Nama : Farrel Fairuz Iskandar

NIM : 2300018157

Daftar Pembagian Tugas

No	Pekan Ke-	List Tugas	PIC
1	1	Skenario Use Case	Farrel Fairuz Iskandar
2	1	Analisis Pengerjaan Proyek	Farrel Fairuz Iskandar
3	12	Software Costing (Use Case Points)	Farrel Fairuz Iskandar
4	14	Desain Antarmuka	Farrel Fairuz Iskandar

Deskripsi

Aplikasi ini dirancang untuk membantu para pemilik usaha penyewaan alat camping dalam mengelola penyewaan secara digital dan efisien. Dalam sistem ini, pelanggan dapat melihat ketersediaan alat, melakukan pemesanan secara online, serta mendapatkan informasi terkait jadwal penyewaan dan pengembalian barang. Di sisi lain, admin atau pengelola dapat dengan mudah mencatat data penyewaan, memantau status alat (tersedia, sedang disewa, rusak), serta membuat laporan transaksi penyewaan.

Selain itu, aplikasi ini dilengkapi dengan sistem pembayaran digital dan notifikasi otomatis untuk mengingatkan pelanggan tentang jadwal pengembalian alat. Dengan desain yang sederhana dan mudah digunakan, aplikasi ini membantu mempercepat proses penyewaan, meningkatkan efisiensi bisnis, dan memberikan pengalaman yang lebih nyaman bagi pelanggan maupun pemilik usaha.

Ruang Lingkup Aplikasi

Aplikasi Camping-Keun dirancang khusus untuk mendukung proses penyewaan alat camping secara digital. Berikut adalah ruang lingkup utama berdasarkan laporan dan kebutuhan pengguna**.**

User (Penyewa Alat Camping)

- Individu atau kelompok yang ingin menyewa alat camping untuk keperluan pribadi, komunitas, atau wisata alam.
- Mahasiswa, pendaki, pecinta alam, traveler, atau umum.
- Kebutuhan mereka: cepat, praktis, jelas dalam melihat ketersediaan dan harga alat.

Admin / Pengelola Usaha

- Pemilik atau pengelola jasa penyewaan alat camping.
- Bertugas mengelola data alat, menyetujui pemesanan, memverifikasi pengembalian, dan memantau transaksi.

Worker / Petugas

- Pihak yang menangani pengecekan fisik alat, penyerahan barang, dan pemeriksaan pasca sewa.

2. Ruang Lingkup Fungsi Sistem

Untuk User:

- Membuat akun dan login
- Mencari dan memilih alat camping
- Melihat ketersediaan & detail produk
- Mengisi form peminjaman
- Melakukan pembayaran (QRIS, VA, dll)
- Menerima notifikasi jadwal pengembalian
- Melihat riwayat penyewaan
- Mengelola profil pengguna

Untuk Admin/Worker:

- Melihat dan memverifikasi data user
- Menambahkan/mengedit/menghapus data alat
- Mencatat status alat (tersedia, disewa, rusak)
- Melihat daftar peminjaman masuk
- Memverifikasi transaksi & pengembalian
- Membuat laporan transaksi

1. Use Case: Register

Aktor: User

Deskripsi: Pengguna melakukan pendaftaran untuk mendapatkan akses ke sistem.

Alur:

- Pengguna mengakses halaman registrasi.
- Mengisi data yang dibutuhkan (nama, email, password, dll).
- Sistem menyimpan data pengguna.
- User berhasil terdaftar.

2. Use Case: Login

Aktor: User

Deskripsi: Pengguna masuk ke sistem menggunakan kredensial yang valid.

Alur:

- Pengguna memasukkan username dan password.
- Sistem memverifikasi kredensial.
- Jika berhasil, user diarahkan ke halaman utama aplikasi.

3. Use Case: Pemilihan Alat Camp

Aktor: User

Deskripsi: Pengguna memilih alat camping yang ingin disewa.

Alur:

- Pengguna melihat daftar alat camping.
- Memilih satu atau lebih alat untuk disewa.
- Sistem mencatat pilihan alat tersebut.

4. Use Case: Pengisian Data untuk Peminjaman

Aktor: User

Deskripsi: Pengguna mengisi data peminjaman setelah memilih alat.

Alur:

- Sistem menampilkan form peminjaman.
- Pengguna mengisi detail tanggal pinjam dan kembali.
- Data disimpan oleh sistem.

5. Use Case: Verifikasi Data Peminjam

Aktor: Admin

Deskripsi: Admin memverifikasi data peminjam sebelum transaksi disetujui.

Alur:

- Admin melihat data peminjaman masuk.

- Admin memeriksa kelengkapan dan keabsahan data.
- Admin menyetujui atau menolak peminjaman.

6. Use Case: Transaksi Peminjaman Barang

Aktor: User

Deskripsi: Proses transaksi peminjaman dilakukan setelah data diverifikasi.

Alur:

- Sistem memproses transaksi setelah verifikasi.
- Mencatat transaksi peminjaman dan memberi nomor referensi.

7. Use Case: Verifikasi Barang Peminjaman

Aktor: Admin, Worker

Deskripsi: Petugas memverifikasi barang sebelum diserahkan ke peminjam.

Alur:

- Worker/Admin mengecek kondisi dan jumlah barang.
- Barang disiapkan untuk diambil oleh peminjam.

8. Use Case: Pengambilan Barang yang Telah Dipinjam

Aktor: User, Worker

Deskripsi: User mengambil barang yang sudah diverifikasi.

Alur:

- User datang ke tempat pengambilan.
- Worker menyerahkan barang sesuai data transaksi.

9. Use Case: Pengembalian Barang yang Dipinjam

Aktor: User, Admin

Deskripsi: Pengguna mengembalikan barang yang telah dipinjam.

Alur:

- User mengembalikan barang ke lokasi pengembalian.
- Admin mengecek barang (kondisi, jumlah) dan Sistem memperbarui status menjadi 'telah dikembalikan'.

