

Panduan Pengukuran Perangkat Lunak Metode Function Point Serta Implementasinya pada Website Pemesanan Tiket Bus

Rizky Parlika^{1*}, Ahmad Maghfur², Ali², Ardiana Deka Maharani³,
Bregsi Atingsari Julastri⁴, Syalum Marsya Pruista⁵

^{1,2,3,4,5} Jurusan Informatika, Fakultas Ilmu Komputer,

Universitas Pembangunan Nasional "Veteran" Jawa Timur

*Email: rizkyparlika.if@upnjatim.ac.id

Abstrak

Pengukuran kualitas perangkat lunak merupakan suatu hal penting bagi perekrayaan perangkat lunak (software). Tujuan diterapkannya pengukuran pada proses perangkat lunak adalah untuk mengetahui seberapa kompleks dan bergunanya sebuah perangkat lunak tersebut. Adapun metode pengukuran yang digunakan adalah metode dari Allan Albrecht, yaitu metode Function Point Analysis, yang dikembangkan oleh International Function Point User Group (IFPUG). Dalam penelitian ini, dilakukan pengukuran kualitas pada sebuah website pemesanan tiket bus. Sebelum tahap pengukuran, disusun sebuah modul yang berisi batasan-batasan setiap poin sehingga dapat dijadikan panduan untuk memberikan skor pada masing-masing tahap pengukuran, yaitu tahap menghitung Crude Function Point (CFP) dan Relative Complexity Adjustment Factor (RCAF). Dibuat juga sebuah aplikasi kalkulator yang dapat menghitung secara otomatis total dari function point sehingga dapat mempermudah pengukuran. Setelah total nilai Function Point dari website tersebut ditemukan, total nilai tersebut dapat digunakan untuk memprediksi harga atau biaya pembuatan perangkat lunak tersebut.

Kata kunci: metode function point, modul panduan scoring, pengukuran perangkat lunak

Abstract

Measuring software quality is an important thing for software engineering. The purpose of applying measurements to software processes is to find out how complex and useful the software is. The measurement method used is Allan Albrecht's method, namely the Function Point Analysis method, which was developed by the International Function Point User Group (IFPUG). In this study, the authors measured the quality of a bus ticket booking website. Prior to the measurement stage, the authors compiled a module that contains the limitations of each point so that it can be used as a guide for scoring at each measurement stage, namely the stage of calculating the Crude Function Point (CFP) and Relative Complexity Adjustment Factor (RCAF). A calculator application is also made that can automatically calculate the total of function points so that it can make measurements easier. After the total Function Point value of the website is found, the total value can be used to predict the price or cost of making the software.

Keywords: function point method, scoring guide module, measurement software

PENDAHULUAN

Pengembangan suatu perangkat lunak yang baik, akan memperhatikan kualitas perangkat lunak secara menyeluruh. Kualitas perangkat lunak tidak hanya ditinjau dari hasil produknya saja, melainkan juga pada tahapan proses pengembangannya (Hariri & Prakasa, 2019). Untuk memastikan kualitasnya, diperlukan suatu pengukuran pada perangkat lunak.

Pengukuran adalah dasar dari disiplin dalam rekayasa perangkat lunak. Pengukuran pencapaian proyek perangkat lunak yang dipantau. Ukuran kuantitatif, yang disebut pengukur, harus disediakan untuk setiap pengukuran yang dilakukan. Berdasarkan terminologi rekayasa perangkat lunak, metrik didefinisikan sebagai pengukuran kuantitatif dari sistem, komponen, atau proses tertentu dengan atribut tertentu.

Dalam penelitian kali ini akan dipaparkan langkah-langkah pengukuran kualitas perangkat lunak yang didasarkan pada ukuran kompleksitasnya dengan menerapkan metode *function point*. Metode *function point* merupakan pengukuran untuk perangkat lunak dengan mengidentifikasi fungsionalitas perangkat lunak dan membaginya menjadi karakteristik-karakteristik yang akan lebih mudah untuk dianalisis dan dipahami. Alasan memilih metode *function point* yaitu karena beberapa kelebihan yang dimiliki oleh metode tersebut, di antaranya:

1. Metode *function point* diyakini dapat mengukur kualitas perangkat lunak dengan akurat (Parlika et al., 2020; Rajeshwari & Beena, 2018).
2. Cocok dan mendukung pengukuran proyek perangkat lunak yang dibangun dalam berbagai bahasa pemrograman.
3. Dapat mengungguli batasan dari LOC.
4. Merupakan metode yang tepat untuk mengukur kompleksitas proyek.
5. Metode yang efektif dan efisien sehingga cukup akurat untuk proses menentukan biaya *software* (Wicaksono, Khairunnisa, et al., 2021).

Maka pada penelitian ini, akan dihitung kriteria pengukuran perangkat lunak dari sebuah sistem pembelian tiket bus berbasis website menggunakan program yang telah dibuat.

TINJAUAN PUSTAKA

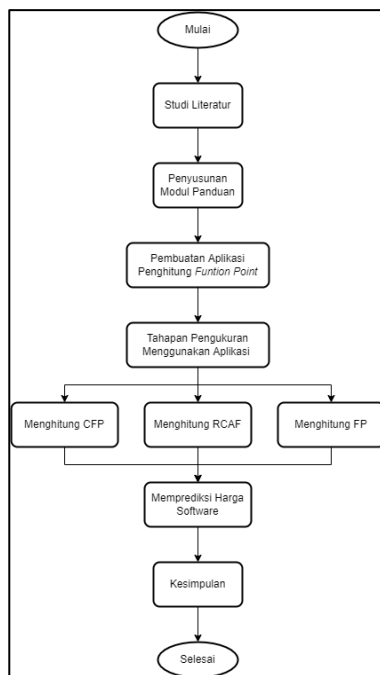
Perbedaan penelitian ini dengan penelitian-penelitian sebelumnya ialah terdapat pada metode perhitungan nilai *Function Point* yang digunakan yaitu untuk memprediksi perhitungan harga atau biaya dari suatu proyek pengembangan perangkat lunak, telah berhasil dibuat sebuah program aplikasi sejenis kalkulator berbasis website untuk dapat membantu perhitungan secara otomatis yang hasilnya akan menampilkan nilai akhir dari pengukuran *Function Point* sesuai dengan perintah yang dimasukkan oleh pihak pengukur. Penelitian ini juga dilengkapi dengan pendefinisian ulang dari masing-masing poin yang dijadikan acuan untuk menilai suatu perangkat lunak agar proses pengukuran mendapatkan hasil yang lebih konsisten sehingga dapat juga dijadikan pedoman bagi para peneliti lain yang menggunakan metode tersebut.

