sehat. Pada iterasi ketiga, terdapat 21 skenario meningkatkan kualitas data melalui validasi *form* dan notifikasi pengiriman data. Hasil keseluruhan menunjukkan keberhasilan pengembangan sistem yang functional, terintegrasi, terhindar dari adanya bug sistem baik dari sisi backend maupun interaksi dengan pengguna.

A.1.5 System Testing

Pada proses *system testing* peneliti menggunakan 2 responden sebagai penerima dari sisi stakeholder, yaitu 1 orang sebagai pegawai KIA dan 1 orang merupakan developer Simpungwangi. Berikut rekap hasil UAT selama 3 iterasi.

Tabel 8. Rekap Pengujian Penerimaan UAT

Iterasi	Fokus Pengujian	Total Skenario Penerimaan	Status Pengujian
1	UAT Fitur Layanan KIA ANC, INC, Kematian <i>Maternal-Perinatal</i>	3	96,67%
2	UAT Integrasi data satu sehat (Encounter dan Condition)	3	93,33%
3	UAT validasi form, Notifikasi, dan input otomatis berbasis indikator	10	94%
Total		16	94,67%

Berdasarkan hasil keseluruhan, dari total 16 skenario UAT yang dilakukan selama 3 iterasi dengan menggunakan 2 responden. Sistem yang dibuat mendapatkan nilai penerimaan sudah sangat bagus, dimana pada iterasi pertama nilai penerimaan sebesar 96,67%, pada iterasi kedua mendapatkan 93,33%, dan selanjutnya pada iterasi ketiga mendapatkan 94%. Dengan hasil rata total mendapatkan kategori *Very Good* pada angka 94,67%, rata - rata dari stakeholder berada penerimaan SS (Sangat Sesuai) dan S (Sesuai).

A.1.6 Retrospective

Pada tahap *retrospective* dilakukan perhitungan estimasi dan hasil realisasi untuk menentukan performa kinerja dan manajemen waktu dengan deadline yang sudah ditentukan diawal. Berikut rekap hasil *retrospective* yang dilakukan selama 3 iterasi.

Tabel 9. Rekap *Retrospective*

Iterasi	Jumlah User Story	Rata-rata Estimasi	Rata-rata Realisasi	Selisih
1	22	4 hari	3 hari	1 hari lebih cepat
2	10	2.5 hari	2.4 hari	0.1 hari lebih cepat
3	8	2.4 hari	2.4 hari	sesuai estimasi

Berdasarkan hasil *retrospective* dari 3 iterasi, dimana peneliti developer menunjukan kinerja yang cukup efisien terutama pada iterasi 1 (Fitur KIA), dimana mayoritas *user story* dengan estimasi 4-5 hari berhasil diselesaikan hanya dalam 3-4 hari, menunjukan percepatan rata - rata 1,1 hari serta iterasi 1 memiliki performa cukup baik. Pada iterasi kedua (Integrasi Satu Sehat) juga menunjukan efisiensi dengan rata - rata realisasi 2,4 hari dari estimasi lebih cepat 0.1 hari. Pada iterasi ketiga sesuai estimasi dan realisasi tepat waktu dalam stabilitas proses kerja.

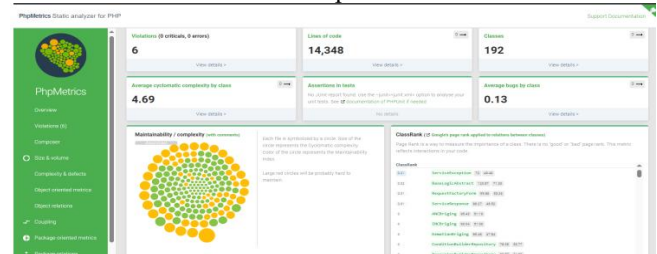
A.2 Pengujian Sistem *Flexible* dan *Maintainable*

Pengujian *Flexible* dan *Maintainable* dilakukan untuk menentukan seberapa flexible mudah adanya perubahan dan perawatan dimasa akan datang pada sebuah arsitektur

aplikasi. Dengan menggunakan *PHPMetrics* dalam menganalisis code dari aspek kompleksitas, *cohesion*, *coupling*, dan *violation* (deteksi bug dan kompleksitas). Proses menjalankan pengukuran berjalan perintah terminal seperti berikut.

```
phpmetrics --report-html=myreport.html
./<<[nama folder app]>>
```

Dengan menjalankan perintah diatas *PHPMetrics* akan membuat sebuah report *console* dan web dari hasil analisis. Berikut untuk hasil report dari *PHPMetrics*.



Gambar 10. Satu Sehat Service

Berikut hasil report *console* *PHPMetrics*.