Analisis Pengerjaan Proyek Aplikasi Camping-Keun

1. Tinjauan dari Sisi Waktu

Proyek ini dikerjakan dalam waktu yang cukup singkat, dengan pembagian tugas yang merata antar anggota. Setiap anggota menyelesaikan jobdesc-nya tepat waktu pada minggu pertama. Meski efisien, pengerjaan secara cepat ini juga mengandung risiko keterbatasan eksplorasi dan pendalaman fitur-fitur sistem. Tahapan selanjutnya seperti desain antarmuka, coding, dan testing masih belum dikerjakan saat analisis ini dibuat.

2. Keterpenuhan Spesifikasi

Spesifikasi sistem yang dirancang mencakup semua fitur utama seperti registrasi, login, peminjaman, dan pengembalian alat camping. Namun, karena proyek masih dalam tahap perancangan, keterpenuhan spesifikasi baru sebatas dokumen, belum ada realisasi bentuk aplikasi. Ini berarti spesifikasi fungsional sudah terpenuhi secara teoritis, namun perlu dilanjutkan ke fase implementasi untuk melihat kesesuaiannya secara nyata.

3. Kendala

Kendala utama berasal dari keterbatasan waktu dan belum meratanya kemampuan teknis antar anggota. Karena proyek masih berada di tahap desain awal, belum banyak tantangan teknis yang ditemukan. Namun, kendala potensial akan muncul saat mulai implementasi fitur seperti otentikasi, pengelolaan stok alat, dan integrasi notifikasi.

4. Tantangan Masa Depan

Jika proyek ini dikembangkan lebih lanjut, tantangan utama meliputi:

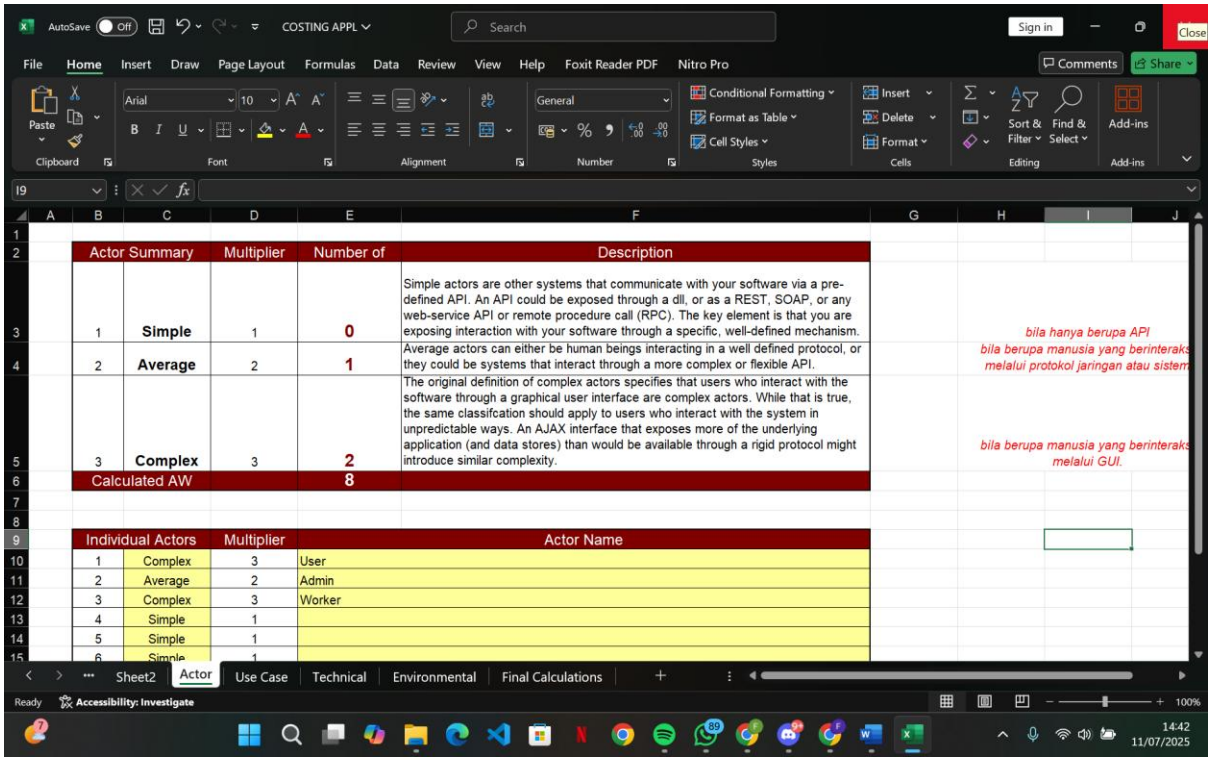
- Membuat sistem real-time untuk status ketersediaan alat
- Integrasi metode pembayaran digital (misal: QRIS, transfer bank)
- Keamanan sistem: perlindungan data pengguna, enkripsi
- Manajemen akun admin vs user dan kontrol peminjaman
- Pemeliharaan alat: pelacakan alat yang rusak, hilang, atau belum dikembalikan

6. Kesimpulan

Proyek "Camping-Keun" sudah memiliki fondasi awal yang cukup baik dalam bentuk dokumen analisis. Pembagian tugas berjalan lancar dan efisien. Tantangan ke depan terletak pada proses implementasi teknis dan pengujian sistem. Tim perlu melanjutkan proyek ini ke tahap pengembangan dan pengujian agar sistem benar-benar bisa digunakan.