Pernyataan-pernyataan di atas didukung dengan sumber-sumber yang dijadikan referensi kemudian membuat ringkasannya, baik penelitian yang sudah terpublikasikan atau belum terpublikasikan. Berikut ini akan diuraikan beberapa penelitian terdahulu, beserta persamaan dan perbedaan yang masih terkait dengan tema yang dikaji. Pertama, penelitian yang dilakukan oleh (Hariri & Prakasa, 2019) yang berjudul “Pengukuran Kompleksitas Sistem E-Learning di UIN Maulana Malik Ibrahim Malang menggunakan *Metrik Function Oriented*”. Dalam penelitiannya tahapan dalam menghitung nilai *Function Point* yang digunakan masih dilakukan secara manual tanpa aplikasi yang dinilai kurang efisien. Kedua, pada penelitian yang dilakukan oleh (Dewi et al., 2018). Dalam penelitiannya menggunakan metode penelitian berupa wawancara kepada *founder* atau *developer game*, sedangkan pada penelitian ini menggunakan metode berupa kuesioner yang memiliki responden dari mahasiswa IT. Selain itu untuk hasil akhir dari penelitiannya yaitu adalah kebutuhan untuk memodifikasi pemahaman parameter permainan komputer mulai dari input, output, inquiry, file logika intern, dan extern.

METODE PENELITIAN

Dalam penyusunan penelitian ini, metode yang digunakan berupa gabungan dari studi literatur dan studi kasus yang disajikan dalam bentuk *flowchart* agar mudah dipahami. Diagram alur penyusunan penelitian ini, ditunjukkan pada Gambar 1.

Pengumpulan data dilakukan untuk menentukan studi kasus yang digunakan, menyusun modul panduan pengukuran berdasarkan sumber-sumber literatur, seperti jurnal baik tingkat nasional maupun internasional, kemudian juga melakukan tahap-tahap pengukuran menggunakan program penghitungan *Function Point* yang telah dibuat, dan yang terakhir adalah melakukan penghitungan untuk memprediksi harga suatu perangkat lunak



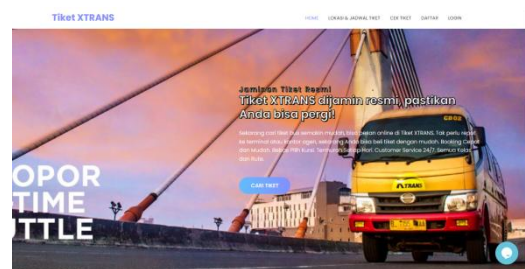
Gambar 1. Diagram Alur Penelitian

2.1 Penentuan Studi Kasus

Studi kasus yang digunakan dalam penelitian ini yaitu sebuah sistem pemesanan tiket bus berbasis website bernama “Tiket Xtrans” yang dapat diakses melalui tautan berikut ini [<http://43.247.36.85/ci-tiket/>]. Tiket Xtrans ini merupakan proyek individu yang dibuat oleh Bahyu Sanciko dan *source code*-nya dapat diakses secara publik melalui [GitHub](#).

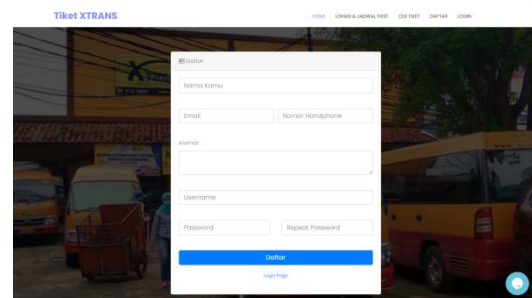
Tiket Xtrans merupakan website yang dapat digunakan untuk memesan tiket bus Xtrans. Untuk saat ini, *database* yang digunakan masih terbatas pada 6 (enam) terminal di wilayah Kota Jakarta dan Bandung. Melalui website ini, *user* dapat melakukan pemesanan tiket, termasuk aktivitas memilih tanggal keberangkatan, menentukan lokasi keberangkatan dan tujuan, memilih kursi, dan melakukan pembayaran secara *online*. Selain itu *user* akan mendapatkan ringkasan pembayaran dan juga e-tiket melalui email.

Pada tampilan website tiket Xtrans terdapat menu Home, lokasi dan jadwal, cek tiket, daftar, dan login, seperti yang ditunjukkan pada Gambar 2.



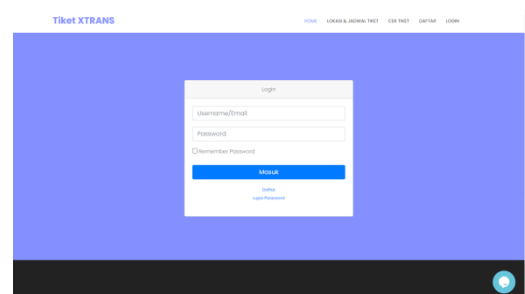
Gambar 2. Tampilan Website Tiket Xtrans

Pada form registrasi akun baru terdapat beberapa informasi untuk melengkapi data diri yang harus dibuat sebelum menggunakan website tersebut, seperti yang ditunjukkan pada Gambar 3.



