BAB II
TINJAUAN PUSTAKA

2.1 Penelitian Terkait

Penelitian ini menyertakan beberapa penelitian yang telah dilakukan sebelumnya mengenai pengukuran estimasi biaya perangkat lunak menggunakan metode *Function Point*.

Pada penelitian yang dilakukan R. F. Taufiqurrahman dengan judul "Estimasi Pengukuran Perangkat Lunak Menggunakan Metode Matrik Function Point Analysis – Studi Kasus Aplikasi Akademik Siswa Berbasis Website" Penelitian ini mengidentifikasi masalah dalam estimasi biaya pemasaran dan waktu produksi perangkat lunak, menggunakan metode *Function Point Analysis* (FPA) sebagai metode penelitian. Metode pengujian yang diterapkan melibatkan analisis fungsionalitas aplikasi akademik siswa berbasis website, dengan hasil menunjukkan nilai Function Point sebesar 152,90, yang dapat digunakan untuk estimasi waktu, biaya, dan biaya manajemen perangkat lunak tersebut.[1]

"Metode *Function Point* Untuk Estimasi Biaya Proyek Pengembangan Aplikasi E-Pres Saja" oleh Nurul Amalia, penelitian ini bertujuan untuk mengatasi masalah penundaan dan over budget dalam proyek perangkat lunak dengan menggunakan metode *Function Point* (FP). Metode penelitian melibatkan studi literatur, wawancara untuk pengumpulan data kebutuhan fungsionalitas, dan analisis menggunakan lima komponen FP: *External Input, External Output, Internal Logical File, External Interface File, dan External Inquiry*. Pengujian dilakukan melalui perhitungan *Crude Function Point* (CFP) untuk menentukan kompleksitas sistem. Hasil penelitian

menunjukkan bahwa biaya pengembangan aplikasi E-Pres Saja diperkirakan sebesar Rp 49.025.482, memberikan pedoman akurat untuk estimasi biaya proyek perangkat lunak berbasis Android.[2]

Penelitian dari Kristi dengan judul "Estimasi Biaya *Software* FAS (*Financing Analysis System*) Menggunakan Metode *Function Point* (Studi Kasus Pada PT BPRS Lantabur Tebuireng)" bertujuan untuk mengatasi masalah penundaan, pembengkakan biaya, dan kegagalan proyek perangkat lunak akibat estimasi yang tidak akurat. Dengan menggunakan metode *Function Point*, penelitian ini mengukur fungsionalitas perangkat lunak berdasarkan lima komponen utama: *External Input, External Output, External Inquiry, Internal Logical File, dan External Interface File*. Data dikumpulkan melalui observasi dan wawancara, kemudian dilakukan penghitungan nilai kompleksitas setiap komponen untuk menghasilkan *Crude Function Point* (CFP) sebesar 634, dengan *effort* mencapai 6.168,94 orang/jam. Total estimasi biaya pengembangan perangkat lunak FAS ini adalah Rp 94.797.120. Hasil penelitian menunjukkan bahwa metode *Function Point* mampu memberikan estimasi biaya yang lebih akurat, sehingga membantu perusahaan perbankan seperti PT BPRS Lantabur Tebuireng dalam perencanaan anggaran yang lebih efisien.[3]

Penelitian dari Sandhea dengan judul “Estimasi Biaya Perangkat Lunak Pada Aplikasi SIBIMA Universitas XYZ dengan Menggunakan Metode *Function Point*" penelitian ini bertujuan untuk mengatasi masalah perencanaan yang buruk dalam pengembangan perangkat lunak yang sering menyebabkan kegagalan proyek. Sistem Informasi Bimbingan Akademik (SIBIMA) di Universitas XYZ dianalisis menggunakan metode *Function Point*, yang mengukur ukuran perangkat lunak

berdasarkan lima komponen utama: *External Input, External Output, External Inquiry, Internal Logical File, dan External Interface File*. Data dikumpulkan melalui observasi, wawancara, dan studi literatur, kemudian dianalisis untuk menghitung *Crude Function Point* (CFP) sebesar 41 yang disesuaikan dengan *Relative Complexity Adjustment Factor* (RCAF) sebesar 48, menghasilkan nilai *effort* sebesar 379,906 man/hour. Estimasi biaya pengembangan perangkat lunak SIBIMA adalah IDR 22.858.705. Penelitian ini membuktikan bahwa metode *Function Point* mampu memberikan estimasi biaya yang akurat, sehingga dapat dijadikan acuan untuk pengembangan perangkat lunak di institusi pendidikan tinggi.[4]

