Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer e-ISSN: 2548-964X
Vol. 4, No. 12, Desember 2020, hlm. 4291-4299 http://j-ptiik.ub.ac.id

Pengembangan Sistem Manajemen Proyek dan Akun *Hosting* di *Software House* Berbasis *Web* (Studi Kasus Elecomp Software House)

Febiyana Nur Yahya[1], Achmad Arwan[2], Agi Putra Kharisma[3]

Program Studi Teknik Informatika, Fakultas Ilmu Komputer, Universitas Brawijaya
Email: [1]yfebiyan@student.ub.ac.id, [2]arwan@ub.ac.id, [3]agi@ub.ac.id

Abstrak

Elecomp *Software house* adalah perusahaan pengembang *software* yang ada di Kota Malang. Elecomp *Software house* menerima pembuatan aplikasi yang berupa *website* dan *mobile*. Dalam menjalankan kegiatan sehari-harinya, elecomp *software house* mendapati beberapa masalah. Masalah tersebut seperti tidak adanya integrasi data akun *hosting* dan *domain* yang digunakan pada proyek milik client yang elecomp kerjakan, pengingatan pembayaran *domain* sesuai permintaan waktu dari tiap *client* secara satu persatu, monitoring progres pengerjaan proyek yang masih manual serta revisi proyek dari klien dan pelaporan progres yang memakan waktu lama. Tujuan dari penelitian ini adalah untuk membangun sistem yang bisa mengintegrasikan data akun *hosting*, *domain* dan proyek dengan baik, mengingatkan pembayaran *domain* ke klien secara otomatis, memudahkan dalam pembagian sumber daya dan monitoring dalam pengerjaan proyek, serta dapat memudahkan klien dalam monitoring dan revisi proyek miliknya. Penelitian ini dikerjakan menggunakan SDCL *Waterfall* sebagai metodenya dan OOP atau pemrograman berorientasi objek sebagai metode perancangan dan pemrogramanya. Setelah sistem selesai dikerjakan, dilakukan pengujian. Pengujian ini ada tiga, pengujian unit pada tiga *method* yang menghasilkan total 19 kasus uji, hasil pengujian 100% *passed*. Pengujian integrasi menghasilkan 2 kasus uji, hasilnya 100% *passed*. Pengujian validasi menguji 78 fungsionalitas dan menghasilkan 115 kasus uji, hasilnya 100% valid.

Kata kunci: *pengembangan, manajemen proyek, software house, hosting, waterfall, oop*

Abstract

Elecomp Software house is a software development company in the city of Malang. Elecomp Software house accepts the creation of applications in the form of websites and mobile. In carrying out its daily activities, elecomp software house encountered several problems. These problems include the lack of integration of hosting account and domain data used in client-owned projects that are being worked on, domain payment reminder according to the time request from each client one by one, monitoring project progress that is still need manual work as well as revision of the project from the client and progress reporting that take a long time. The purpose of this research is to build a system that can integrate hosting account, domain and project data, remind domain payments to clients automatically, facilitate the distribution of resources and monitoring in project work, and can facilitate clients in monitoring and revising their projects. . This research was carried out using SDCL Waterfall as the method, and OOP or object-oriented programming as the design and programming method. After the system is finished, testing is done. There are three tests, unit tests on three methods that produce a total of 19 test cases, the test results are 100% passed. Integration testing resulted in 2 test cases, the results were 100% passed. Validation testing tests 78 functionalities and produces 115 test cases, the results are 100% valid..

Keywords: *development, project management, software house, hosting, waterfall, oop*

1. PENDAHULUAN

Elecomp *Software house* adalah perusahaan pengembang *software* dan *website* yang ada di Kota Malang. Elecomp *Software house* menerima pemesanan *website* (toko online, sistem informasi, *website*, company pro*file*, dan lain-lain), program/*Software* komputer (program kasir, accounting, dan lain - lain), yang berbentuk aplikasi Android dan aplikasi Web (Fernandes, 2020). Elecomp didirikan oleh Fernandes, S.Kom sejak tahun 2014 sampai dengan sekarang. Perusahaan ini sudah mendapat beragam permintaan pembuatan *software* dari berbagai kalangan, mulai dari perorangan, usaha-usaha kecil, pemerintahan, hingga luar negeri.

