BAB II
TINJAUAN PUSTAKA

2.1. Studi Literatur

Studi literatur dilakukan penulis untuk mencari refrensi yang relevan terhadap topik penulis. Refrensi bisa didapat dari buku, jurnal, artikel, laporan, dan lain-lainnya untuk memperkuat landasan teori penulis

2.2. Estimasi Perangkat Lunak

Estimasi perangkat lunak merupakan kegiatan pengukuran/perkiraan terhadap biaya, effort, waktu yang dibutuhkan dari suatu pengembangan proyek perangkat lunak. Banyaknya kegagalan proyek perangkat lunak biasanya dikarenakan kurang matangnya perencanaan dan manajemen risiko yang dilakukan oleh pengembang[11]. Pengembang harus bisa memanajemen risiko agar dapat melihat faktor-faktor apa saja yang dapat mempengaruhi kegagalan dalam pengembangan perangkat lunak dan membuat solusi terhadap faktor-faktor tersebut[12]. Ada beberapa faktor utama yang biasanya menyebabkan kegagalan dalam pengembangan perangkat lunak yaitu [13] :

1. Estimasi biaya dan waktu yang kurang tepat
2. Manajemen risiko yang lemah
3. Penggunaan metode estimasi yang kurang tepat.

2.2.1 Metode Estimasi Perangkat Lunak

Terdapat banyak sekali metode estimasi perangkat lunak yang telah dikembangkan oleh berbagai macam orang. Metode estimasi perangkat lunak ini dibagi menjadi dua yaitu :

2.2.1.1. Algorithmic

Metode *algorithmic* ini menggunakan teknik estimasi yang bergantung pada data-data dari aplikasi terdahulu seperti lama pengerjaan, jumlah orang yang mengerjakan, dan biaya pengerjaan. Dikarenakan bergantung pada aplikasi terdahulu, keunggulan metode yang menggunakan teknik algoritma ini ialah bersifat lebih objektif[14], cocok untuk proyek besar atau kompleks, dan bisa digunakan sejak dari awal perkembangan perangkat lunak

Namun metode ini juga memiliki kelemahan yaitu data yang digunakan harus akurat, kurang fleksibel jika terjadi perubahan dalam pengembangan perangkat lunak karena dapat meningkatkan *effort* dan harus dilakukan penghitungan ulang[15]., dan kurang cocok untuk proyek-proyek kecil yang sifatnya *agile*[16].

2.2.1.2. Non-algorithmic

Metode *non-algoritmic* ini menggunakan pendekatan yang bersifat Subjektif karena tidak menggunakan data ataupun rumus dalam penghitungan estimasinya, hanya menggunakan analogi dari pakar ahli dan perbadingan dengan proyek terdahulu untuk perkiraan saja sehingga bisa menghasilkan perhitungan yang lebih cepat dan fleksibel dalam pengembangan perangkat lunak[17].

Dikarenakan perhitungan estimasi yang bersifat subjektif, metode ini memiliki kekurangan yaitu bisa terjadinya perbedaan pendapat diantara para pakar ahli.

2.2.2 *Function Point Analysis*

Pengukuran estimasi dengan metode *Function Point Analysis* ini dibuat dengan memikirkan kemampuan *Problem Solving* manusia, dimana manusia senang memecahkan masalah sulit dengan cara membuat memecah msalah sulit tersebut menjadi poin-poin kecil yang lebih mudah untuk dipahami. Memberi klasifikasi terhadap poin-poin tersebut dan memberi kategori terhadap poin-poin tersebut. *Function Point Analysis* ini adalah metode yang

memecah komponen-komponen yang ada dalam suatu sistem. Ketika komponen-komponen tersebut sudah dipecah menjadi bagian-bagian yang lebih kecil, maka komponen tersebut akan jadi lebih mudah untuk di dianalisis dan dipahami[18].