Serta penelitian yang dilakukan oleh Parlika dengan judul " Pengukuran Kualitas Perangkat Lunak Website Pendataan Ekskul Siswa Menggunakan *Function Point*" bertujuan untuk mengukur kualitas perangkat lunak dengan menggunakan metode *Function Point* untuk memberikan estimasi biaya dan ukuran fungsionalitas perangkat lunak. Website Pendataan Ekskul Siswa dianalisis dengan membagi fungsionalitas sistem menjadi lima komponen utama: *External Input, External Output, External Inquiry, Internal Logical File,* dan *External Interface Files*. *Crude Function Point* (CFP) dihitung berdasarkan kompleksitas masing-masing fitur, sementara *Relative Complexity Adjustment Factor* (RCAF) dinilai berdasarkan 14 parameter, menghasilkan total *Function Point* (FP) sebesar 86,4. Berdasarkan perhitungan, estimasi biaya pengembangan perangkat lunak untuk website ini adalah Rp 2.378.419. Penelitian ini menunjukkan bahwa metode *Function Point* efektif untuk mengukur kualitas perangkat lunak sekaligus memberikan estimasi biaya yang efisien.[5].

2.2 Landasan Teori

2.2.1 Sistem Informasi

Sistem informasi adalah kumpulan komponen yang saling berhubungan dan bekerja sama untuk mengumpulkan, memproses, menyimpan, dan mendistribusikan informasi guna mendukung pengambilan keputusan dan pengelolaan organisasi. Menurut Jogiyanto dalam Edy [6] , sistem informasi meliputi perangkat keras, perangkat lunak, data, prosedur, dan orang-orang yang terlibat dalam manajemen informasi. Informasi merupakan data yang telah diproses dan disusun sedemikian rupa agar memiliki makna lebih berarti, sehingga dapat digunakan untuk mendukung berbagai kegiatan dalam suatu organisasi.

Dalam konteks penerapan aplikasi atau sistem informasi untuk memantau kuliah di Universitas Kristen Indonesia Toraja, sistem informasi berfungsi sebagai penghubung utama, pusat data yang terintegrasi, serta alat otomasi yang mengaitkan semua pihak yang terlibat dalam lingkungan akademik, termasuk mahasiswa, dosen, admin/staff akademik, dan manajemen .

2.2.2 Pengukuran Perangkat Lunak

Menurut Institute of Electrical and Electronics Engineers (IEEE), pengukuran adalah ukuran kuantitatif dari sebuah sistem, komponen, atau proses yang memiliki atribut tertentu. Mengukur berarti menunjukkan angka terkait luasan, dimensi, dan kapasitas. Secara historis, pengukuran adalah kegiatan yang bertujuan untuk menentukan angka suatu objek secara sistematis, yang mencerminkan karakteristik objek tersebut. Dalam konteks pengukuran perangkat lunak, diperlukan satuan yang disebut metrik, yang dibagi menjadi dua kategori menurut Pressman[7]:

1. **Pengukuran secara langsung (*Direct Metric*)**: Ini adalah metrik yang berhubungan langsung dengan objek perangkat lunak. Contohnya meliputi jumlah baris kode (*Line of Code*/LOC), waktu kinerja, penggunaan memori, dan jumlah kesalahan dalam periode tertentu.
2. **Pengukuran secara tak langsung (*Indirect Metric*)**: Ini adalah metrik yang diperoleh dari interaksi perangkat lunak dengan lingkungan sekitarnya. Contoh metrik ini mencakup kualitas, fungsionalitas, efisiensi, kompleksitas, dan reliabilitas.