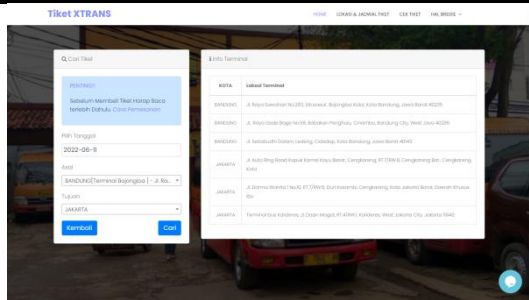
Gambar 3. Form Registrasi Akun Baru

Pada form login terdapat username dan password yang digunakan oleh user maupun admin untuk dapat login ke laman website tersebut, seperti yang ditunjukkan pada Gambar 4.



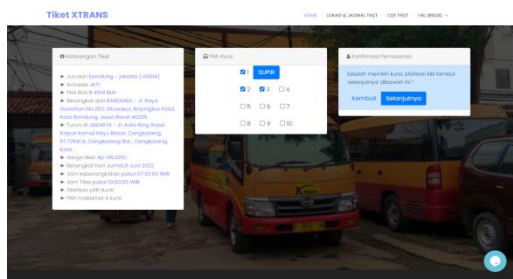
Gambar 4. Form Login

Pada halaman informasi digunakan untuk mempermudah pengunjung atau penumpang mengetahui beberapa informasi terkait jadwal operasi dan lokasi keberangkatan bus, seperti yang ditunjukkan pada Gambar 5.



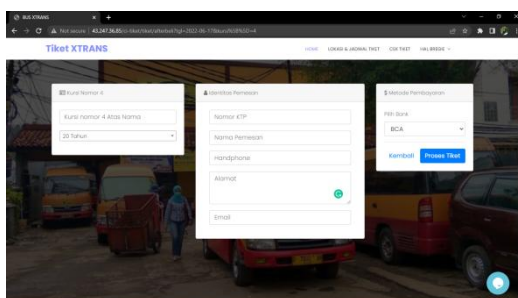
Gambar 5. Halaman Informasi Jadwal Operasi dan Lokasi Keberangkatan Bus

Pada halaman pemilihan kursi digunakan agar pengunjung atau penumpang dapat mengetahui kursi mana yang masih kosong atau tersedia, seperti yang ditunjukkan pada Gambar 6.



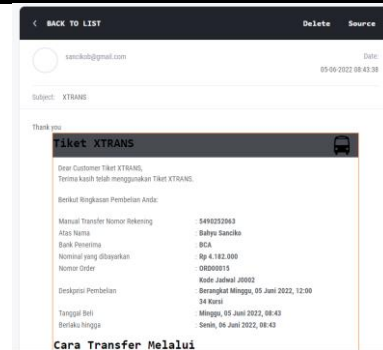
Gambar 6. Halaman Pemilihan Kursi

Pada halaman pengisian detail pemesanan terdapat beberapa keterangan mengenai pembelian tiket, seperti yang ditunjukkan pada Gambar 7.



Gambar 7. Halaman Pengisian Detail Pemesan

Pada Gambar 8 terdapat contoh konfirmasi email pembayaran yang digunakan agar pengunjung atau penumpang untuk dapat melakukan pembayaran tiket yang telah dipesan.



Gambar 8. Contoh Email Konfirmasi Pembayaran

2.2 Penyusunan Modul Panduan Pengukuran

Sebelum melakukan pengukuran, harus mengetahui bagaimana cara menentukan skor pada setiap komponen Fungsi Poin. Oleh karena itu disusunlah sebuah modul panduan untuk mengetahui batasan pada pemberian setiap skor agar proses pengukuran mendapatkan hasil yang konsisten. Modul panduan ini akan sangat bermanfaat ketika diterapkan pada suatu perusahaan karena dapat dijadikan pedoman pada setiap kali melakukan pengukuran kualitas perangkat lunaknya.

2.2.1 Tahap CFP

CFP memiliki 5 komponen pokok yang masing-masing akan diberi bobot tergantung level kompleksitasnya. Penjelasan setiap bobotnya antara lain sebagai berikut ini.

a) Tipe User Input

- (1). Mudah: tidak memerlukan terlalu banyak data untuk diinputkan.
- (2). Sedang: membutuhkan data dari user untuk diinputkan secara sederhana.
- (3). Kompleks: membutuhkan banyak data yang perlu diinputkan. Misalnya, pada sistem presensi memerlukan data nomor induk, waktu real time saat pengisian, dan keterangan lainnya.

b) Tipe Output

- (1). Mudah : hanya perlu menampilkan teks
- (2). Sedang: menghasilkan laporan, misalnya dalam bentuk pdf, csv, dan sebagainya.
- (3). Kompleks: membutuhkan banyak proses query untuk menampilkannya (banyak data yang diperlukan).