Elecomp membutuhkan jasa pihak ketiga untuk melakukan *hosting* aplikasinya. Setiap aplikasi memiliki *domain*nya masing-masing, dalam satu *domain* bisa ditempati oleh lebih dari satu aplikasi. Elecomp harus memastikan semua data tersebut terpetakan dengan baik dengan aplikasi yang ada. Selama ini Elecomp hanya membuat daftar nama *domain* dan nama *project* yang ada di dalamnya di dalam *file excel*. Hal ini kurang efisien karena tidak adanya integrasi data antara detail *project* dan *domain* yang digunakannya serta *client* yang memilikinya. Hal ini menjadi masalah terutama saat ingin mencari *client* yang harus diingatkan untuk pembayaran domainnya. Karena data tersebut berada di tempat yang berbeda, sehingga elecomp harus melakukan *cross reference* secara manual.

Elecomp memiliki kebijakan untuk mengingatkan kepada *client* mengenai pembayaran domain yang digunakan *project* milik mereka. Permintaan waktu *reminder* dari *client* untuk masalah batas akhir pembayaran *domain* beragam. Proses pemberitahuan ke masing-masing *client* dilakukan dengan memberi tahu satu persatu melalui aplikasi Whatsapp. Elecomp harus menyesuaikan waktu pemberitahuan sesuai keinginan *client* dengan data *domain* yang digunakan oleh tiap *project*. Masalah yang timbul adalah seperti keterlambatan pemberitahuan. Dengan terlambatnya pemberitahuan ke *client*, *domain* tidak dibayar dan aplikasi yang menggunakan *domain* tersebut tidak bisa diakses. Tidak bisanya akses aplikasi akan menimbulkan komplain dari *client.*

Kegiatan untuk memperoleh data yang dilakukan dengan monitoring project sangat penting untuk menghindari adanya ketidaklengkapan milestone project serta digunakan untuk membuat keputusan yang dapat mengontrol keluaran project (Russel J, 2018). Proses monitoring progres dan pembagian tugas masing-masing developer selama ini masih manual yakni dengan memberi tahu ke developer melalui tatap muka. Pemilik software house harus bertemu dengan developer untuk bisa melihat progres pengerjaan aplikasi dan memberikan tugas secara detail. Hal ini sulit dilakukan karena sibuknya pemilik software house yang sering melakukan acara di luar kantor.

Pemilik *software house* kerap menanyakan progres aplikasi kepada *developer* melalui Whatsapp jika pemilik sedang tidak ada di kantor. *Client* sering bertanya bagaimana progres aplikasi milik mereka. Pemilik *software* bertanya kepada *developer* mengenai progres *project*. Setelah mendapat respon dari *developer*, pemilik memberitahu progres tersebut kepada *client*. Disini terjadi lamanya proses pemberitahuan progres. Pada bagian owner menjadi lama karena kesibukan dan tidak bisa memberikan respon dengan cepat. Pada bagian *developer* menjadi lama karena harus mengecek aplikasi yang dikerjakan terlebih dahulu. Dengan banyaknya pesan Whatsapp yang masuk proses pemberitahuan progres ke *client* menjadi lama. Jika pesan tertumpuk kemudian tenggelam bisa mengakibatkan *client* harus mengirim pesan lebih dari sekali. Hal tersebut menyebabkan ketidakpuasan *client* oleh layanan dari Elecomp.

Maka dari itulah diusulkan sebuah sistem manajemen *project* dan akun *hosting* di *software house* berbasis web pada Elecomp *Software house*. Sistem ini melakukan manajemen semua data *domain* di akun *hosting* yang dipakai, pengingat pembayaran otomatis, Pembagian alokasi sumber daya, monitoring progres pengerjaan dan revisi *project* bagi *admin* atau owner, *developer* serta *client.*

2. LANDASAN KEPUSTAKAAN

2.1. Tinjauan Pustaka

Penelitian yang dilakukan oleh Try Ratnasari (2017) mengenai pembangunan sistem manajemen *project* untuk pengembang perangkat lunak didasari pada masalah kurangnya tenaga kerja dan tidak adanya

integrasi antar data *project*. Hal ini menyebabkan penyajian informasi perangkat lunak tidak akurat. Sistem ini bisa melakukan alokasi sumber daya, penjadwalan *project*, pelaporan status *project* dan dokumentasi *project*. Pengembangannya dilakukan dengan menggunakan SDLC *Waterfall*. Platform yang digunakan dalam pembuatan sistem ini adalah berbasis website.