Umumnya, pengukuran langsung lebih mudah diimplementasikan dibandingkan dengan pengukuran tidak langsung karena hasilnya dapat diperoleh secara langsung. Sebaliknya, pengukuran tidak langsung memerlukan proses yang lebih kompleks untuk menghitung dan mengumpulkan informasi yang diperlukan

Pengukuran perangkat lunak adalah proses yang digunakan untuk menentukan kualitas, ukuran, dan kompleksitas perangkat lunak. Metode pengukuran ini penting untuk memastikan bahwa perangkat lunak memenuhi standar kualitas yang diharapkan dan dapat berfungsi dengan baik dalam lingkungan operasionalnya. Terdapat berbagai metode pengukuran kualitas perangkat lunak yang telah dikembangkan, seperti *Constructive Cost Model (COCOMO)* dan *Use Case Points*, yang masing-masing memiliki pendekatan dan kriteria evaluasi yang berbeda.

COCOMO adalah salah satu metode estimasi biaya perangkat lunak yang paling banyak digunakan. Metode ini pertama kali diperkenalkan oleh Barry Boehm dan menggunakan rumus regresi untuk memperkirakan usaha, waktu, dan sumber daya yang dibutuhkan untuk menyelesaikan proyek perangkat lunak berdasarkan ukuran

kode (KLOC) (Nizar & Perdanakusuma [8]. *COCOMO* membantu manajer proyek dalam merencanakan anggaran dan sumber daya dengan lebih efektif, sehingga mengurangi risiko kesalahan estimasi.

Di sisi lain, *Use Case Points* adalah metode pengukuran yang berfokus pada fungsionalitas sistem berdasarkan use case yang diidentifikasi dalam analisis sistem. Metode ini mengukur kompleksitas sistem dengan mempertimbangkan faktor-faktor seperti jumlah aktor, jumlah use case, dan tingkat kompleksitas masing-masing u*se case* (Fathoni, 2019). Dengan pendekatan ini, pengembang dapat memperkirakan effort dan biaya pengembangan perangkat lunak secara lebih akurat.

Pengukuran perangkat lunak adalah proses penting dalam rekayasa perangkat lunak untuk memperkirakan ukuran komponen perangkat lunak yang menjadi dasar dalam perencanaan manajemen proyek lainnya. Pada aplikasi monitoring perkuliahan, pengukuran ini berfungsi untuk menghitung ukuran fungsional perangkat lunak sebagai langkah awal dalam estimasi *effort* dan biaya yang diperlukan untuk pemeliharaan maupun pengembangan. Perencanaan perangkat lunak yang efektif memerlukan estimasi yang mencakup jadwal pengerjaan, sumber daya, dan risiko untuk menghindari kegagalan proyek perangkat lunak[9].

.Dalam konteks sistem informasi, salah satu metode yang sering digunakan untuk pengukuran perangkat lunak adalah *Function Point*. Metode ini mengukur ukuran perangkat lunak berdasarkan fungsionalitas yang ditawarkan kepada pengguna, seperti input, output, dan penyimpanan data. Menurut ISPA dalam Anggi (2019), kegagalan proyek perangkat lunak sering kali disebabkan oleh ketidakmampuan mengukur proyek secara akurat, menentukan kebutuhan tenaga kerja yang sesuai, serta

kurangnya definisi kebutuhan perangkat lunak. Dengan menerapkan *Function Point*, ukuran perangkat lunak dapat dihitung secara lebih objektif, sehingga mendukung estimasi *effort* dan biaya yang realistis untuk pengembangan dan pemeliharaan sistem informasi.Mengukur ukuran perangkat lunak pada tahap awal sangat penting untuk memastikan estimasi *effort* yang realistis. Hal ini dapat diterapkan pada aplikasi monitoring perkuliahan untuk memberikan perencanaan anggaran yang lebih akurat dan mendukung pengambilan keputusan dalam pengelolaan sistem. Estimasi yang akurat tidak hanya mengurangi risiko kegagalan proyek, tetapi juga memastikan bahwa sumber daya yang digunakan sesuai dengan kebutuhan aktual[10].

2.2.3 Metode *Function Point Analysis*

Metode *Function Point Analysis* pertama kali diperkenalkan oleh Allan Albrecht pada tahun 1979 dalam publikasi berjudul "*Measuring Application Development Productivity*" di IBM. Pada tahun 1986, *International Function Point User Group* (IFPUG) didirikan dan hingga kini telah menerbitkan beberapa versi panduan untuk perhitungan *Function Point*.