Metode *Function Point Analysis* ini pertama kali dikembangkan oleh Allan Albrecht pada tahun 1979 dan sampai saat ini masih diperbarui dan dikembangkan oleh *International Function Point User Group* (IFPUG). Dalam metode *Funtion Point Analysis* (FPA) ini terdapat 5 fungsi yang digunakan sebagai parameter dalam pengukuran estimasi perangkat lunak yaitu :

1. *External Input* (EI)

   External Input adalah data yang masuk dari luar sistem yang berasal dari user atau sistem/aplikasi lainnya yang kemudian data tersebut akan disimpan kedalam *Internal Logical File* pada sistem tersebut
2. *External Output* (EO)

   External Output adalah data yang telah tersimpan didalam *Internal Logical File* kemudian ingin dikeluarkan kepada sistem lainnya berupa laporan sistem atau bisa berupa output data yang sudah melalui algoritma sistem.
3. *External Inquiry* (EQ)

   External Inquiry adalah proses yang dimana data *input* dan *output* yang dihasilkan dan tersimpan dalam *Internal Logical File* diambil untuk ditampilkan sistem tanpa merubah data yang ada dalam *Internal Logical Files*.
4. *Internal Logical File* (ILF)

   Internal Logical Files adalah kelompok data atau informasi yang tersimpan didalam sebuah sistem itu sendiri.
5. *External Interface File* (EIF)

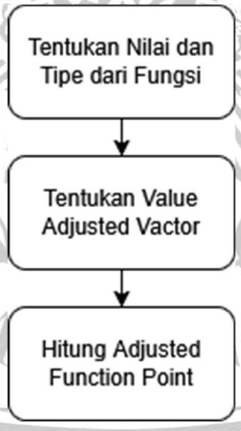
   External Interface File adalah kelompok data atau informasi yang dirujuk oleh sistem tetapi didapat dari sistem lain.

Fungsi yang sudah diklasifikasikan selanjut harus ditentukan kompleksitas tiap fungsinya. Kompleksitas tiap fungsi dibagi menjadi 3 yaitu *low*, *medium*, dan *high*[19]. Tingkat kompleksitas memiliki nilainya masing-masing seperti pada tabel berikut :

Tabel 1. Bobot User Function Type

User Function Type	**Complexity Weight**		
	Low	**Medium**	**High**
External Input (EI)	3	4	6
External Output (EO)	4	5	7
External Inquiry (EQ)	3	4	6
Internal Logical File (ILF)	7	10	15
External Interface File (EIF)	5	7	10

Alur penghitungan metode *Function Point Analysis* tertera pada gambar berikut :



Gambar 1. Alur Perhitungan Metode Function Point Analysis

Ketika kita telah mengetahui nilai dan tipe dari sebuah *User Function* maka selanjutnya adalah penentuan *Value Adjusted Factor.* VAF adalah faktor yang mempengaruhi Tingkat kesulitan implementasi dari sebuah sistem. Penilaian dari faktor tersebut dibagi dari skala 0-5 dimana :

0 : tidak ada, atau tidak berpengaruh terhadap sistem

1 : berpengaruh kecil

2 : berpengaruh sedang

3 : berpengaruh cukup signifikan

4 : berpengaruh signifikan

5 : berpengaruh sangat penting

VAF memiliki 14 faktor yang ditampilkan pada table berikut :

Tabel 2. Value Adjusted Factor

ID	Factor	Deskripsi
C1	Data communications	Berapa banyak fasilitas komunikasi yang ada untuk membantu mentransfer atau bertukar informasi dengan aplikasi atau sistem?
C2	Distributed function	Bagaimana data terdisribusi dan fungsi pemrosesan ditangani?
C3	Performance objectives	Apakah sistem memiliki kebutuhan performa tinggi, seperti respon cepat atau beban tinggi?
C4	Heavily used configuration	Apakah sistem berjalan pada konfigurasi perangkat keras atau OS yang kompleks?
C5	Transaction rate	Seberapa sering transaksi (input/output tinggi)?