Penelitian yang dilakukan oleh Wildan Suharso (2018) mengangkat topik tentang manajemen *project* perangkat lunak. Penelitian ini membuat sebuah sistem manajemen perangkat lunak yang di dalamnya menggunakan metode scrum. Dalam hal estimasi waktu dan biaya *project* digunakan algoritma CoCoMo (*Constructive Cost Model*). Implementasi yang dilakukan pada penelitian ini tidak hanya menggunakan scrum tetapi membuat aplikasi di mana seluruh aktivitas scrum dapat dilakukan, meliputi product backlog, sprint planning, sprint, dan sprint review. Teknologi yang digunakan dalam sistem ini adalah CI (Codeigniter) yang merupakan *framework* untuk bahasa pemrograman PHP.

Penelitian yang dilakukan oleh Ardian Riftha (2017) adalah penelitian mengenai Sistem manajemen *project* berbasis web pada perusahaan konstruksi dan manufaktur. Adanya penelitian ini dikarenakan terdapat masalah dalam tidak adanya alat untuk mengukur kesesuaian antara perencanaan dengan realisasi yang ada di lapangan. Pemangku kepentingan juga tidak bisa melihat data secara realtime. Sistem ini dibuat agar bisa melakukan pengawasan *project* dan memberikan data kepada masing-masing pemangku kepentingan secara cepat. Pengembangan yang dilakukan menggunakan konsep MVC (*Model View Controller*.

2.2. NodeJS

Node JS adalah sebuah *runtime* Javascript yang berjalan secara Asinkron dan *non-blocking*. Menurut penjelasan di situs resmi Node JS, Node JS digunakan untuk membangun sebuah aplikasi yang berjalan di server menggunakan Javascript. Untuk melakukan eksekusi program di server dengan baik, dibutuhkan sebuah *Engine*. Node JS menggunakan *Engine* V8 dari Google.

2.3. Manajemen Project

Project adalah adalah suatu upaya yang dilakukan sementara untuk menghasilkan sebuah produk yang unik. Hal ini menjelaskan bahwa *project* dikerjakan hanya sekali. Manajemen *project* adalah pengaplikasian pengetahuan, kemampuan dan teknik dalam membuat suatu *project* yang bertujuan untuk memenuhi kebutuhan *project* tersebut. manajemen *project* dapat dicapai dengan melalui proses *initiating, planning, executing, monitoring, controlling* dan *closing.* (Heagney, 2012).

2.4. Web Hosting

Web *Hosting* merupakan tempat di mana *file*, data dan segala macam hal yang diperlukan oleh suatu *website* dapat disimpan dan bisa diakses melalui internet (Waryanto, 2018). Tiap – tiap layanan *hosting* memiliki paket *hosting* yang berbeda. Paket ini dibedakan dalam Bandwithnya, kapasitas penyimpanannya, harganya, jumlah *domain* yang bisa dimiliki hingga jenis *hosting* itu sendiri. Beberapa jenis *hosting* diantaranya adalah *shared hosting*, *Cloud Hosting* dan VPS atau *virtual private server.*

3. METODOLOGI PENELITIAN

Alur metodologi dapat dilihat pada gambar 1

Gambar 1. Metodologi Penelitian

Studi literatur dilakukan untuk mencari data yang dapat mendukung dalam masalah dan penyelesaian dalam penelitian. Data tersebut bisa berasal dari buku, jurnal, studi kasus, serta wawancara. Literatur lainnya meliputi pengembangan perangkat lunak, pendekatan berorientasi objek, pengujian perangkat lunak yang terbagi menjadi *Black box* Testing dan *White box* Testing serta SDLC atau *software* development life cycle. SDLC yang digunakan dalam penelitian ini adalah dengan model *waterfall*. Kemudian digunakan juga literatur mengani teknologi yang digunakan seperti Node JS dan PostgreSQL. Selain itu literatur lainnya juga ditambahkan seperti penjelasan mengenai web *hosting.*

Analisis kebutuhan dilakukan dengan wawancara kepada pihak *Software house*. Wawancara dilakukan beberapa kali. wawancara pertama menanyakan hal-hal umum seperti penjelasan Elecomp, kegiatan yang dilakukan di dalamnya, serta bagaimana elecomp bekerja. Wawancara kedua mengutarakan masalah yang dipilih. Dari masalah yang terpilih, pemilik elecomp menjelaskan lebih dalam di masing-masing masalah tersebut. Diskusi dengan pemilik untuk menentukan aktor dan fungsionalitasnya. Setelah itu membuat *use case* diagram dan *scenario* dari kebutuhan yang sudah didapatkan.