Software Costing (Use Case Points)

Actor Weighting



Actor Summary		Multiplier	Number of	Description
1	Simple	1	0	Simple actors are other systems that communicate with your software via a pre-defined API. An API could be exposed through a dll, or as a REST, SOAP, or any web-service API or remote procedure call (RPC). The key element is that you are exposing interaction with your software through a specific, well-defined mechanism.
2	Average	2	1	Average actors can either be human beings interacting in a well defined protocol, or they could be systems that interact through a more complex or flexible API.
3	Complex	3	2	The original definition of complex actors specifies that users who interact with the software through a graphical user interface are complex actors. While that is true, the same classifcation should apply to users who interact with the system in unpredictable ways. An AJAX interface that exposes more of the underlying application (and data stores) than would be available through a rigid protocol might introduce similar complexity.
Calculated AW			8	

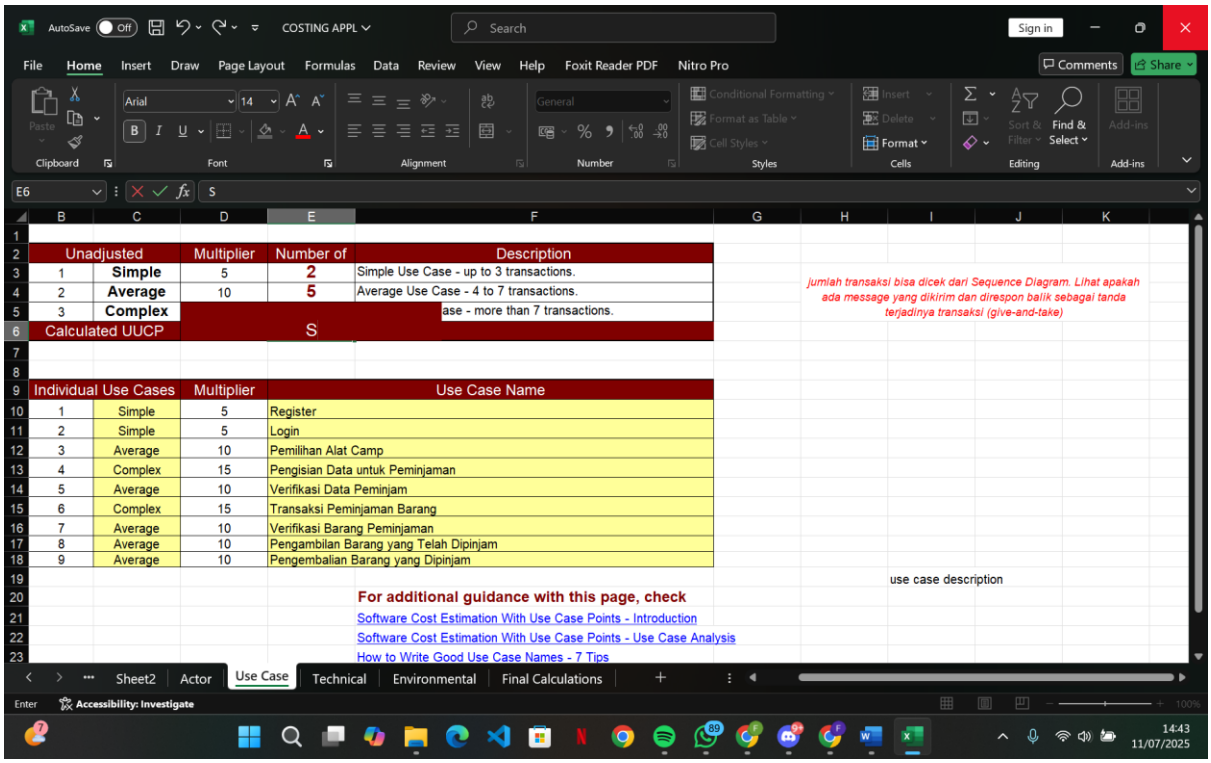
bila hanya berupa API
bila berupa manusia yang berinterak
melalui protokol jaringan atau siste

bila berupa manusia yang berinterak
melalui GUI.

Individual Actors		Multiplier	Actor Name
1	Complex	3	User
2	Average	2	Admin
3	Complex	3	Worker
4	Simple	1	
5	Simple	1	
6	Simple	1	

Sheet2 | Actor | Use Case | Technical | Environmental | Final Calculations

UUCP



Unadjusted		Multiplier	Number of	Description
1	Simple	5	2	Simple Use Case - up to 3 transactions.
2	Average	10	5	Average Use Case - 4 to 7 transactions.
3	Complex			ase - more than 7 transactions.
Calculated UUCP			S	

jumlah transaksi bisa dicek dari Sequence Diagram. Lihat apakah ada message yang dikirim dan direspon balik sebagai tanda terjadinya transaksi (give-and-take)

Individual Use Cases		Multiplier	Use Case Name
1	Simple	5	Register
2	Simple	5	Login
3	Average	10	Pemilihan Alat Camp
4	Complex	15	Pengisian Data untuk Peminjaman
5	Average	10	Verifikasi Data Peminjam
6	Complex	15	Transaksi Peminjaman Barang
7	Average	10	Verifikasi Barang Peminjaman
8	Average	10	Pengambilan Barang yang Telah Dipinjam
9	Average	10	Pengembalian Barang yang Dipinjam