c) Tipe Online User Query/Search/View

- (1). Mudah: data yang perlu ditampilkan hanya sedikit, query yang perlu

- dituliskan tidak lebih dari 3 baris. Misalnya, menampilkan data absen harian, yaitu hanya menampilkan data yang masuk dan yang tidak.
- (2). Sedang: data yang perlu ditampilkan memiliki rentang waktu. Misalnya, data absensi dalam satu minggu.
 - (3). Kompleks: data yang ditampilkan perlu mengidentifikasi data minoritas (perkecualian). Misalnya, menampilkan data gaji pegawai yang memiliki jam kerja tidak penuh, dan sebagainya.
- d) Tipe File/Tabel/Database
- (1). Mudah: memerlukan banyak tabel
 - (2). Sedang: memerlukan banyak tabel untuk menyimpan dalam periode waktu yang lama
 - (3). Kompleks: memerlukan sangat banyak tabel yang berelasi secara bertingkat
- e) Tipe Interface Eksternal
- (1). Mudah: terhubung dengan sistem lain yang dirujuk oleh aplikasi
 - (2). Sedang: terhubung dengan sistem lain yang sudah ada, tetapi dipelihara oleh aplikasi lain.
 - (3). Kompleks: Terhubung dengan sistem lain yang sudah ada, tetapi sulit ketika mengaksesnya dan tidak boleh merusak sistem sebelumnya. Misalnya membuat aplikasi sms server yang membutuhkan koneksi pada perangkat keras modem telepon.
- 2.2.2 Tahap RCAF**
- RCAF bertujuan untuk menghitung kompleksitas suatu sistem perangkat lunak dari beberapa karakteristik subjek. Pada skala 0 sampai 5, setiap topik yang memiliki standar pengaruhnya masing-masing pada pengembangan yang diperlukan dievaluasi.
- a) Tingkat Kompleksitas Komunikasi Data
- (1). 0 = Kompleksitas komunikasi data tidak berpengaruh sama sekali terhadap proses *development software*.
 - (2). 1 = Kompleksitas komunikasi data memiliki pengaruh yang kecil terhadap jalannya proses *development software*.
 - (3). 2 = Kompleksitas komunikasi data sedikit berpengaruh terhadap jalannya proses *development software*.
 - (4). 3 = Kompleksitas komunikasi data memiliki besar pengaruh yang sedang atau setara rata-rata terhadap proses *development software*.
 - (5). 4 = Kompleksitas komunikasi data memiliki pengaruh yang cukup signifikan terhadap proses *development software*.
 - (6). 5 = Kompleksitas komunikasi data berpengaruh sangat besar dan bersifat esensial dalam proses *development software*.
- b) Tingkat Kompleksitas Pemrosesan Terdistribusi
- (1). 0 = Perangkat lunak hanya berada pada satu komputer sehingga tidak bisa digunakan secara bersama-sama.
 - (2). 1 = Perangkat lunak hanya dapat digunakan oleh perangkat atau komputer yang berada dalam satu jaringan sederhana saja, misalnya jaringan LAN dalam suatu perusahaan. Sistem ini memungkinkan untuk digunakan secara bersamaan, serta dapat mengumpulkan data pada server.
 - (3). 2 = Perangkat lunak dapat digunakan oleh publik, tetapi tidak memerlukan akses pada *database* sistem lain. Penggunaannya cukup sederhana, seperti website yang menampilkan suatu artikel tulisan tanpa menyimpan data pengguna atau tanpa adanya aktivitas pertukaran data yang kompleks.
 - (4). 3 = Perangkat lunak dapat digunakan secara luas oleh publik sehingga memiliki banyak *user* yang dapat mengakses secara bersamaan dengan aktivitas yang berbeda. Sistem yang berada level ini membutuhkan *database* untuk menyimpan data pengguna.
 - (5). 4 = Perangkat lunak yang dapat digunakan oleh publik melalui jaringan internet, memiliki aktivitas *user* yang cukup kompleks, serta memerlukan perubahan pada *database* secara *realtime*.
 - (6). 5 = Ketika suatu perangkat lunak telah melibatkan *database* dari sistem lain untuk menyimpan data dari aktivitas dalam perangkat lunak kita. Misalnya,

- website *e-commerce* yang memiliki pilihan metode pembayaran dari *e-wallet*, maka data pembayaran tersebut juga akan disimpan dalam histori transaksi pada *database e-wallet* tersebut.
- c) Tingkat Kompleksitas Performance
- (1). 0 = Sistem hanya tahu bahwa performa mendukung persyaratan kinerja.
 - (2). 1 = Tidak adanya kepastian pada perencanaan pemenuhan persyaratan teknis dari sistem yang dapat ditangani di awal proses.
 - (3). 2 = Sistem lebih mengutamakan konsentrasi pada ketepatan *delivery* atau *request* dan menunda *performance* sampai tahap pengujian.
 - (4). 3 = Kinerja dari sistem dengan pengujian beban *request* yang terperinci dapat memprediksi dan mengelola pengembangan kinerja aktual dan spesifikasi.
 - (5). 4 = Kemampuan untuk memenuhi waktu respons yang diukur dengan seberapa cepat merespons sebuah *request* dari kinerja perangkat lunak yang sedang dikembangkan.
 - (6). 5 = Aplikasi dapat menunjukkan atau menggambarkan sejauh mana tingkat waktu respons, performa, dan komponen-komponennya dalam memenuhi tujuan untuk dapat menangani jumlah beban *request* yang terus menerus bertambah secara tepat waktu dan sesuai dengan perencanaan sebelumnya.
- d) Tingkat Kompleksitas Konfigurasi
- (1). 0 = kompleksitas konfigurasi tidak berpengaruh sama sekali terhadap proses *development software*.
 - (2). 1 = tingkat kompleksitas konfigurasi memiliki pengaruh yang kecil terhadap jalannya proses *development software*.
 - (3). 2 = kompleksitas konfigurasi sedikit berpengaruh terhadap jalannya proses *development software*.
 - (4). 3 = kompleksitas konfigurasi memiliki besar pengaruh yang sedang atau setara rata-rata terhadap proses *development software*.
 - (5). 4 = tingkat kompleksitas konfigurasi pengaruh yang cukup signifikan terhadap proses *development software*.
 - (6). 5 = kompleksitas konfigurasi berpengaruh sangat besar dan bersifat esensial dalam proses *development software*.
- e) Tingkat Frekuensi Penggunaan Software
- (1). 0 = Banyak *software* yang dibuat tanpa tujuan yang jelas sehingga tidak bisa memenuhi kebutuhan manusia. Sistem yang memiliki bobot ini adalah sistem yang tidak pernah digunakan setelah selesai dibuat.
 - (2). 1 = *Software* dibutuhkan untuk mendukung suatu kegiatan, tetapi jarang digunakan.
 - (3). 2 = *Software* dibutuhkan dan digunakan oleh banyak orang, tetapi masih jarang dibutuhkan dalam kehidupan sehari-hari.
 - (4). 3 = *Software* digunakan oleh banyak orang untuk memenuhi kebutuhan mereka, misalnya *e-commerce*, *e-tiket*.
 - (5). 4 = *Software* digunakan untuk mendukung kegiatan tertentu pada waktu yang telah dijadwalkan. Contohnya adalah sistem presensi yang digunakan dalam suatu instansi.
 - (6). 5 = *Software* telah digunakan oleh banyak orang dan sering digunakan setiap waktu. Contohnya adalah *search engine*, seperti google, dan aplikasi media sosial, seperti Instagram, Twitter, Facebook, TikTok, Youtube, dan lain sebagainya.
- f) Tingkat Frekuensi *Input Data*
- (1). 0 = Frekuensi *input data* sangat sedikit sehingga tidak berpengaruh pada proses pengembangan perangkat lunak.
 - (2). 1 = Frekuensi *input data* memiliki pengaruh yang sangat kecil terhadap proses pengembangan.
 - (3). 2 = Frekuensi *input data* dapat berjumlah lebih banyak sehingga memiliki sedikit pengaruh pada proses pengembangan.
 - (4). 3 = Frekuensi *input data* pada level ini memiliki berpengaruh sedang.
 - (5). 4 = Frekuensi *input data* berpengaruh signifikan bagi berjalannya perangkat lunak.
 - (6). 5 = Frekuensi *input data* sangat berpengaruh bagi jalannya perangkat