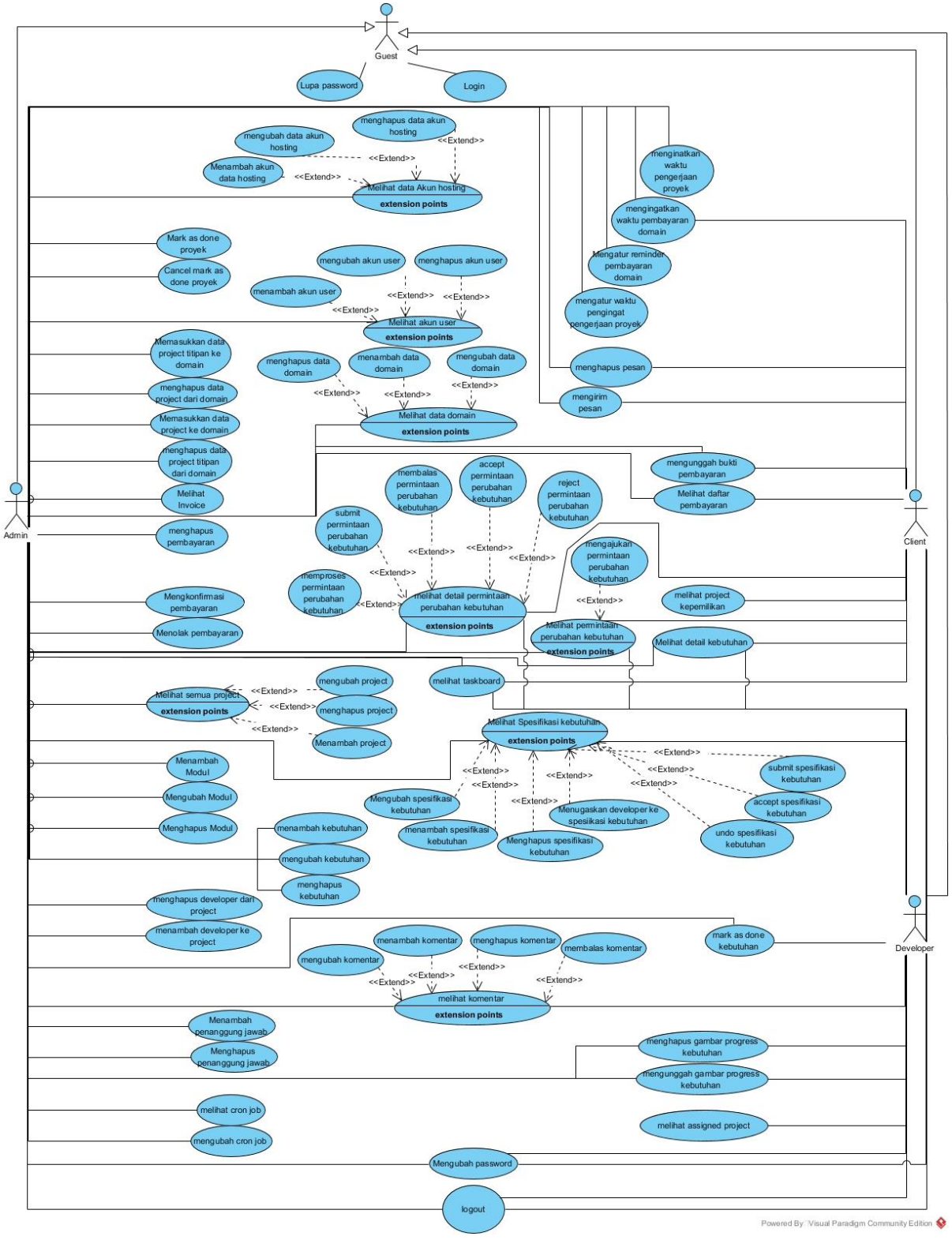
Perancangan dilakukan dengan membuat *sequence* diagram dari *scenario*, membuat *class* diagram, algoritme, perancangan data menggunakan ERD, dan perancangan antarmuka. Setelah proses perancangan selesai, selanjutnya dilakukan proses implementasi. Proses implementasi ini mengacu pada perancangan yang sudah dibuat sebelumnya. Diimplementasikan arsitektur, rancangan komponen, rancangan data hingga pembuatan antarmuka.