LOC	
Lines of code	14348
Logical lines of code	8609
Comment lines of code	5739
Average volume	389.87
Average comment weight	40.14
Average intelligent content	40.14
Logical lines of code by class	45
Logical lines of code by method	12
Object oriented programming	
Classes	192
Interface	25
Methods	697
Methods by class	3.63
Lack of cohesion of methods	1.5
Coupling	
Average afferent coupling	0.67
Average efferent coupling	1.83
Average instability	0.97
Depth of Inheritance Tree	1.02
Package	
Packages	1
Average classes per package	221
Average distance	0
Average incoming class dependencies	0
Average outgoing class dependencies	0
Average incoming package dependencies	0
Average outgoing package dependencies	0
Complexity	
Average Cyclomatic complexity by class	4.69
Average Weighted method count by class	7.32
Average Relative system complexity	45.04
Average Difficulty	5.1
Bugs	
Average bugs by class	0.13
Average defects by class (Kan)	0.37
Violations	
Critical	0
Error	0
Warning	6
Information	0

A.2.1 Pengujian *Flexibility*

Pengujian untuk menentukan nilai *Flexibility* sebagai dasar perhitungan untuk membangun sistem dengan membandingkan 2 parameter dari Robert C Martin

(*Instability* dan *distance*) [16]. Berikut hasil analisis report yang dihasilkan *PHPMetrics*.

- Nilai rata - rata *Instability* dari report yang dihasilkan *phpmetrics* dengan pengujian 192 class yaitu 0,97 menunjukkan secara tidak langsung, bahwa modul program memiliki potensi tinggi untuk diubah tanpa menyebabkan efek berantai yang parah pada modul lain.
- Nilai rata - rata *Distance* sebesar 0 adalah hasil yang sangat ideal dalam kerangka Robert C Martin. Dimana secara rata - rata implementasi arsitektur modul yang tidak stabil dengan begitu arsitektur memiliki *flexible* dan terstruktur dengan baik.
- Rata - rata Ca 0,67 dan Ce 1,83 menunjukkan kopling relatif rendah pada tingkat kelas mendukung adanya kemandirian modul pada setiap code program.
- Kompleksitas rata *Cyclomatic Complexity* per kelas sebesar 4,59 menunjukkan kompleksitas kode rendah, memfasilitasi pemahaman dan pemeliharaan, sehingga meningkatkan *flexibility* pada aplikasi dikembangkan.

A.2.3 Pengujian *Maintainability*

Pengujian untuk menentukan nilai MI (*Maintainability Index*) sebagai dasar perhitungan untuk membangun sistem yang mendukung *Maintainability* [16]. Tabel 10 berikut merupakan hasil pengujian dengan menggunakan tools *PHPMetrics*.

Tabel 10. *Maintainability Index report PHPMetrics*

Value of <i>Maintainability</i> Index	Jumlah Class Indication	Classification
MI < 65	0	Low Maintainability
MI 65-84	61	Maintainability Status
MI > 85	131	High Maintainability Status
Average: 89,28	Total : 192	High Maintainability Status

Berdasarkan hasil dari 192 kelas hasil pengembangan sebagai variabel pengujian untuk menentukan nilai *maintainability* indeks dari aplikasi, ditemukan 131 kelas masuk dalam kategori *High Maintainability Status* dengan index (MI > 85), sedangkan 61 kelas sisanya masuk dalam kategori *Maintainability Status* dengan index (MI 64-84). Tidak hanya itu, didapatkan rata *Maintainability Index* dari 192 kelas yang diuji memiliki nilai 89,28 yang masuk dalam kategori *High Maintainability Status*.

B. Pembahasan

Pengembangan sistem pelayanan KIA dilakukan melalui tiga iterasi dengan metode pengembangan *Personal Extreme Programing (PXP)*. Setiap iterasi berhasil menyelesaikan pengujian dengan kelulusan 100%, mencakup fitur layanan ANC, INC, PNC, Neonatus, Tumbuh Kembang, serta Kematian Maternal dan Perinatal. Hasil pengujian UAT yang dilakukan dengan 2 responden menunjukkan tingkat penerimaan rata-rata 94,67% dalam kategori *Very Good*. Tidak hanya itu, dari sisi arsitektur sistem, pengujian

sampling 192 kelas kode yang dihasilkan menggunakan *PHPMetrics* menunjukkan nilai *Instability* 0,97 dan *Distance* 0, menunjukkan sistem *flexible* dan modular. Dari sisi *maintainability* sistem mendapatkan rata-rata 89,28 menunjukkan sistem mudah dipelihara. Nilai *coupling* yang rendah serta kompleksitas kode yang terkendali turut mendukung kualitas sistem.