- lunak. Pada level ini berarti perangkat lunak sangat mengandalkan masukan *user*.
- g) Tingkat Kemudahan Penggunaan Bagi *User*
- (1). 0 = Software sulit dioperasikan oleh *user* dari segala kalangan.
 - (2). 1 = Software memiliki pengaruh yang sangat kecil saat dioperasikan oleh *user* dari segala kalangan.
 - (3). 2 = Tingkat kemudahan penggunaan bagi *user* sangat sederhana terhadap pengaruh keberhasilan *software*.
 - (4). 3 = Tingkat kemudahan penggunaan bagi *user* sederhana terhadap pengaruh keberhasilan *software*.
 - (5). 4 = Tingkat kemudahan penggunaan bagi *user* memiliki pengaruh yang besar terhadap keberhasilan tujuan *software*.
 - (6). 5 = Software dapat dengan sangat mudah dipahami dan dioperasikan oleh *user* dari segala kalangan tanpa memerlukan petunjuk penggunaan yang rinci.
- h) Tingkat Frekuensi *Update Data*
- (1). 0 = Perangkat lunak tidak memiliki pengaruh dalam melakukan pembaharuan (*update*) data.
 - (2). 1 = Perangkat lunak berpengaruh sangat kecil terhadap pembaharuan (*update*) sebuah data.
 - (3). 2 = Perangkat lunak sedikit berpengaruh dalam melakukan pembaharuan (*update*) data.
 - (4). 3 = Perangkat lunak mampu melakukan pembaharuan (*update*) dengan melakukan sinkronisasi versi baru.
 - (5). 4 = Perangkat lunak interaktif ketika data dimasukkan dan seberapa sering transaksi yang dijalankan dalam memperbaharui data.
 - (6). 5 = Perangkat lunak dapat melakukan proses *update* data dengan praktis karena bisa mengunduh berkas pembaharuan dan dapat memasangnya secara langsung.
- i) Tingkat Kompleksitas *Data Processing*
- (1). 0 = Software tidak memiliki data awal untuk diproses.
 - (2). 1 = Software hanya membutuhkan data yang sedikit sehingga kompleksitas *data processing*-nya rendah.
 - (3). 2 = Software menggunakan banyak data, tetapi pengembang tidak memprosesnya dengan baik.
 - (4). 3 = Software memiliki jumlah data yang cukup banyak, tetapi masih termasuk sederhana. Namun, pengembang dapat menerjemahkannya ke dalam sistem informasi yang terlihat kompleks.
 - (5). 4 = Software berupa sistem informasi yang menghimpun data dari banyak tabel dalam *database* dan menampilkannya menjadi suatu situs yang utuh sehingga dapat memudahkan pengguna.
 - (6). 5 = Software merupakan gabungan dari sistem informasi dan *software* yang melayani proses transaksi pengguna sehingga fitur yang ditawarkan sangat beragam dan tahap *processing* datanya kompleks.
- j) Tingkat Kemungkinan Penggunaan Kembali/*Reusable* Kode Program
- (1). 0 = Software hanya dibuat khusus untuk individu dan aktivitas tertentu sehingga tidak banyak dikembangkan juga oleh *programmer* lain.
 - (2). 1 = Pengembangan *software* masih menerapkan metode pemrograman prosedural, dan program memiliki fitur-yang berbeda-beda sehingga kode program tidak dapat digunakan kembali.
 - (3). 2 = Kode program dapat digunakan kembali, tetapi kemungkinannya masih kecil.
 - (4). 3 = Setengah dari keseluruhan kode program dapat digunakan kembali oleh aplikasi lain karena memiliki fitur yang umum digunakan.
 - (5). 4 = Lebih dari setengah kode program dapat digunakan kembali karena memiliki fitur aplikasi yang banyak dikembangkan oleh developer lainnya.
 - (6). 5 = Program dibangun dengan menerapkan *framework* sebuah bahasa pemrograman sehingga kemungkinan besar kode program dapat digunakan kembali karena telah