use case description

For additional guidance with this page, check

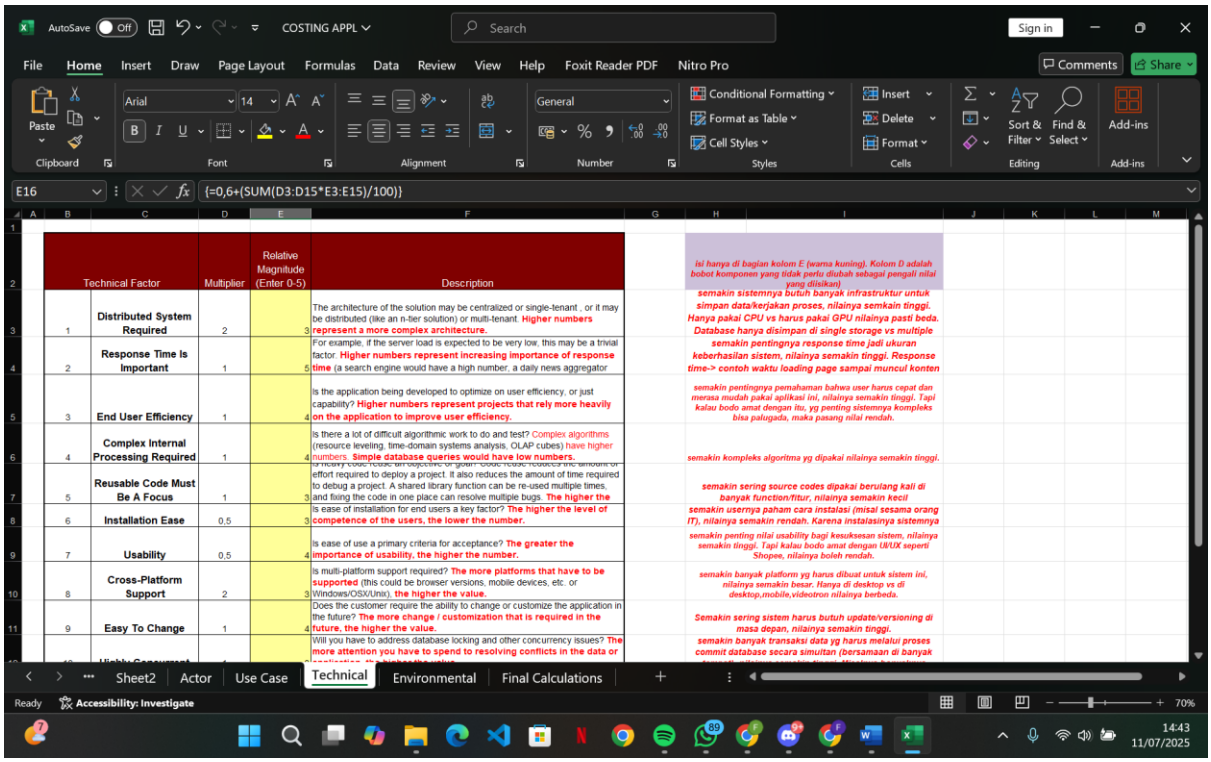
Software Cost Estimation With Use Case Points - Introduction

Software Cost Estimation With Use Case Points - Use Case Analysis

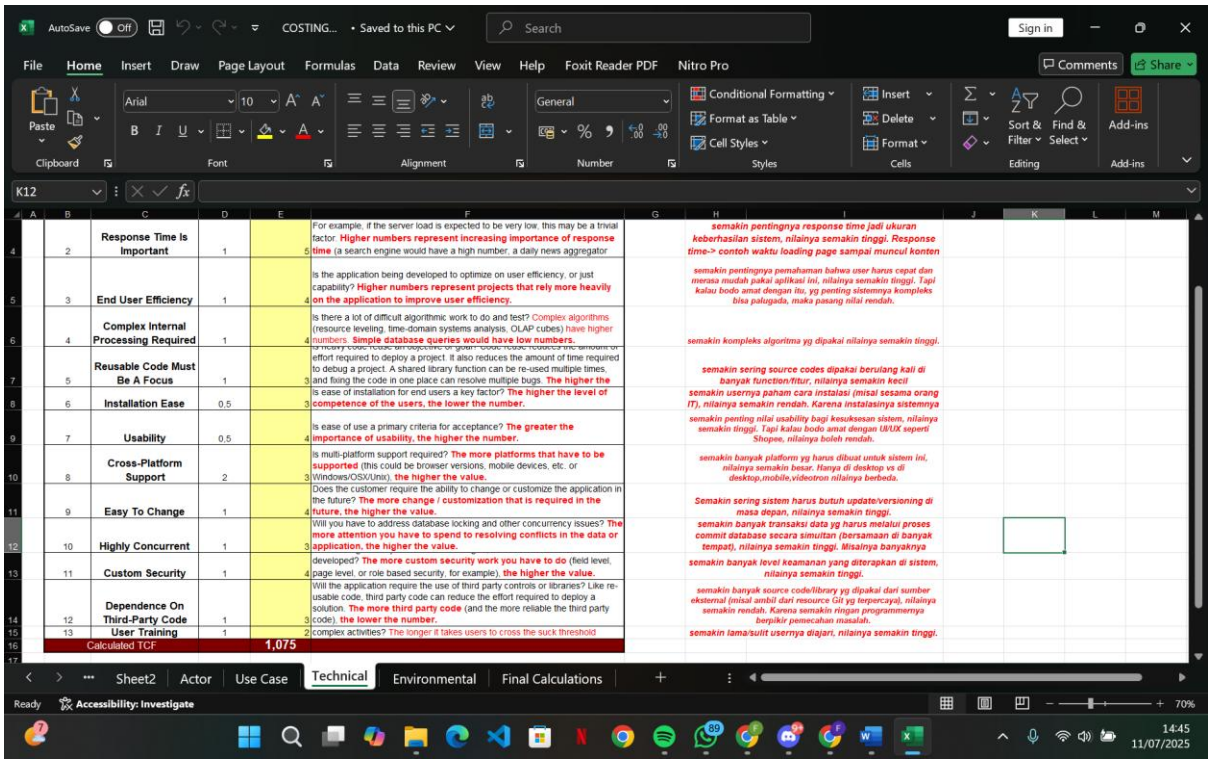
How to Write Good Use Case Names - 7 Tips