Function Point adalah suatu metode yang membagi sistem menjadi bagian-bagian yang lebih kecil, sehingga memudahkan pemahaman dan analisis (Longstreet, 2002). Metode ini mengadopsi pendekatan yang berfokus pada fungsionalitas dan kompleksitas untuk memperkirakan ukuran perangkat lunak selama proses pengembangan[11].

FPA adalah sebuah teknik terstruktur dalam memecahkan masalah dengan cara memecah sistem menjadi komponen yang lebih kecil dan menetapkan beberapa karakteristik dari sebuah perangkat lunak sehingga dapat lebih mudah dipahami dan

dianalisis. Singkatnya, metode FPA menyediakan tujuan, ukuran perbandingan yang membantu dalam evaluasi, perencanaan, pengelolaan, dan pengendalian produksi perangkat lunak.

Penelitian ini menggunakan metode FPA karena metode tersebut mengukur perspektif fungsional dari perangkat lunak yang akan dibangun, mengabaikan bahasa pemrograman, dan merupakan metode development atau platform perangkat keras yang digunakan. Metode ini perhitungannya didasarkan pada ukuran dan kompleksitas fungsi yang diinginkan dalam proyek perangkat lunak. Dibandingkan dengan pendekatan berbasis ukuran baris (*Line Of Code* atau biasa disebut LOC), pendekatan FP lebih independen terhadap bahasa pemrograman sehingga bisa diterapkan pada jenis aplikasi yang berbeda baik aplikasi database yang non-procedural, sistem informasi berbasis website maupun aplikasi penghitungan, misalnya payroll. Pendekatan ini juga lebih mudah diprediksi daripada LOC karena parameternya dihitung berdasarkan data yang lebih diketahui.

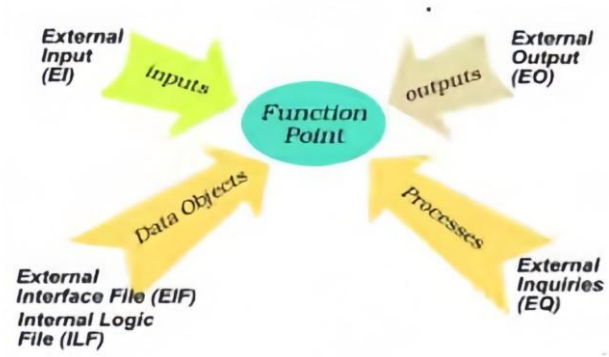
Metode FPA dianggap memiliki berbagai keunggulan jika dibandingkan dengan metode yang lainnya, seperti mampu menyediakan perkiraan volume proyek, perkiraan dapat disiapkan pada tahap pra-proyek, kehandalan metode relatif tinggi karena berbasis dokumen spesifikasi *requirement* dan lebih mudah dipahami pengguna non-teknis . Sehingga dengan penggunaan metode ini diharapkan pengembang proyek mampu memperkirakan proyeknya lebih awal dan lebih cepat untuk meminimalisir resiko kegagalan proyek[12].

2.2.3.1 Konsep Metode FPA

Metode FPA berbasis observasi dari perangkat lunak nyata mempunyai spesifik transaksional dan data karakteristik, tipe fungsi transaksional pada FP didasarkan pada konsep proses dasar. Proses yang mengirimkan data ke pengguna dapat dikelompokan dalam 4 langkah:

1. Menerima *request* dari pengguna
2. Membaca informasi yang dibutuhkan
3. Membuat penghitungan
4. Mengirim kembali hasil ke pengguna

Dalam metode FPA, ukuran sebuah sistem dapat dihitung dengan beberapa komponen seperti pada Gambar 2.1.



Gambar 2. 1 Komponen *Function Point*

External Input (EI) adalah proses dasar yang memproses data dan informasi kontrol yang datang dari luar batasan aplikasi. Data digunakan untuk memelihara satu atau lebih berkas logika internal. dapat untuk mengubah perilaku sistem.

External Output (EO) adalah sebuah proses dasar dimana hasil data dilewatkan dari dalam ke keluar dari batasan aplikasi. EO meng-*update Internal Logic*

File (ILF). Laporan dan file tersebut dibuat dari satu atau lebih ILF dan *External Interface File* (EIF).