- menerapkan metode pemrograman berbasis objek.
- k) Tingkat Kemudahan dalam Instalasi
- (1) 0 = Software tidak memerlukan penginstalan.
 - (2) 1 = Software menunjukkan bahwa sistem tidak dapat *di-install* dengan mudah.
 - (3) 2 = Software menunjukkan intensitas penggunaan dan interaksi antara pengguna (*user*) dalam kemudahan instalasi.
 - (5) 3 = Software dapat menggambarkan tingkat konversi dan instalasi yang mengacu pada bagaimana interaksi yang jelas dan dapat dimengerti.
 - (6) 4 = Software memberikan indikasi bahwa sistem yang dirancang bukan untuk menyulitkan pemakaiannya, akan tetapi penggunaan sistem justru mempermudah dalam mengoperasikan sistem sesuai yang diinginkan pengguna (*user*).
 - (7) 5 = Software yang dibuat menunjukkan tingkat fungsionalitas seberapa mudah dapat *di-install* di berbagai macam perangkat yang berbeda.
- l) Tingkat Kemudahan Operasional *Software* (*Backup, Recovery*, dan sebagainya)
- (1) 0 = Software tidak mampu melakukan operasional *software* seperti *backup, recovery*, dan sebagainya.
 - (2) 1 = Software mampu melakukan operasional secara sederhana untuk mengamankan data dan menyalin data.
 - (3) 2 = Software sedikit mampu melakukan pengamanan data dan menyalin data.
 - (4) 3 = Software berpengaruh sedang dalam pengamanan data dan menyalin data.
 - (5) 4 = Software mampu mengamankan dan melakukan *restore* data apabila sewaktu-waktu data rusak, hilang atau terkena virus.
 - (6) 5 = Software dapat menyalin data secara otomatis dan dapat dikembalikan ketika ada kesalahan, serta dapat mem-backup data ke dalam penyimpanan eksternal, seperti cloud.
- m) Tingkat *Software* Dibuat untuk Multi Organisasi/Perusahaan/*Client*
- (1) 0 = *Software* yang dikembangkan dengan target pengguna adalah individu, yang berarti *software* tersebut dibuat hanya untuk digunakan pribadi.
 - (2) 1 = *Software* digunakan oleh beberapa orang dalam satu lingkup organisasi.
 - (3) 2 = *Software* dapat digunakan untuk berbagai organisasi dalam satu instansi. Misal, website sistem informasi suatu program studi akan memiliki *template* atau fitur-fitur yang sama dengan program studi lain dalam satu fakultas.
 - (4) 3 = *Software* dapat digunakan oleh cabang-cabang perusahaan.
 - (5) 4 = *Software* dapat digunakan oleh banyak perusahaan sekaligus, tetapi masih dalam satu lingkup, perusahaan induk dan perusahaan anak.
 - (6) 5 = *Software* dikembangkan untuk digunakan banyak perusahaan. Misalnya, proyek aplikasi sistem informasi sekolah yang dibangun untuk disewakan sehingga dapat digunakan oleh banyak sekolah sekaligus. Artinya fitur-fitur yang ditawarkan adalah fitur yang dapat digunakan secara umum.
- n) Tingkat Kompleksitas dalam Mengikuti Perubahan/Fleksibel
- (1) 0 = *Software* memiliki tingkat kompleksitas yang tidak berpengaruh terhadap jalannya program.
 - (2) 1 = Perangkat lunak sulit menerima perubahan dalam pengembangannya.
 - (3) 2 = Perangkat lunak dapat menerima perubahan sistem, tetapi sulit untuk menerapkan perubahannya.
 - (4) 3 = Kemampuan *software* dalam melacak semua perancangan dan konstruksi komponen yang mengalami perubahan.
 - (5) 4 = *Software* dapat menggambarkan sejauh mana tingkat pengembangan dalam kemudahan modifikasi pemrosesan logika atau struktur data, sehingga terjadi perubahan dan dinamika pada sebuah sistem.
 - (6) 5 = *Software* mengimplementasikan proses terstruktur dalam pengembangannya dan telah memiliki

panduan untuk menerapkan suatu perubahan pada sistem.

sama sekali hingga 5 (lima), yang menunjukkan adanya pengaruh yang besar terhadap sistem.

2.3 Tahap Pengukuran dengan Metode Function Point

2.3.1 Menghitung CFP

Pada *Crude Function Point* (CFP) akan dihitung komponen-komponen yang berkaitan dengan data dari perangkat lunak yang diukur. CFP Function Point terdiri dari 5 Komponen yaitu:

- Tipe Input, yaitu berkaitan dengan tampilan antarmuka pada bagian form-form data yang dimasukkan oleh pengguna dalam aplikasi tersebut.
- Tipe Output, yaitu berkaitan dengan keluaran aplikasi tersebut. Dapat berupa tampilan pada layar, maupun berkas yang dapat disimpan dalam perangkat dan dicetak.
- Tipe File/Tabel/Database, yaitu berkaitan dengan basis data relasional untuk menyimpan data.
- Tipe Query/Search/View, yaitu berkaitan dengan query atau baris kode untuk mengolah data pada aplikasi.
- Tipe Interface Eksternal, yaitu sistem lain yang terkait dan dibutuhkan oleh aplikasi kita.

Tabel 1. Tabel pedoman Perhitungan CFP

Tipe Komponen	Level Kompleksitas									Total CFP
	Mudah			Sedang			Kompleks			
	J	B	P	J	B	P	J	B	P	
User Input	3			4			6			?
User Output	4			5			7			?
Query/ Search/View	3			4						?
File/Tabel/ Database	7			10			15			?
External Interface	5			7			10			?

Keterangan:

- J = jumlah komponen dalam sistem
- B = bobot setiap level
- P = jumlah poin komponen setelah dikali dengan bobot

2.3.2 Menghitung RCAF

RCAF atau Relative Complexity Adjustment Factor akan mengukur *software* berdasarkan pada kompleksitas sistem yang dibedakan menjadi 14 karakteristik dan masing-masing memiliki 6 (enam) skala dari 0 (nol), yang menandakan tidak memiliki pengaruh

Tabel 2. Karakteristik Software

No	Affecting Subject	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 Software	0 1 2 3 4 5
6	Tingkat Frekuensi Input Data	0 1 2 3 4 5
7	Tingkat Kemudahan Penggunaan Bagi User	0 1 2 3 4 5
8	Tingkat Frekuensi Update Data	0 1 2 3 4 5
9	Tingkat Kompleksitas Prosesing Data	0 1 2 3 4 5
10	Tingkat Kemungkinan Penggunaan Kembali / Reusable Kode Program	0 1 2 3 4 5
11	Tingkat Kemudahan dalam Instalasi	0 1 2 3 4 5
12	Tingkat Kemudahan Operasional Software (backup, recovery)	0 1 2 3 4 5
13	Tingkat Software dibuat untuk multi organisasi/perusahaan/client	0 1 2 3 4 5
14	Tingkat kompleksitas dalam mengikuti perubahan/fleksibilitas	0 1 2 3 4 5
TOTAL		?