Sheet2 | Actor | Use Case | Technical | Environmental | Final Calculations

Technical Complexity Factor

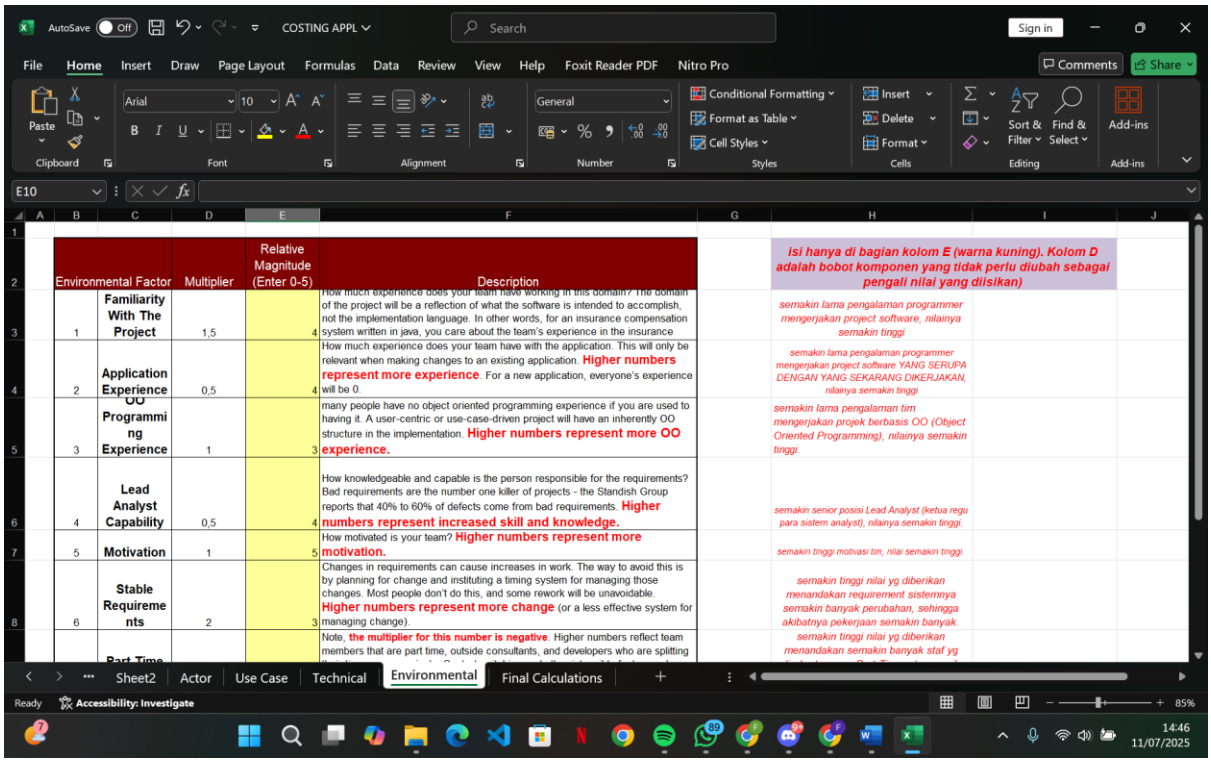


		Technical Factor	Multiplier	Relative Magnitude (Enter 0-5)	Description	
						Isi hanya di bagian kolom E (warna kuning). Kolom D adalah bobot komponen yang tidak perlu diubah sebagai pengali nilai yang diisikan
3	1	**Distributed System Required**	2	3	The architecture of the solution may be centralized or single-tenant , or it may be distributed (like an n-tier solution) or multi-tenant. **Higher numbers represent a more complex architecture.**	*semakin sistemnya butuh banyak infrastruktur untuk simpan data/kerjakan proses, nilainya semkain tinggi. Hanya pakai CPU vs harus pakai GPU nilainya pasti beda. Database hanya disimpan di single storage vs multiple*
4	2	**Response Time Is Important**	1	5	For example, if the server load is expected to be very low, this may be a trivial factor. **Higher numbers represent increasing importance of response time** (a search engine would have a high number, a daily news aggregator	*semakin pentingnya response time jadi ukuran keberhasilan sistem, nilainya semakin tinggi. Response time-> contoh waktu loading page sampai muncul konten*
5	3	**End User Efficiency**	1	4	Is the application being developed to optimize on user efficiency, or just capability? **Higher numbers represent projects that rely more heavily on the application to improve user efficiency.**	*semakin pentingnya pemahaman bahwa user harus cepat dan merasa mudah pakai aplikasi ini, nilainya semakin tinggi. Tapi kalau bodo amat dengan itu, yg penting sistemnya kompleks bisa palugada, maka pasang nilai rendah.*
6	4	**Complex Internal Processing Required**	1	4	Is there a lot of difficult algorithmic work to do and test? Complex algorithms (resource leveling, time-domain systems analysis, OLAP cubes) have higher numbers. **Simple database queries would have low numbers.**	*semakin kompleks algoritma yg dipakai nilainya semakin tinggi.*
7	5	**Reusable Code Must Be A Focus**	1	3	effort required to deploy a project. It also reduces the amount of time required to debug a project. A shared library function can be re-used multiple times, and fixing the code in one place can resolve multiple bugs. **The higher the**	*semakin sering source codes dipakai berulang kali di banyak function/fitur, nilainya semakin kecil*
8	6	**Installation Ease**	0,5	3	Is ease of installation for end users a key factor? **The higher the level of competence of the users, the lower the number.**	*semakin usernya paham cara instalasi (misal sesama orang IT), nilainya semakin rendah. Karena instalasinya sistemnya*
9	7	**Usability**	0,5	4	Is ease of use a primary criteria for acceptance? **The greater the importance of usability, the higher the number.**	*semakin penting nilai usability bagi kesuksesan sistem, nilainya semakin tinggi. Tapi kalau bodo amat dengan UI/UX seperti Shopee, nilainya boleh rendah.*
10	8	**Cross-Platform Support**	2	3	Is multi-platform support required? **The more platforms that have to be supported** (this could be browser versions, mobile devices, etc. or Windows/OSX/Unix), **the higher the value.**	*semakin banyak platform yg harus dibuat untuk sistem ini, nilainya semakin besar. Hanya di desktop vs di desktop,mobile,videotron nilainya berbeda.*
11	9	**Easy To Change**	1	4	Does the customer require the ability to change or customize the application in the future? **The more change / customization that is required in the future, the higher the value.**	*Semakin sering sistem harus butuh update/versioning di masa depan, nilainya semakin tinggi.*
					Will you have to address database locking and other concurrency issues? **The more attention you have to spend to resolving conflicts in the data or**	*semakin banyak transaksi data yg harus melalui proses commit database secara simultan (bersamaan di banyak*