External Inquiry (EQ) memiliki fungsi utama menyediakan informasi ke pengguna melalui pengambilan atau pemrosesan data atau informasi kontrol dari ILF/EIF. Layar beberapa tipe laporan dan pencarian merupakan komponen yang tepat untuk EQ. Tidak meng-*update* ILF atau EIF, hanya mengambil data untuk ditampilkan.

ILF adalah kelompok data atau kelompok informasi kontrol yang digunakan dalam aplikasi. Peran utama ILF adalah menyimpan data yang dipelihara oleh satu atau lebih proses dalam aplikasi.

2.2.3.2 Tahap Metode FPA

FPA harus dilakukan oleh orang terlatih dan berpengalaman dalam s*oftware development*. Hal ini dikarenakan dalam memberikan nilai-nilai dari setiap komponen *function point* itu bersifat subjektif dan akan wajar apabila hasil perhitungan *function point* seseorang akan berbeda dengan yang lain.

Pengerjaan FPA harus dimasukkan sebagai bagian dari rencana proyek secara keseluruhan. Artinya harus dijadwalkan dan direncanakan pengerjaannya sehingga dalam mengimplementasikan metode FPA, sebaiknya melakukan langkah-langkah seperti berikut:

1. Tahap 1: Menghitung *Crude Function Point* (CFP) CFP adalah untuk menghitung bobot nilai dari komponen-komponen *function point* yang dikaitkan dengan perangkat lunak yang akan dibuat. Komponen komponen FP terdiri dari 5 buah, yaitu sebagai berikut.

a. Tipe *Input*, berkaitan dengan *interface* yang dilakukan pengguna dalam memasukan data pada aplikasi.

b. Tipe *Output*, berkaitan dengan output yang dihasilkan aplikasi untuk pengguna yang dapat berupa laporan dicetak atau yang ditampilkan pada layar.

c. Tipe *Query/Search/View*, berkaitan dengan *query* terhadap data yang tersimpan.

d. Tipe File/Tabel/*Database*, berkaitan dengan *logic* penyimpan data yang dapat berupa file atau semacam database relasional.

e. Tipe *Interface External*, berkaitan dengan komunikasi data pada parangkat/mesin yang lain..

Selanjutnya setiap tipe komponen tersebut diberikan bobot berdasarkan kompleksitasnya, Total *Crude Function Point* (CFP) didapat dari menghitung jumlah 5 fungsi pada tahap metode FPA dikali dengan bobot masing-masing fungsi sesuai dengan tabel di bawah ini.

Tabel 2. 1 Bobot Kompleksitas FP

Tipe Komponen	*Bobot Level Kompleksitas FP*			*Total CFP*
	Low	*Average*	*High*	
Tipe Input (EI)	***3***	***4***	***6***	***?***
Tipe Output (EO)	***4***	***5***	***7***	***?***
Tipe Query/Search/View (EQ)	***3***	***4***	***6***	***?***
Tipe File/Table/ Database (ILF)	***7***	***10***	***15***	***?***
Tipe Interface Eksternal (EIF)	***5***	***7***	***10***	***?***

Sumber : International Function Point Users Group (IFPUG)

Nilai-nilai bobot dari setiap komponen diatas adalah ketetapan atau konstanta yang dibuat oleh *International Function Point User Group (IFPUG)*[13]. Setelah menghitung total CFP, maka untuk mendapatkan *adjusted* FP dibutuhkan nilai masing masing faktor yang mempengaruhi FP. Faktor-faktor yang mempengaruhi nilai FP dapat dilihat pada Tabel 2.1. Derajat kompleksitas pada penelitian ini mengacu pada standar International *Function Point Users Group* (IFPUG) versi 4.3.1, dengan mengukur fungsi berdasarkan kombinasi jumlah DET *(Data Element Types)* dan FTR *(File Types Referenced)* atau RET *(Record Element Types*) sesuai komponen F*unction Point* yang dianalisis seperti berikut.