C6	On-line data entry	Berapakah persentase infomasi yang dimasukkan secara online?
C7	End-user efficiency	Apakah aplikasi dirancang untuk efisiensi enduser?
C8	On-line update	Apakah data dapat diperbarui langsung di sistem secara online?
C9	Complex processing	Apakah sistem melakukan logika pemrosesan yang kompleks seperti extensive logical atau mathematical processing?
C10	Reusability	Apakah komponen sistem dapat digunakan kembali untuk proyek atau modul lain
C11	Installation ease	Seberapa mudah sistem dipasang/dikonfigurasi di lingkungan baru.
C12	Operational ease	Seberapa mudah sistem dioperasikan (termasuk backup, recovery, restart).
C13	Multiple sites	Apakah sistem digunakan di beberapa lokasi atau organisasi berbeda.
C14	Facilitate change	Seberapa fleksibel sistem menghadapi perubahan di masa depan

Ketika kita sudah menentukan nilai dari 14 faktor tersebut, total nilai dari 14 faktor tersebut adalah *Total Degree Influence* (TDI). Perhitungan terakhir dari metode estimasi *Function Point Analysis* adalah penentuan *Function Point* (FP) dengan rumus sebagai berikut :

$$FP = UFP \times VAF$$

Untuk mendapatkan nilai *Unadjusted Function Point* (UFP) kita harus mengkategorikan tiap *User Function* dan memberinya bobot kemudian masukan ke rumus pada table berikut :

Tabel 3. Unadjusted Function Point (UFP)

User Function Type	**Complexity Weight**			
	Low	**Medium**	**High**	**Total**
External Input (EI)	…x3 = …	…x4 = …	…x6 = …	
External Output (EO)	…x4 = …	…x5 = …	…x7 = …	
External Inquiry (EQ)	…x3 = …	…x4 = …	…x6 = …	
Internal Logical File (ILF)	… x7 = …	… x10 = …	…x15 = …	
External Interface File (EIF)	… x5= …	…x7 = …	…x10 = …	
Total Unadjusted Function Point				

Untuk mendapatkan nilai *Value Adjusted Factor* (VAF) kita harus memberi bobot nilai kepada 14 faktor VAF kemudian menjumlah total nilai tersebut untuk mendapatkan nilai *Total Degree Influence* (TDI) kemudian masukan ke rumus berikut :

$$VAF = 0.65 + 0.01 \times TDI$$

Tabel 4. Total Degree Influence (TDI)

ID	Factor	Nilai
C1	Data communications	
C2	Distributed function	
C3	Performance objectives	
C4	Heavily used configuration	
C5	Transaction rate	
C6	On-line data entry	
C7	End-user efficiency	
C8	On-line update	
C9	Complex processing	
C10	Reusability	
C11	Installation ease	
C12	Operational ease	
C13	Multiple sites	
C14	Facilitate change	
	Total Degree Influence	=

2.2.3 *Low-Code Development Platform* (LCDP)

Low-Code Development Platform (LCDP) adalah hasil dari peningkatan teknologi yang pesat karena memudahkan penggunanya untuk menyelesaikan aplikasi atau sistem dengan cepat sehingga tidak banyak memakan biaya. LCDP menawarkan produktifitas tinggi dan deliver yang cepat bahkan walaupun yang menggunakan adalah orang yang minim background IT atau bisa diesebut sebagai *Citizen Developer*, karena konsep dari LCDP ini ialah *drag-and-drop* terhadap visual model daripada proses tradisional coding. Dikarenakan kemudahannya LCDP ini mengubah bagaimana aplikasi atau sistem yang digunakan sebuah perusahaan dibuat.

Dalam studi [20] rata-rata *delivery rate* dari proyek LCDP adalah 1,8 jam per-fungsi dibandingkan dengan proyek yang masih menggunakan metode

tradisional adalah 8,3 jam. Perbandingan tersebut menunjukkan bahwa teknologi LCDP ini menunjukkan produktifitas tinggi yang membuat LCDP ini menjadi populer dan mendominasi pengembangan perangkat lunak[21].