Hasil penelitian ini selaras dengan tujuan penelitian yaitu mengembangkan sistem pelayanan KIA yang terintegrasi sesuai regulasi nasional, dan *flexible* dengan adanya perubahan serta *maintainable* untuk masa depan. Hal ini dapat dilihat sistem berhasil dirancang menyesuaikan regulasi nasional pada resource FHIR (Ecounter, Condition, Episode of Care) Satu Sehat milik Kementerian Kesehatan. Selain itu, Berdasarkan hasil pengujian *PHPMetrics* menunjukkan bahwa sistem memiliki tingkat modularitas dan fleksibilitas yang tinggi. Temuan ini sejalan dengan kajian teori yang menegaskan bahwa DRY berkontribusi pada efisiensi dan KISS menekankan kesederhanaan, namun keduanya memiliki keterbatasan pada aplikasi kompleks. Demikian pula, metode Waterfall dinilai kaku terhadap perubahan, sementara XP lebih adaptif tetapi menuntut koordinasi tim yang *intensif*. Dalam penelitian ini, kombinasi SOLID dan PXP terbukti lebih relevan karena mampu mendukung modularitas, fleksibilitas, dan iterasi cepat. Kondisi ini menjadi bukti kemampuan adaptasi sistem terhadap perubahan, serta memastikan proses pengembangan berkelanjutan dapat dilakukan tanpa menimbulkan gangguan pada layanan yang telah berjalan. Penerapan prinsip SOLID yang dipadukan dengan metode PXP membentuk dasar pengembangan yang stabil sekaligus adaptif. Melalui kombinasi ini, sistem pelayanan KIA berhasil diwujudkan sesuai regulasi nasional dan tetap siap menghadapi perubahan kebutuhan di masa depan.

Penelitian ini memiliki keterbatasan dalam jumlah responden uji pada *User Acceptance Test (UAT)* hanya melibatkan 2 subjek responden dan pengujian dilakukan dalam lingkungan terbatas. Penerapan resource FHIR Satu Sehat hanya fokus pada pelayanan KIA belum dapat mengcover case lain. Tidak hanya itu, belum ada sistem spesifik untuk memonitoring dan laporan layanan KIA. Oleh karena itu, penelitian selanjutnya disarankan untuk melakukan uji coba dengan cakupan lingkungan yang lebih luas, penerapan resource FHIR pada case lain.

IV. KESIMPULAN

Peneliti berhasil mengembangkan pada pelayanan KIA (Kesehatan Ibu dan Anak) menggunakan pendekatan PXP dan menerapkan SOLID Design Principle dalam tiga iterasi, hal ini dibuktikan dengan UAT mencapai 94,67% kategori *very Good*. Tidak hanya itu, penerapan prinsip SOLID memberikan dampak cukup baik terhadap fleksibilitas dan *maintainability* sistem, dengan pengujian terhadap 192 kelas menggunakan *PhpMetrics* menunjukkan arsitektur yang

stabil (*Distance* = 0), *flexibility* (*Instability* = 0.97), serta mudah dipelihara (*Maintainability Index* rata-rata = 89,28). Penerapan prinsip desain SOLID dipadukan dengan metode PXP menjadi dasar penting dalam penelitian ini. SOLID menjaga arsitektur tetap modular, stabil, dan mudah dipelihara, sedangkan PXP memungkinkan iterasi pengembangan yang cepat dan fleksibel. Keterpaduan keduanya menghasilkan pendekatan pengembangan yang responsif terhadap perubahan kebutuhan maupun regulasi, tanpa mengorbankan kualitas arsitektur. Melalui pendekatan ini, sistem pelayanan KIA berhasil dikembangkan sesuai regulasi nasional serta siap menghadapi dinamika pelayanan kesehatan di masa depan. Pada penelitian selanjutnya dapat diarahkan pada uji coba pada lingkungan yang lebih luas, melengkapi penerapan resource FHIR yang lebih lengkap untuk layanan case lain. Tidak hanya itu, menerapkan prinsip SOLID secara konsisten untuk menjaga kualitas dan fleksibilitas sistem dalam jangka panjang.

UCAPAN TERIMA KASIH

Peneliti mengucapkan terimakasih kepada Dinas Kesehatan Kabupaten Banyuwangi atas dukungan, kolaborasi, dan izin yang diberikan selama proses penelitian ini berlangsung. Besar harapan penelitian ini dapat memberikan manfaat langsung bagi Dinas Kesehatan, meningkatkan efisiensi pencatatan dan pelaporan pelayanan KIA melalui sistem SIMPUSWANGI yang dapat terintegrasi dengan SATU SEHAT. Ucapan terimakasih juga disampaikan kepada Politeknik Negeri Banyuwangi dan prodi Teknologi Rekayasa Perangkat Lunak (TRPL) telah membimbing dari sisi akademi untuk mendukung terlaksananya penelitian ini.