2.3.3 Menghitung Function Point (FP)

Setelah mendapatkan nilai total pada penghitungan *CFP* dan *RCAF*, tahap selanjutnya yaitu menghitung total nilai Function Point yang dapat dihitung dengan rumus:

$$FP = CFP \times (0.65 + (0.01 \times RCAF))$$

Angka 0.65 dan 0.01 merupakan ketentuan yang telah diatur oleh Function Point Internasional User Group (IFPUG). Setelah menerima nilai FP, maka bisa digunakan menjadi acuan untuk mengestimasi kualitas dan aplikasi melalui perbandingan nilai FP menggunakan *poly error* (kesalahan).

2.4 Penghitungan Prediksi Harga Software

Dalam memprediksi harga perangkat lunak yang telah diukur dengan metode *Function Point*, menerapkan teknik pada penelitian terdahulu yang dilakukan oleh (Widyastuti & Ariandi, 2021). Pertama, kita harus mengetahui harga per Function Point yang dapat dihitung menggunakan rumus berikut.

$$\text{Harga} = \frac{\text{Gaji Programmer}}{\text{Produktivitas}}$$

Yang dimaksud produktivitas di atas adalah banyaknya Function Point dapat diselesaikan oleh satu programmer per bulan. Selanjutnya, dapat dihitung total harga perangkat lunak dengan rumus:

$$\text{Total Biaya} = \text{FP} \times \text{Harga per FP}$$

HASIL DAN PEMBAHASAN

Pada bagian ini, diuraikan hasil pengukuran menggunakan aplikasi yang telah dibuat pada website yang dijadikan studi kasus, yaitu pada sistem pemesanan tiket bus berbasis website. Setelah meneliti website berdasarkan komponen Function Point pada modul pedoman yang telah disusun sebelumnya, didapatkan pengukuran seperti yang ditunjukkan pada Gambar 9.

No.	Tipe Komponen	Mudah			Level Kompleksitas			Kompleks			Total CFP
		JML	BOBOT	POINT	JML	BOBOT	POINT	JML	BOBOT	POINT	
1	Type Input	3	3	9	4	4	16	5	5	25	34
2	Type Output	2	4	8	3	9	18	2	7	14	27
3	Type Query/Insert/View	2	3	6	3	4	16	2	6	12	32
4	Type File/Table/Database	2	7	14	0	0	0	1	16	16	32
5	Type Interface External	0	0	0	0	0	0	0	0	0	30
TOTAL											135

Gambar 9 . Tampilan Aplikasi Perhitungan Crude Function Point

Setelah melakukan pengukuran dan memasukkan setiap nilainya pada aplikasi, didapatkan nilai total CFP sebesar 135.

Tahap selanjutnya adalah menghitung nilai RCAF. Berdasarkan perhitungan RCAF pada 14 karakteristik seperti yang ditunjukkan pada Gambar 10, nilai total RCAF website pemesanan tiket bus tersebut adalah sebesar 45.

No.	Karakteristik	Bobot
1	Tingkat kompleksitas Komunikasi Data	4
2	Tingkat Kompleksitas Pemrosesan Terdistribusi	3
3	Tingkat Kompleksitas Performance	2
4	Tingkat Kompleksitas Konfigurasi	2
5	Tingkat Frekuensi Penggunaan Software	3
6	Tingkat Frekuensi Input Data	5
7	Tingkat Kemudahan Penggunaan Bagi User	5
8	Tingkat Frekuensi Update data	4
9	Tingkat kompleksitas Prosesing Data	4
10	Tingkat Kemungkinan Penggunaan Kembali/Reusable Kode Program	4
11	Tingkat Kemudahan Dalam Instalasi	2
12	Tingkat Kemudahan Operasional Software (Backup, Recovery, dsb)	3
13	Tingkat Software Dibuat Untuk Multi Organisasi/Perusahaan/Client	1
14	Tingkat Kompleksitas Dalam Mengikuti Perubahan/Fleksibel	3
TOTAL		45

Gambar 10. Tampilan Aplikasi Hitung RCAF

Setelah pengukuran pada CFP dan RCAF selesai dikalkulasikan, selanjutnya adalah menghitung total nilai function point dengan memasukkannya pada rumus yang telah ditentukan, dan didapatkan nilai FP sebesar 148,50, seperti yang ditunjukkan pada Gambar 11.

Function Point (FP)
Rumus menghitung Function Point = CFP x (0.6 + 0.01 x RCAF)
Hasil Perhitungan Function Point Anda Adalah :
148.50

Gambar 11. Total Function Point

Selanjutnya, untuk memprediksi harga perangkat lunak dalam metode Function Point, dibutuhkan informasi gaji seorang *Software Developer* di Indonesia. Maka dari itu, digunakan panduan standar gaji berbagai jenis pekerjaan di Indonesia, yaitu Kelly Services agar prediksi harga website ini lebih akurat. Jika dilihat dari sistem website pemesanan tiket bus ini, pengerjaannya dapat diselesaikan oleh seorang *Junior Software Developer*. Berdasarkan panduan tersebut, rentang gaji untuk junior software developer yang memiliki pengalaman selama 1-3 tahun berkisar antara 8-12 juta per bulan dan produktivitas programmer dapat menyelesaikan 30 Function Point dalam

sebulan. Dengan demikian didapatkan perhitungan:

$$\text{Harga} = \frac{\text{Gaji Programmer}}{\text{Produktivitas}}$$

$$\text{Harga} = \frac{8.000.000}{30}$$

$$\text{Harga} = 266.666 \approx 267.000$$

Jadi, harga per Function Point adalah Rp267.000.

Sesuai perhitungan jumlah Function Point pada tahap sebelumnya, website pemesanan tiket bus ini memiliki 148,50 Function Point. Maka, harga website adalah total $FP \times \text{harga per FP}$, yaitu $148,5 \times 267.000 = 39.649.500$. Dengan demikian, harga prediksi dari sistem pemesanan tiket bus berbasis web “Tiket Xtrans” tersebut adalah Rp39.649.500.