Pengujian dilakukan terhadap sistem menggunakan pengujian *black box* dan pengujian *white box*. Pengujian *white box* digunakan untuk menguji alur kode dan logika program sistem. Pengujian *white box* ini meliputi pengujian *unit* dan integrasi. Pengujian *black box* digunakan untuk menguji kebutuhan yang sudah didefinisikan. Pengujian ini diterapkan pada masing-masing kondisi yang ada pada kebutuhan tanpa melihat kode program. Terakhir pemberian kesimpulan dan saran. Hal ini mengacu terhadap penelitian yang telah dilakukan. Kesimpulan yang diberikan harus bersesuaian dengan rumusan masalah pada penelitian ini. Saran yang diberikan bisa ditujukan untuk pengembangan lanjut maupun masukan kepada peneliti selanjutnya.

4. ANALISIS KEBUTUHAN

penelitian melakukan wawancara dan diskusi dengan pemilik *software house* untuk menentukan kebutuhan mana saja yang bisa diterapkan di dalam sistem yang akan dibuat. Dari proses wawancara dan diskusi, terbentuklah kebutuhan-kebutuhan yang digunakan oleh masing-masing aktor dalam interaksinya dengan sistem. Aktor tersebut adalah *guest, admin* (pemilik), *developer*, dan *client.*

Setelah semua kebutuhan di dapatkan dan aktor sudah teridentifikasi, selanjutnya membuat *use case* diagram. *Use case* diagram digunakan untuk memetakan apa yang bisa masing-masing aktor lakukan dalam sistem yang dilihat dari sudut pandang sistem. *Use case* diagram ini ditarik dari kebutuhan yang sudah terdefinisi. Setelah *use case* diagram dibuat, selanjutnya adalah membuat *use case scenario*. *Use case* ini dibuat untuk menjelaskan *scenario* yang terjadi pada masing-masing *use case* yang ada pada *use case* diagram. *Class* diagram dapat dilihat pada gambar 2

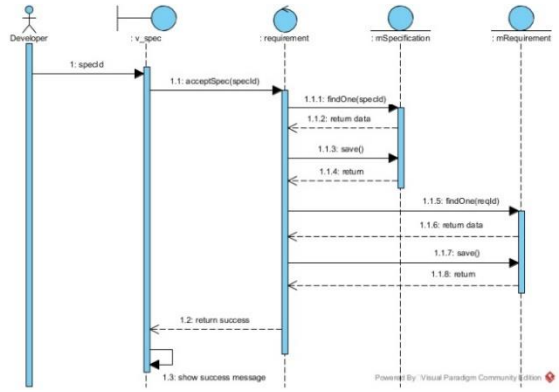


Gambar 2. Use Case Diagram

5. PERANCANGAN DAN IMPLEMENTASI

5.1. Perancangan arsitektur

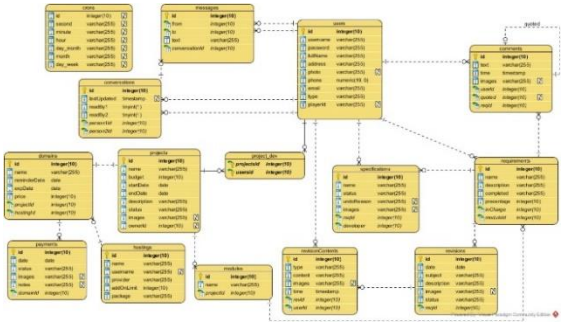
Perancangan arsitektur ini dijelaskan menggunakan *sequence* diagram. *Sequence* diagram sendiri adalah sebuah diagram yang menjelaskan alur dari suatu fungsionalitas dalam sistem berjalan. Di dalam diagram terdapat object yang saling berinteraksi dan bertukar pesan. Diagram juga menjelaskan urutan kejadian dari pesan yang terjadi antar object tersebut. Selain penjelasan dengan *sequence* diagram, arsitektur ini dijelaskan menggunakan *class* diagram yang menunjukkan struktur di dalam sistem yang dibentuk dalam kelas-kelas. *Sequence* dan *class* diagram dapat dilihat di gambar 3 dan 7.