		Technical Factor	Multiplier	Relative Magnitude (Enter 0-5)	Description	
4	2	**Response Time Is Important**	1	5	For example, if the server load is expected to be very low, this may be a trivial factor. **Higher numbers represent increasing importance of response time** (a search engine would have a high number, a daily news aggregator	*semakin pentingnya response time jadi ukuran keberhasilan sistem, nilainya semakin tinggi. Response time-> contoh waktu loading page sampai muncul konten*
5	3	**End User Efficiency**	1	4	Is the application being developed to optimize on user efficiency, or just capability? **Higher numbers represent projects that rely more heavily on the application to improve user efficiency.**	*semakin pentingnya pemahaman bahwa user harus cepat dan merasa mudah pakai aplikasi ini, nilainya semakin tinggi. Tapi kalau bodo amat dengan itu, yg penting sistemnya kompleks bisa palugada, maka pasang nilai rendah.*
6	4	**Complex Internal Processing Required**	1	4	Is there a lot of difficult algorithmic work to do and test? Complex algorithms (resource leveling, time-domain systems analysis, OLAP cubes) have higher numbers. **Simple database queries would have low numbers.**	*semakin kompleks algoritma yg dipakai nilainya semakin tinggi.*
7	5	**Reusable Code Must Be A Focus**	1	3	effort required to deploy a project. It also reduces the amount of time required to debug a project. A shared library function can be re-used multiple times, and fixing the code in one place can resolve multiple bugs. **The higher the**	*semakin sering source codes dipakai berulang kali di banyak function/fitur, nilainya semakin kecil*
8	6	**Installation Ease**	0,5	3	Is ease of installation for end users a key factor? **The higher the level of competence of the users, the lower the number.**	*semakin usernya paham cara instalasi (misal sesama orang IT), nilainya semakin rendah. Karena instalasinya sistemnya*
9	7	**Usability**	0,5	4	Is ease of use a primary criteria for acceptance? **The greater the importance of usability, the higher the number.**	*semakin penting nilai usability bagi kesuksesan sistem, nilainya semakin tinggi. Tapi kalau bodo amat dengan UI/UX seperti Shopee, nilainya boleh rendah.*
10	8	**Cross-Platform Support**	2	3	Is multi-platform support required? **The more platforms that have to be supported** (this could be browser versions, mobile devices, etc. or Windows/OSX/Unix), **the higher the value.**	*semakin banyak platform yg harus dibuat untuk sistem ini, nilainya semakin besar. Hanya di desktop vs di desktop,mobile,videotron nilainya berbeda.*
11	9	**Easy To Change**	1	4	Does the customer require the ability to change or customize the application in the future? **The more change / customization that is required in the future, the higher the value.**	*Semakin sering sistem harus butuh update/versioning di masa depan, nilainya semakin tinggi.*
12	10	**Highly Concurrent**	1	3	Will you have to address database locking and other concurrency issues? **The more attention you have to spend to resolving conflicts in the data or application, the higher the value.**	*semakin banyak transaksi data yg harus melalui proses commit database secara simultan (bersamaan di banyak tempat), nilainya semakin tinggi. Misalnya banyaknya*
13	11	**Custom Security**	1	4	developed? **The more custom security work you have to do** (field level, page level, or role based security, for example), **the higher the value.**	*semakin banyak level keamanan yang diterapkan di sistem, nilainya semakin tinggi.*
14	12	**Dependence On Third-Party Code**	1	3	Will the application require the use of third party controls or libraries? Like re-usable code, third party code can reduce the effort required to deploy a solution. **The more third party code** (and the more reliable the third party code), **the lower the number.**	*semakin banyak source code/library yg dipakai dari sumber eksternal (misal ambil dari resource Git yg terpercaya), nilainya semakin rendah. Karena semakin ringan programmernya berpikir pemecahan masalah.*
15	13	**User Training**	1	2	complex activities? **The longer it takes users to cross the suck threshold**	*semakin lama/sulit usernya diajari, nilainya semakin tinggi.*
16		Calculated TCF		1,075		

Environmental Factor



	Environmental Factor	Multiplier	Relative Magnitude (Enter 0-5)	Description
1	Familiarity With The Project	1,5	4	How much experience does your team have working in this domain? The domain of the project will be a reflection of what the software is intended to accomplish, not the implementation language. In other words, for an insurance compensation system written in java, you care about the team's experience in the insurance
2	Application Experience	0,5	4	How much experience does your team have with the application. This will only be relevant when making changes to an existing application. **Higher numbers represent more experience**. For a new application, everyone's experience will be 0.
3	OO Programming Experience	1	3	many people have no object oriented programming experience if you are used to having it. A user-centric or use-case-driven project will have an inherently OO structure in the implementation. **Higher numbers represent more OO experience.**
4	Lead Analyst Capability	0,5	4	How knowledgeable and capable is the person responsible for the requirements? Bad requirements are the number one killer of projects - the Standish Group reports that 40% to 60% of defects come from bad requirements. **Higher numbers represent increased skill and knowledge.**
5	Motivation	1	5	How motivated is your team? **Higher numbers represent more motivation.**
6	Stable Requirements	2	3	Changes in requirements can cause increases in work. The way to avoid this is by planning for change and instituting a timing system for managing those changes. Most people don't do this, and some rework will be unavoidable. **Higher numbers represent more change** (or a less effective system for managing change).
	Part-Time			Note, **the multiplier for this number is negative**. Higher numbers reflect team members that are part time, outside consultants, and developers who are splitting

isi hanya di bagian kolom E (warna kuning). Kolom D adalah bobot komponen yang tidak perlu diubah sebagai pengali nilai yang dilsikan)

semakin lama pengalaman programmer mengerjakan project software, nilainya semakin tinggi

semakin lama pengalaman programmer mengerjakan project software YANG SERUPA DENGAN YANG SEKARANG DIKERJAKAN, nilainya semakin tinggi