SIMPULAN

Berdasarkan hasil penelitian yang dilakukan, diperoleh kesimpulan bahwa penyusunan modul panduan telah berhasil dilakukan untuk mengukur kualitas perangkat lunak berdasarkan metode Function Point. Sebuah program perhitungan telah berhasil dibuat untuk dapat menampilkan nilai akhir pengukuran Function Point sesuai yang dimasukkan oleh pihak pengukur. Hasil pengujian yang dilakukan terhadap sistem pemesanan tiket bus dengan menggunakan metode function point menunjukkan tingkat kompleksitasnya sebesar 148,50. Harga proyek perangkat lunak telah berhasil diprediksi dan didapatkan harga website pemesanan tiket bus tersebut sebesar Rp39.649.500.

DAFTAR PUSTAKA

- Aljohani, M., & Qureshi, R. (2017). Comparative Study of Software Estimation Techniques. *International Journal of Software Engineering & Applications*, 8(6), 39–53.
- Dewi, R. S., Andari, T. W., A. Rasyid, M. B., & Candra A.P., R. (2018). Ekstraksi Faktor Kompleksitas Game Menggunakan Metode Function Points. *Jurnal Teknologi Dan Sistem Informasi*, 4(3), 115–122.
- Eti Kapita, R. B. H., Angelin, M., & Wicaksono, S. R. (2019). Analisis Kualitas Perangkat Lunak Menggunakan Metode Function Point Analysis (Study Kasus: Transaksi Pembelian di eBay). *Jurnal Informatika Dan Rekayasa Perangkat Lunak*, 1(1).
- Hapsari, R. K., & Husen, M. J. (2015). Estimasi Kualitas Perangkat Lunak Berdasarkan Pengukuran Kompleksitas Menggunakan Metrik Function Oriented. *Seminar Nasional Sains Dan Teknologi Terapan III*, 425–434.
- Hariri, F. R., & Prakasa, J. E. W. (2019). Pengukuran Kompleksitas Sistem E-Learning di UIN Maulana Malik Ibrahim Malang Menggunakan Metrik Function Oriented. *Seminar Nasional Inovasi Teknologi UN PGRI Kediri*, 245–250.
- Hendrawan, H., Jusia, P. A., Rasywir, E., & Pratama, Y. (2019). Eksperimen Estimasi Biaya Proyek Perangkat Lunak Menggunakan Metode Function Point Analysis (FPA). *Riset Komputer (JURIKOM)*, 6(4), 340–347.
- Khairani, D. (2015). Studi Kasus Pengukuran Sistem Informasi Menggunakan Function Point (Fp). *Jurnal Teknik Informatika*, 8(2).
- Parlika, R., Wijaya, D. C. M., Khariono, H., & Fernanda, R. A. (2020). Studi literatur perbandingan antara metode LOC, COCOMO, FPA dalam ranah software metric. *Jurnal Pendidikan Informatika Dan Sains*, 9(1), 66–74.
- PERSOLKELLY. (2020). *Indonesia Salary Guide 2020/21*. Diakses pada Juni 2022 <https://www.kellyservices.co.id/media/kellyservicesindo/client/PDFFiles/PERSOLKELLY-INDONESIA-SALARY-GUIDE-2020-2021.pdf>
- Rachmat, N., & Saparudin. (2017). Estimasi Ukuran Perangkat Lunak Menggunakan Function Point Analysis-Studi Kasus Aplikasi Pengujian dan Pembelajaran Berbasis Web. *Annual Research Seminar (ARS)*, 3(1), 57-60.
- Rajeshwari, K., & Beena, R. (2018). A Critique on Software Cost Estimation. *International Journal of Pure and Applied Mathematics*, 118(20), 3851–3862.
- Ritzkal, Goeritno, A., & Hadi P, E. (2017). Pengukuran Kualitas Perangkat Lunak Sistem E-Learning Menggunakan Metric Function Oriented. *Prosiding SNATIF*, 4, 769-776.
- Rizka, C. L. D., Dewi, F. S., & Wicaksono, S. R. (2019). Pengukuran Dan Kualitas

- Perangkat Lunak Website “Linkedin” Menggunakan Metode Function Point Analysis. *JISKA (Jurnal Informatika Sunan Kalijaga)*, 3(2), 79.
- Sanciko, B. (2022). Tiket Bus. <https://github.com/bahyusanciko/ci-tiket>. (diakses pada Mei 2022).
- Sariyanti, S., & Ardiansyah. (2017). Kakas Estimasi Perangkat Lunak Menggunakan Function Point dan Use Case Point untuk Praktikum Rekayasa Perangkat Lunak di Universitas Ahmad Dahlan. *Prosiding Annual Research Seminar 2017*, 3(1), 85-92.
- Septi, E. T. C., Sartika, F. D., & Dewi, R. S. (2020). Pengukuran Sistem Informasi Menggunakan Metode Function Point (FP) pada Siakad Universitas XYZ. *JURIKOM (Jurnal Riset Komputer)*, 7(1), 122.
- Wicaksono, S. R., Valentina, I., Ekadana, F. A., & Chandra, M. N. (2021). Pengukuran Kualitas Perangkat Lunak Menggunakan Function Point Analysis (Studi Kasus: Fishbowl). *Decode: Jurnal Pendidikan Teknologi Informasi*, 1(2), 43–49.
- Widyastuti, S., & Ariandi, W. (2021). Sistem Informasi Menghitung Biaya Pembangunan Perangkat Lunak dengan Menggunakan Metode Matrik Function Point. *Syntax Literate: Jurnal Ilmiah Indonesia*, 6(10), 5339–5355.
- Wulandari, Y., Fatkha, B., & Rizky, S. (2021). Pengukuran Perangkat Lunak dengan Menggunakan Metode Function Point (Studi Kasus : Cashlez). *Jurnal Akuntansi Manajemen Bisnis Dan Teknologi*, 1(2), 271–284.