Gambar 3. Sequence Diagram

5.2. Perancangan data

Dalam diagram ini dijelaskan hubungan antara masing-masing entitas. Dalam menggambarkan relasinya, di sini digunakan ERD. ERD dapat dilhat pada gambar 4



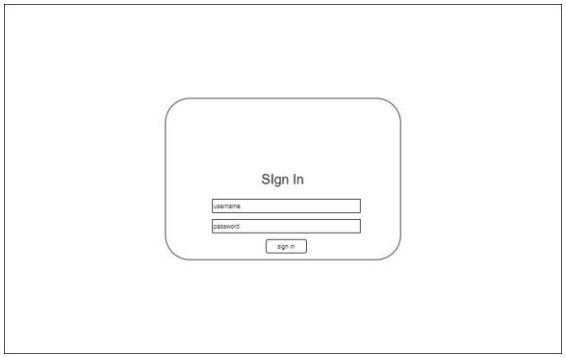
Gambar 4. ERD

5.3. Perancangan komponen

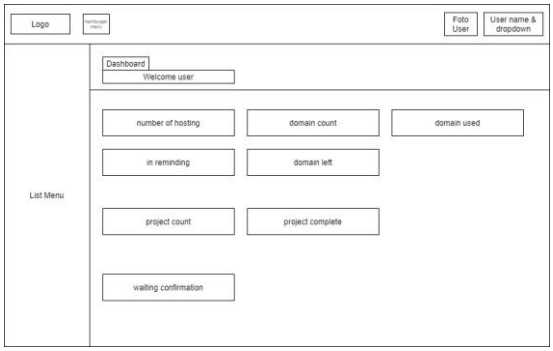
Perancangan komponen dilakukan untuk menjelaskan kode yang berjalan pada sistem. Kode-kode ini nantinya akan berjalan sesuai fungsionalitas yang diberikannya masing-masing. Dalam perancangan komponen dijelaskan menggunakan *pseudocode.*

5.4. Perancangan antarmuka

Perancangan antarmuka ini ditujukan untuk menggambarkan antarmuka atau UI yang dipakai oleh sistem. Antarmuka ini nantinya adalah bagian yang bisa dilihat dan dapat diinteraksikan langsung oleh *user*. Perancangan antarmuka ini digambarkan dengan menggunakan wireframe, bisa dilihat pada gambar 5 dan 6