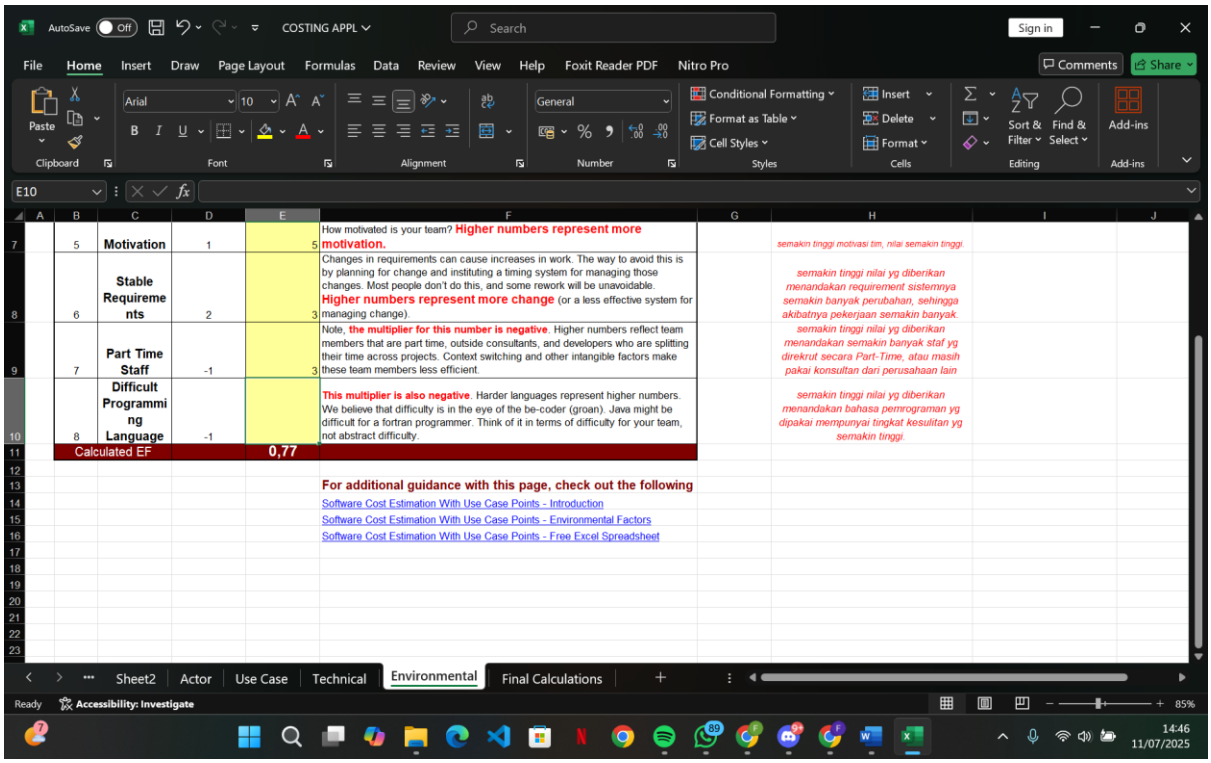
semakin lama pengalaman tim mengerjakan projek berbasis OO (Object Oriented Programming), nilainya semakin tinggi.

semakin senior posisi Lead Analyst (ketua regu para sistem analyst), nilainya semakin tinggi.

semakin tinggi motivasi tim, nilai semakin tinggi.

semakin tinggi nilai yg diberikan menandakan requirement sistemnya semakin banyak perubahan, sehingga akibatnya pekerjaan semakin banyak.

semakin tinggi nilai yg diberikan menandakan semakin banyak staf yg



5	Motivation	1	5	How motivated is your team? **Higher numbers represent more motivation.**
6	Stable Requirements	2	3	Changes in requirements can cause increases in work. The way to avoid this is by planning for change and instituting a timing system for managing those changes. Most people don't do this, and some rework will be unavoidable. **Higher numbers represent more change** (or a less effective system for managing change).
7	Part Time Staff	-1	3	Note, **the multiplier for this number is negative**. Higher numbers reflect team members that are part time, outside consultants, and developers who are splitting their time across projects. Context switching and other intangible factors make these team members less efficient.
8	Difficult Programming Language	-1		**This multiplier is also negative**. Harder languages represent higher numbers. We believe that difficulty is in the eye of the be-coder (groan). Java might be difficult for a fortran programmer. Think of it in terms of difficulty for your team, not abstract difficulty.
	Calculated EF		0,77	

semakin tinggi motivasi tim, nilai semakin tinggi.

semakin tinggi nilai yg diberikan menandakan requirement sistemnya semakin banyak perubahan, sehingga akibatnya pekerjaan semakin banyak.

semakin tinggi nilai yg diberikan menandakan semakin banyak staf yg direkrut secara Part-Time, atau masih pakai konsultan dari perusahaan lain

semakin tinggi nilai yg diberikan menandakan bahasa pemrograman yg dipakai mempunyai tingkat kesulitan yg semakin tinggi.