REFERENSI

- [1] Indonesia, Kementerian Kesehatan, Keputusan Menteri Kesehatan Nomor 128/Menkes/SK/II/2004 tentang Kebijakan Dasar Puskesmas, Jakarta: Kementerian Kesehatan Republik Indonesia, 2004
- [2] E. T. T.P. Tukan, L. Rizqolloh, and R. Dwi Anggraynie, "ANALISIS PENGGUNAAN APLIKASI SIMPUS DENGAN METODE TAM (TECHNOLOGY ACCEPTANCE MODEL) DI PUSKESMAS LAMPER TENGAH," *Med. TRADA*, vol. 4, no. 1, 2023, doi: 10.59485/jtemp.v4i1.29
- [3] Indonesia, Kementerian Kesehatan, Peraturan Menteri Kesehatan Republik Indonesia Nomor 24 Tahun 2022 tentang Rekam Medis Elektronik, Jakarta: Kementerian Kesehatan Republik Indonesia, 2022
- [4] Kementerian Kesehatan Republik Indonesia. (n.d.). Dokumentasi Satu Sehat Interoperability. SATUSEHAT. Diakses pada 30 Oktober 2024, dari <https://satusehat.kemkes.go.id/platform/docs>
- [5] V. Sarcar, "Use the DRY Principle," in *Simple and Efficient Programming with C#*, 2023. doi: 10.1007/978-1-4842-8737-8_5.
- [6] N. Jali, D. Greer, and P. Hanna, "Class Responsibility Assignment (CRA) for Use Case Specification to Sequence Diagrams (UC2SD)," in *2014 8th Malaysian Software Engineering Conference, MySEC 2014*, 2014. doi: 10.1109/MySec.2014.6985982
- [7] A. P. Wibawa, M. Diantoro, I. Idris, A. Purnomo, and N. C. Kurniawan, "Pengembangan Sistem Informasi Pengelolaan Konferensi Internasional Universitas Negeri Malang dengan Menggunakan Metode Waterfall," *J. Teknol. Sist. Inf. dan Apl.*, vol. 7, no. 1, pp. 352–361, 2024, doi: 10.32493/jtsi.v7i1.34974
- [8] R. Martin, "Design principles and design patterns," *Object Mentor*, no. c, pp. 1–34, 2000, [Online]. Available: http://www.cogs.susx.ac.uk/users/ctf20/dphil_2005/Photos/Principles

- _and_Patterns.pdf%5Cnhttp://scm0329.googlecode.com/svn-history/r78/trunk/book/Principles_and_Patterns.pdf
- [9] J. R. Coyanda, N. Ariati, H. Sunardi, and K. G. K. Ghazali, "Sistem Informasi Ekspedisi Barang Pada PT. New Power Global Energy Dengan Menggunakan Metode Extreme Programming," *J. Ilm. Inform. Glob.*, vol. 13, no. 2, 2022, doi: 10.36982/jiig.v13i2.2295
 - [10] G. E. Iyawa, "Personal extreme programming: Exploring developers' adoption," in *26th Americas Conference on Information Systems, AMCIS 2020*, 2020
 - [11] F. Darmawan, M. Ihsan Alfani Putera, and S. R. Natasia, "Rancang Bangun Sistem Informasi Pengelolaan Surat Menggunakan Metode Personal Extreme Programming (Studi Kasus: Kelurahan Sepinggan)," *SPECTA J. Technol.*, vol. 6, no. 1, 2022, doi: 10.35718/specta.v6i1.700
 - [12] Larman, C. (2004). *Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development*. Analysis (p. 736)
 - [13] E. Miranda, "Moscow Rules: A Quantitative Exposé," in *Lecture Notes in Business Information Processing*, 2022. doi: 10.1007/978-3-031-08169-9_2
 - [14] F. Ardiansyah, "Personal Extreme Programming Dengan Penentuan Prioritas Moscow," *UMM Institutional Repos.*, no. July, 2018.
 - [15] R. Setyadi, A. A. Rahman, and T. Anwar, "Evaluation of The Orthopedic Hospital Website 's Performance Using User Acceptance Testing," vol. 8, no. 1, pp. 65–70, 2025, doi: 10.15408/aism.v8i1.42951
 - [16] Clean Code: A Handbook of Agile Software Craftsmanship. (2009). Kybernetes, 38(6), 1035–1035. <https://doi.org/10.1108/03684920910973252>