Tabel 2. 2 Bobot Kompleksitas Untuk ILF dan EIF

No	***Number of Record Types (RET)***	***Data Element Types (DET)***		
		1 – 19	**20 – 50**	**> 50**
1	1	Low	Low	Average
2	2 to 5	Low	Average	High
3	>5	Average	High	High

Tabel 2. 3 Bobot Kompleksitas untuk *Input*

No	***Number of File Types Referenced (FTR)***	***Data Element Types (DET)***		
		1 – 4	**5 – 15**	**>15**
1	0 or 1	Low	Low	Average
2	2	Low	Average	High
3	>2	Average	High	High

Tabel 2. 4 Bobot kompleksitas untuk *Output* dan *Inquiry*

No	***Number of* File Types Referenced (FTR)**	***Data Element Types (DET)***		
		1 – 5	**6 – 19**	**>19**
1	0 or 1	Low	Low	High
2	2 or 3	Average	High	High
3	>3	High	High	High

2. Tahap 2: Menghitung *Relative Complexity Adjustment Factor* (RCAF), RCAF digunakan untuk menghitung bobot kompleksitas dari perangkat lunak berdasarkan 14 karakteristik. Penilaian kompleksitas memiliki skala 0 s/d 5 di mana 0 berarti sangat sederhana dan 5 yang berarti sangat kompleks atau rumit. Berikut adalah penilaiannya.

   0 = Tidak Pengaruh

   1= Insidental

   2= Moderat

   3 = Rata-rata

   4 = Signifikan

   5 = Essential

Tabel 2. 5 Karakter Perangkat Lunak

No	Karakteristik	Bobot
1	Tingkat Kompleksitas Komunikasi Data	[0/1/2/3/4/5]
2	Tingkat Kompleksitas Pemrosesan Terdistribusi	[0/1/2/3/4/5]
3	Tingkat Kompleksitas *Performance*	[0/1/2/3/4/5]
4	Tingkat Kompleksitas Konfigurasi	[0/1/2/3/4/5]
5	Tingkat Frekuensi Penggunaan Perangkat Lunak	[0/1/2/3/4/5]
6	Tingkat Frekuensi *Input* Data	[0/1/2/3/4/5]

7	Tingkat Kemudahan Penggunaan Bagi Pengguna	[0/1/2/3/4/5]
8	Tingkat Frekuensi Update Data	[0/1/2/3/4/5]
9	Tingkat Kompleksitas *Processing* Data	[0/1/2/3/4/5]
10	Tingkat Kemungkinan Penggunaan Kembali/*Reusable* Kode Program	[0/1/2/3/4/5]
11	Tingkat Kemudahaan dalam Instalasi	[0/1/2/3/4/5]
12	Tingkat Kemudahaan Operasional perangkat lunak (*backup, recovery*, dan sebagainya)	[0/1/2/3/4/5]
13	Tingkat Perangkat Lunak Dibuat untuk Multi Organisasi/Perusahaan/Klien	[0/1/2/3/4/5]
14	Tingkat Kompleksitas dalam Mengikuti Perubahaan/Fleksibel	[0/1/2/3/4/5]
Total		?

Karakteristik 14 poin merupakan ketetapan atau konstanta yang dibuat oleh IFPUG.

3. Tahap 3: Menghitung FP Adalah proses melakukan perhitungan untuk mendapat nilai *function point* dari perangkat lunak yang akan dibangun. Rumus yang digunakan yaitu persamaan berikut ini :

   $$FP = CFP \times (0.65 + (0.01 \times RCAFF)) \quad (1)$$

   Angka 0.65 dan 0.01 adalah ketetapan atau konstanta yang dibuat oleh IFPUG[14].

2.2.4 Estimasi *Effort* (Usaha)

Salah satu elemen terpenting dalam tahap perencanaan pembuatan proyek perangkat lunak adalah estimasi, yaitu dalam hal waktu dan sumber daya. Estimasi adalah pengukuran yang didasarkan pada hasil kuantitatif dan dapat diukur berdasarkan tingkat akurasinya. Pembuatan jadwal dan anggaran yang sesuai merupakan komponen penting estimasi dalam perencanaan proyek. Proses estimasi jumlah karyawan dan jumlah waktu sebenarnya yang diperlukan untuk menyelesaikan proyek perangkat lunak dikenal sebagai estimasi upaya perangkat lunak. Meskipun

estimasi tidak mungkin menghasilkan temuan yang sangat akurat, ada sejumlah teknik yang dapat digunakan untuk mengurangi ketidakakuratan tergantung pada proyek yang diestimasi. Pendekatan *Function Point Analysis* (FPA) akan digunakan dalam perhitungan studi ini untuk memperkirakan upaya. [15].