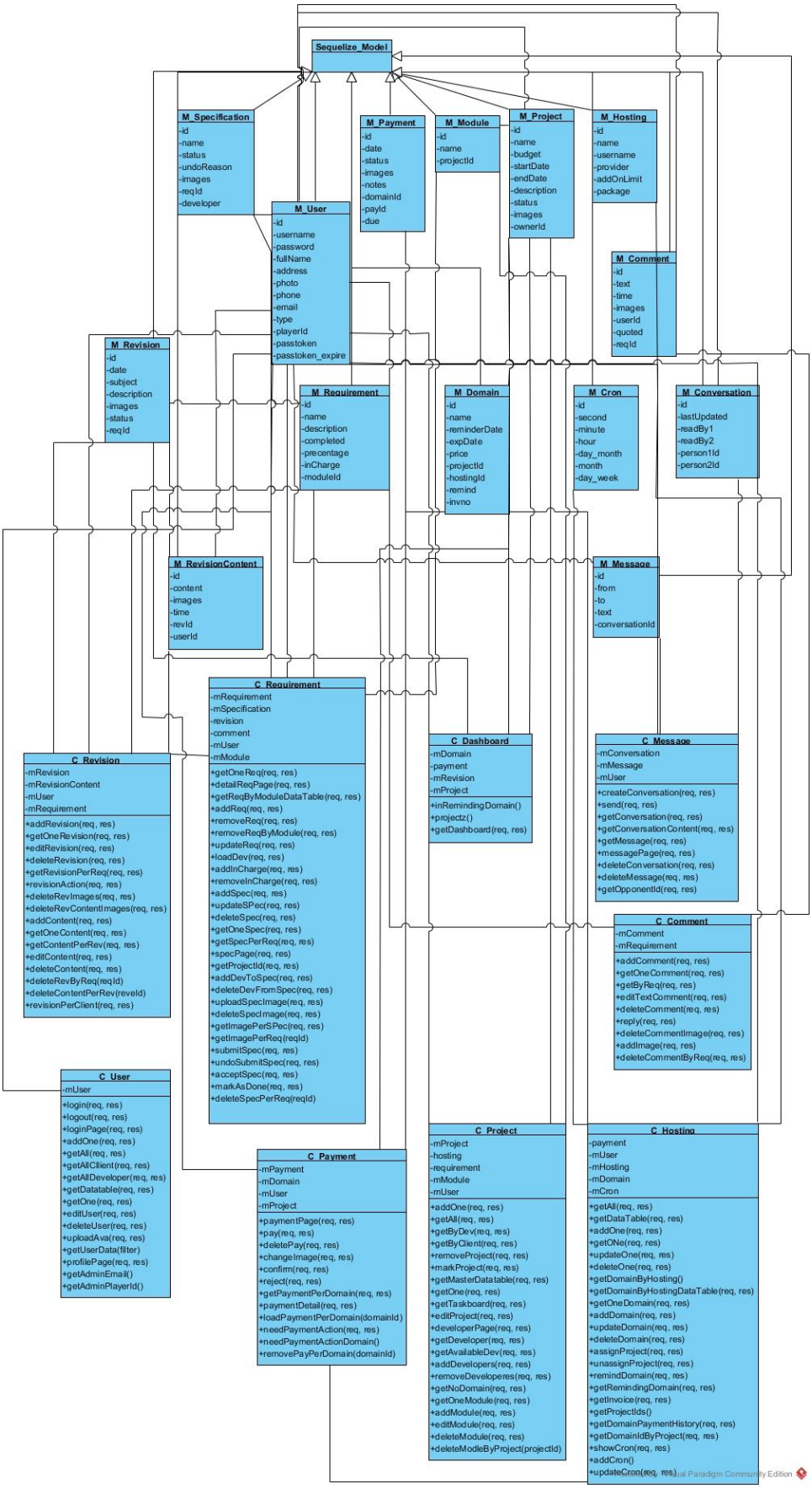
Gambar 5. Perancangan antarmuka 1



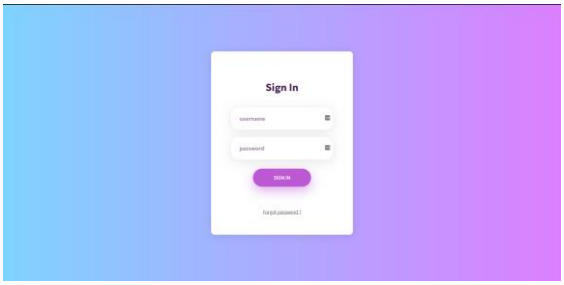
Gambar 6. Perancangan antarmuka 2

5.5. Implementasi sistem

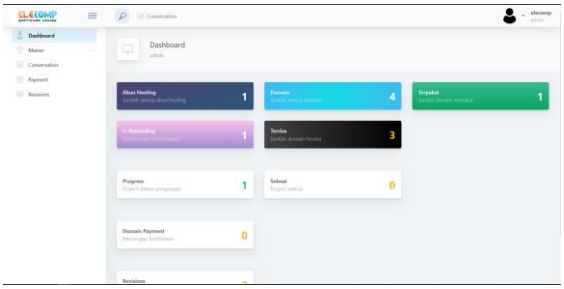
Implementasi sistem ini dilakukan setelah perancangan selesai dilakukan. Semua hal yang sudah dirancang akan diimplementasikan untuk membuat sebuah sistem yang utuh. Implementasi tersebut diantaranya implementasi kode program, yang mana akan menulis kode program tersebut dan dijalankan di dalam sistem. Kemudian implementasi basis data, membuat *entity* yang sudah dirancang lengkap dengan atributnya. Yang terakhir implementasi antarmuka untuk membuat tampilan yang akan dilihat oleh *user*. Implementasi antarmuka dapat dilihat pada gambar 8 dan 9.