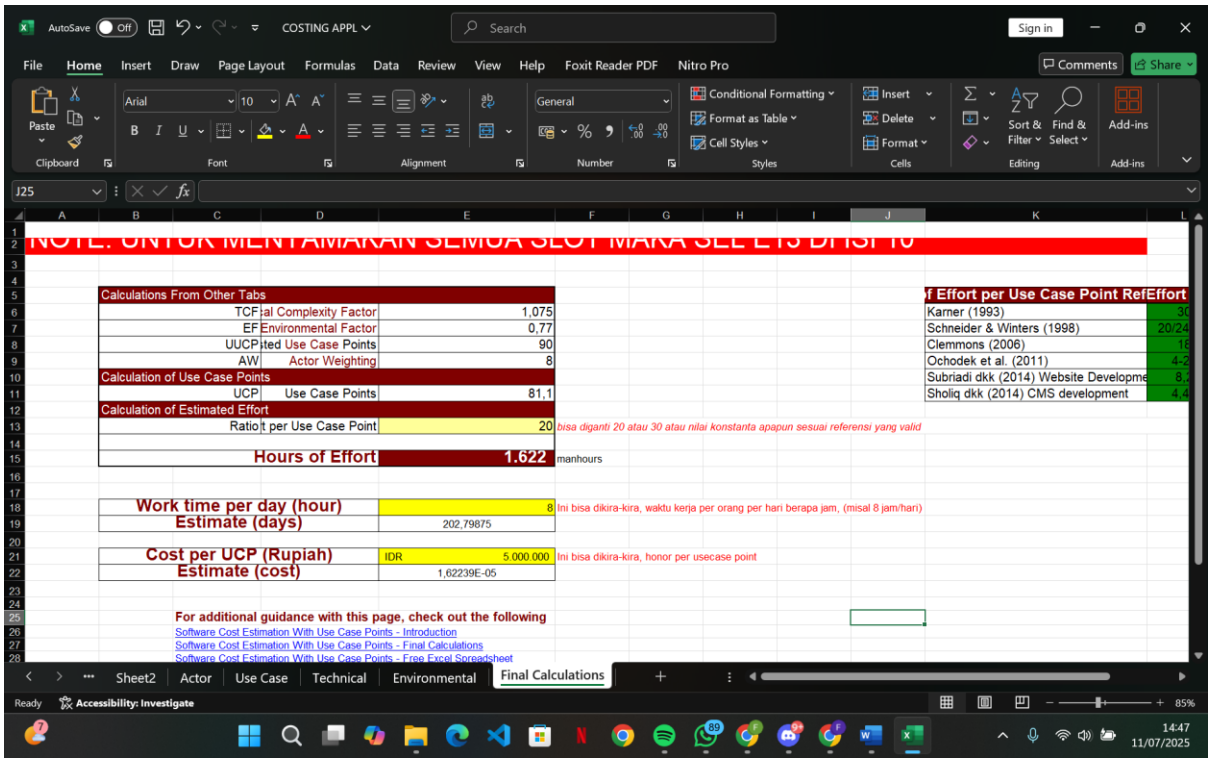
For additional guidance with this page, check out the following

Software Cost Estimation With Use Case Points - Introduction

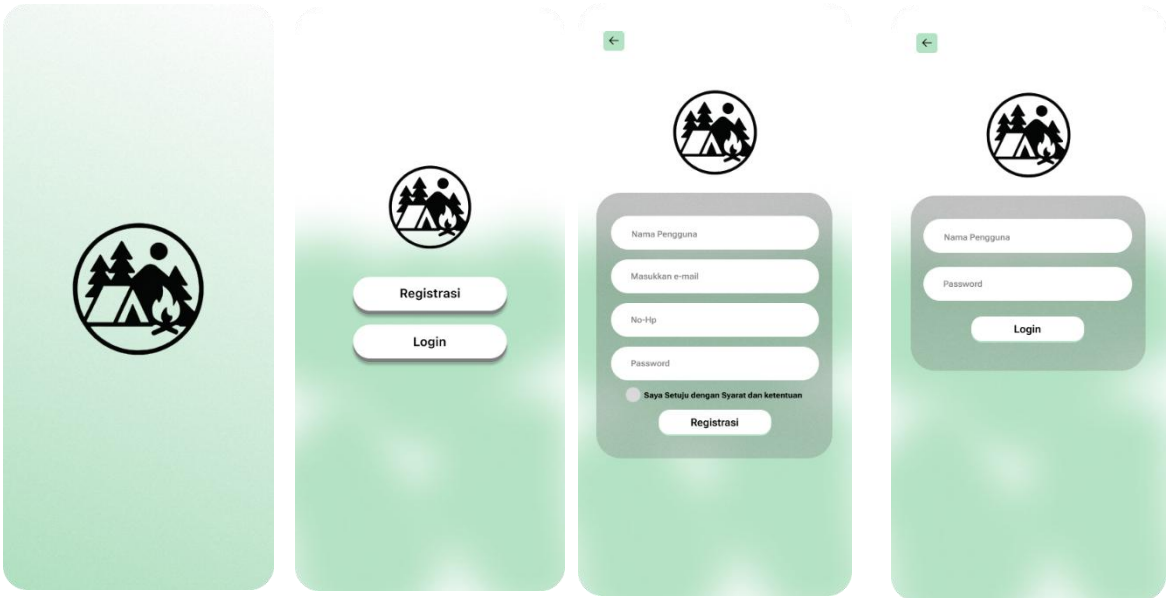
Software Cost Estimation With Use Case Points - Environmental Factors

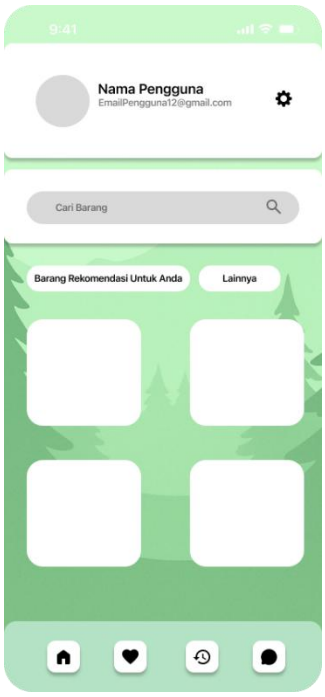
Software Cost Estimation With Use Case Points - Free Excel Spreadsheet

Final calculations



Desain Antarmuka




9:41
Nama Pengguna
EmailPengguna12@gmail.com
Cari Barang
Barang Rekomendasi Untuk Anda
Lainnya


9:41
Hai Pengguna!
LogOut
Nama Pengguna
EmailPengguna
No-Hp


9:41
Cari Barang


9:41
Favorit Anda


9:41
Riwayat


9:41
Chat Toko
Admin Toko CampingKen