a. Rumus Estimasi *Effort*

$$Effort = FP \times 9 \quad (2)$$

Rumus ini digunakan untuk memperkirakan jumlah waktu kerja total (dalam jam) yang dibutuhkan untuk mengembangkan sistem berdasarkan nilai *Function Point* (FP). Nilai konversi 9 jam per FP diperoleh dari hasil wawancara dengan developer sistem.

b. Rumus Estimasi Waktu Kerja

$$WaktuKerja = Effort / JamKerjaPerHari \quad (3)$$

Rumus ini mengonversi estimasi *effort* menjadi estimasi waktu kerja dalam satuan hari kerja, berdasarkan jumlah jam kerja efektif per hari.

c. Rumus Estimasi Biaya Total

$$Cost = Effort \times UpahPerJam \quad (4)$$

Rumus ini digunakan untuk menghitung total estimasi biaya pembuatan perangkat lunak berdasarkan estimasi waktu kerja (*effort*) dan tarif upah per jam dari tenaga kerja.

2.2.5 Standar Biaya Tenaga Kerja Konsultan IT (INKINDO)

Dalam proses estimasi biaya pembuatan aplikasi, salah satu unsur utama yang harus diperhitungkan adalah besaran upah tenaga kerja. Penelitian ini menggunakan

acuan dari Pedoman Remunerasi/Biaya Personil (*Billing Rate*) yang diterbitkan oleh Ikatan Nasional Konsultan Indonesia (INKINDO) Tahun 2025, sebagai standar resmi dalam menentukan biaya jasa profesional. Pedoman ini menyajikan tarif minimal yang berlaku untuk berbagai kategori tenaga ahli berdasarkan kualifikasi pendidikan, tingkat pengalaman, serta wilayah kerja.

Untuk kategori Teknisi/Analis Sub-Profesional, yaitu pekerja dengan latar belakang pendidikan minimal S1 dan pengalaman kerja terbatas, standar remunerasi di wilayah DKI Jakarta tercantum sebesar Rp16.500.000,00 per bulan. Jumlah ini kemudian disesuaikan dengan indeks provinsi guna menghitung standar gaji di daerah lain. Sebagai contoh, Sulawesi Selatan memiliki indeks penyesuaian sebesar 0,970, sehingga gaji disesuaikan menjadi sekitar Rp16.005.000,00 per bulan[16].

Nilai tersebut kemudian dikonversi menjadi tarif per jam dengan membagi jumlah gaji bulanan dengan jumlah hari dan jam kerja dalam satu bulan. Selain itu, proses konversi ini juga memperhitungkan faktor biaya lainnya seperti beban operasional, keuntungan jasa, dan biaya tidak langsung. Nilai akhir inilah yang digunakan untuk memperkirakan total biaya tenaga kerja berdasarkan estimasi waktu kerja (*effort*) yang diperoleh melalui metode *Function Point Analysis* (FPA).

2.2.6 Estimasi Biaya

Setelah nilai *effort* dihitung, proses estimasi biaya bertujuan untuk menentukan total anggaran yang diperlukan untuk pembuatan dan pemeliharaan perangkat lunak. Estimasi ini mencakup biaya tenaga kerja, infrastruktur, serta pengeluaran lain yang terkait dengan keberlanjutan sistem. Perhitungan dilakukan dengan mengalikan nilai *effort* (dalam *man-hour*) dengan rata-rata upah tenaga kerja per jam. Selain itu, biaya

pemeliharaan rutin juga dipertimbangkan untuk memastikan keberlanjutan operasional perangkat lunak[17].

Sebagai acuan, penelitian Sandhea et al. dalam pengembangan Sistem Informasi Bimbingan Akademik (SIBIMA) menggunakan metode *Function Point* menunjukkan bahwa *effort* sebesar 379,906 *man-hour* dengan tingkat produktivitas tertentu menghasilkan estimasi biaya sebesar IDR 22.858.705.

2.3 Kerangka Pikir

Gambar 2. 2 Kerangka Pikir