Gambar 7. Class Diagram



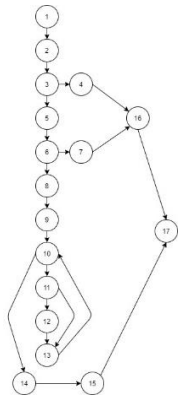
Gambar 8. Implementasi antarmuka 1



Gambar 9. Implementasi antarmuka 2

6. PENGUJIAN

Pengujian yang dilakukan ada tiga macam. Pengujian tersebut adalah unit, integrasi dan validasi. Pengujian unit dilakukan pada tiga fungsi, fungsi tersebut antar lain *acceptSpec()* dari kelas *Requirement* yang menghasilkan 6 *test case*. Fungsi *processRevision()* dari kelas *Revision* yang menghasilkan 5 *test case*, dan fungsi *markAsDone()* pada kelas *Requirement* yang menghasilkan 8 *test case*. Ketiganya menghasilkan 100% *passed. Flow graph* pengujian unit dapat dilihat pada gambar 6.1



Gambar 10. Flow graph (*processRevision()*)

Pengujian integrasi dilakukan pada fungsi *deleteDomain*(). Fungsi *deleteDomain*() dari kelas *Hosting* ini memanggil fungsi removePayPer*Domain*() dari kelas *Payment*. Pengujian ini menghasilkan 2 *test case* dan hasilnya 100% *passed.*

Pengujian validasi dilakukan pada semua kebutuhan. Pengujian ini menghasilkan 115 *test case*. Semua *test case* tersebut hasilnya valid.

7. KESIMPULAN

Analisis kebutuhan menghasilkan 78 kebutuhan fungsional dan 4 aktor. Kebutuhan tersebut kemudian dimodelkan menggunakan pendekatan objek yang menghasilkan 78 *use case* diagram dan *use case scenario.*

Perancangan di Sistem manajemen proyek dan akun *hosting* berbasis web di Elecomp *software house*, menghasilkan *sequence* diagram, *class* diagram, entity relationship diagram, algoritme dan wireframe.

Implementasi sistem manajemen proyek dan akun *hosting* berbasis web di elecomp *software house* menggunakan NodeJS sebagai back-end nya, Express JS sebagai *framework* nya dan PostgreSQL sebagai basis datanya.

Pada bagian pengujian sistem manajemen proyek dan akun *hosting* berbasis web di elecomp *software house*, digunakan pengujian *unit* yang menguji tiga fungsionalitas yang menghasilkan total 19 kasus uji, hasil pengujian 100% *passed*. Pengujian integrasi menghasilkan 2 kasus uji, hasilnya 100% *passed*. Pengujian validasi menguji 78 fungsionalitas dan menghasilkan 115 kasus uji, hasilnya 100% valid.

8. DAFTAR PUSTAKA

About Node.js; Tersedia pada : https://nodejs.org/en/about/.

J Russell, W Pferdehirt, J Nelson. 2018. Technical project Management in Living and Geometric Order. *Board of Regents of University of Wisconsin System.*

Joseph, Heagney., 2012. Fundamentals of project Management (4th edition). *American Management Assosciation.*

Ratnsary, Try., 2018. Rancang Bangun Sistem Informasi Manajemen Project Untuk Pengembang Perangkat Lunak Pada Pt. Quantum Leap. Dalam : *Seminar Nasional Sistem Informasi 2017 Fakultas Teknologi Informasi – UNMER Malang*, 14 September 2017.

Raymond, Fernandes S.Kom. “Proses Bisni di software house”. *Hasil Wawancara Pribadi*: 4 Januari 2020, Elecomp

Software house.

Riftha, Ardian., 2017. Pengembangan Sistem Aplikasi Manajemen Project Berbasis Web (Studi Kasus: PT. Swadaya Graha). Jurnal Pengembangan Teknologi *Informasi dan Ilmu Komputer*, 1(11), hal.1–9.

Suharso, Wildan., 2018. Penerapan Scrum dan Algoritma COCOMO Pada Aplikasi Manajemen Project Perangkat Lunak. *Jurnal SATIN – Sains dan Teknologi Informasi*, 4(1), hal.1–8.

Waryanto. 2018. Apa itu Hosting? Berikut Penjelasannya [daring]. [diakses 2020 Jan 8];Tersedia pada: https://www.niagahoster.co.id/blog/hosting-adalah/