2.2.4 Estimasi Waktu dan Biaya

2.2.4.1. Estimasi Waktu

Platform *Low-Code* memiliki Tingkat produktifitas tinggi dengan perbandingan 3-10x lipat dari pengembangan tradisional [21]. Dari studi yang dilakukan pada pengembangan LCDP didapat kecepatan pengerjaan dari pengguna adalah 25 *Function Point* (FP) perbulan jika yang mengerjakan adalah pengembang yang belum menguasai LCDP sepenuhnya ataupun *Citizen Developer* yang kurang memiliki pengetahuan teknis dibidang pengembangan perangkat lunak. Maka untuk mendapatkan estimasi waktu dibutuhkan total FP dari fungsi-fungsi yang telah dibobot kemudian dibagi dengan produktifitas dari 1 pengembang LCDP yang dirumuskan sebagai :

$$Waktu = FP \div 25\ FP/\text{dev/bulan}$$

2.2.4.2. Estimasi Biaya

Menurut beberapa websites, gaji seorang *developer* memiliki banyak variasi tergantung dari Lokasi, tingkat keahlian dan Perusahaan bekerja. Setelah mendapat hasil rata-rata dari berbagai variasi penulis mengambil rata-rata Rp.8.600.000. untuk mendapatkan hasil estimasi biaya dibutuhkan rumus sebagai berikut :

$$Biaya = Waktu \times 8.600.000/\text{dev/bulan}$$

2.3. Kajian Penelitian Terdahulu

Penilitian terdahulu digunakan untuk menjadi acuan oleh penulis serta membantu penulis agar dapat lebih mendalami landasan teori penulis. Oleh karena itu penulis menyertakan beberapa penelitian sebelumnya yang ditampilkan pada Tabel 1 Berikut :

Tabel 5. Kajian Penelitian Terdahulu

No.	Penulis	Tahun	Judul	Hasil
1.	Nur Rachmat dan Saparudin	2017	Estimasi Ukuran Perangkat Lunak Menggunakan Function Point Analysis - Studi Kasus Aplikasi Pengujian dan Pembelajaran Berbasis Web	Peneliti mendapatkan hasil estimasi dengan tingkat akurasi yang tinggi. [6]
2.	Reinindhi Puspitasari, Dian Nugraini dan Renny Sari Dewi	2019	*Estimasi Biaya Perangkat Lunak Menggunakan Metode Function Point Analysis (Studi Kasus: Sistem BKD Universitas XYZ)*	Peneliti mendapatkan hasil estimasi yang diperlukan jika perguruan tinggi ingin membuat aplikasi BKD (Beban Kinerja Dosen) untuk menunjang proses bisnisnya adalah Rp.42,181,198 [7].
3.	Rizky Parlika, Rahmadany Fahreza	2022	ESTIMASI PENGUKURAN	Peneliti mendapat hasil estimasi dari Aplikasi Akademik

No.	Penulis	Tahun	Judul	Hasil
	Taufiqurrahman, Hafi Ihza Farhana, Rahmat Dimas Syahputra dan Fairuz Aldifa		PERANGKAT LUNAK MENGGUNAKAN METODE MATRIK FUNCTION POINT ANALYSIS – STUDI KASUS APLIKASI AKADEMIK SISWA BERBASIS WEBSITE	Siswa Berbasis Website (AASBW) sebesar Rp. 7,645,000 untuk biaya pemasaran dan biaya pengembangan sebesar Rp. 3,850,000 yang dikerjakan selama 29 hari[8].
4.	Mohamad Doris Sambodo Putro, Denny Agung Situmerang dan Hans Kristian Putra Fajar	2023	Pengukuran Perangkat Lunak Menggunakan Function Point Analysis – Studi Kasus Software Akuntansi "Beecloud"	Peneliti mendapatkan hasil pengukuran dari dari software akuntasi "Beecloud" merupakan aplikasi kompleks dan menyimpulkan bahwa range harga software dari Rp.130,900 – 421,300 perbulan sudah layak digunakan [9].
5.	Nurul Amalia, Nur Ika Royanti, Indrayanti	2024	METODE FUNCTION POINT UNTUK ESTIMASI BIAYA PROYEK PENGEMBANGAN	Peneliti mendapatkan hasil bahwa metode *Function Point Analysis* terbukti mampu digunakan sebelum proses

No.	Penulis	Tahun	Judul	Hasil
			APLIKASI E-PRES SAJA	pengembangan dimulai, sehingga bisa menghindari kendala biaya seperti *over/under budget* dan dapat mengatur alokasi sumber daya dengan lebih baik[10].