JURNAL ILMIAH FIFO **P-ISSN 2085-4315 / E-ISSN 2502-8332**
DOI: http://dx.doi.org/10.22441/fifo.2022.v14i1.002

Pemanfaatan MEAN Stack Dalam Digitalisasi Administrasi Tugas Akhir Menggunakan Kombinasi Iteratif dan Scrum Model

Nina Sevani*[1], Rita Wiryasaputra[2], Jeremy Wijaya[3], Vini Janti Anggelica[4]

[1,2,3,4]*Program Studi Informatika, Fakultas Teknik dan Ilmu Komputer,*
Universitas Kristen Krida Wacana,
Jl. Tanjung Duren Raya No. 4, Jakarta
Email: *[1]nina.sevani@ukrida.ac.id, [2]Rita.wiryasaputra@ukrida.ac.id,
[3]jeremy.2017tin005@civitas.ukrida.ac.id, [4]vini.2017tin003@civitas.ukrida.ac.id
*) Korespondensi author

(***received*****:** 29-03-22, ***revised*****:** 26-04-22, ***accepted*****:** 11-05-22)

Abstract

The best administration of an organization reflects t*he image of an organization that involves leadership, policies, and human relations. In the digitalization era, the administration process such as writing, duplicating, and recording documents transforms more transparent and easier than before. As part of a complex organization, a department has manual administration. It triggers many problems such as human error, the diversity of formating documents, delayed communication with complicated coordination of the complexity of Undergraduate Student's Final Project administration. These circumstances influence the decision-maker to make the proper choice due to limited time. The web digitalization administration using the Traditional Iterative Model with the Scrum Model reduce the issue. The web development using MEAN (MongoDB, Express, Angular, and NodeJS) technology accommodates the administration software as a SPA (Single-Page Application) using JavaScript framework on the client-side and the server-side with fast, general use by developers and independent. This cutting-edge technology will gain an adaptive system with a small work team and affordable cost*.

Keywords*: administration, digitalization, iterative model, MEAN stack, Scrum*

Abstrak

Citra suatu organisasi tercermin dari baiknya penyusunan administrasi organisasi yang melibatkan kepemimpinan, kebijakan, dan hubungan antar manusia. Kegiatan administrasi mencakup kegiatan pengendalian informasi yaitu tulis menulis/mencatat, menggandakan, menyimpan, dimana kegiatannya bertransformasi menjadi lebih praktis dan transparansi pada era digitalisasi. Program studi sebagai bagian dari perguruan tinggi menghadapi kompleksitas administrasi Tugas Akhir (TA) mahasiswa yang menimbulkan beberapa permasalahan antara lain *human error*, keberagaman formating berkas, komunikasi panjang dan koordinasi bertingkat antar unsur dalam perguruan tinggi jika proses administrasi tersebut dikerjakan secara manual. Permasalahan berdampak pada terganggunya pengambilan keputusan, dan dapat diminimalisir dengan pendigitalisasian administrasi TA mahasiswa berbasis web dengan teknologi *open-source MEAN (MongoDB, Express, Angular, dan NodeJS)* yang didukung Google menggunakan metodologi kombinasi antara *Traditional Iterative* model dengan Scrum model. Teknologi *MEAN* mengakomodir pengembangan web administrasi TA sebagai sebuah SPA (*Single-Page Application*) berkerangka pemrograman JavaScript baik dari *client-side* maupun *server-side* dengan akses lebih cepat, lebih umum digunakan oleh pengembang perangkat lunak, dan fungsionalitasnya tidak membebani ketergantungan *server-side.* Upaya ini dilakukan agar perangkat lunak yang dihasilkan dapat lebih adaptif atas perubahan sistem, namun biaya pengembangan terjangkau dan pengendalian dapat dilakukan pada setiap tahap secara transparan dalam sebuah tim kerja kecil.

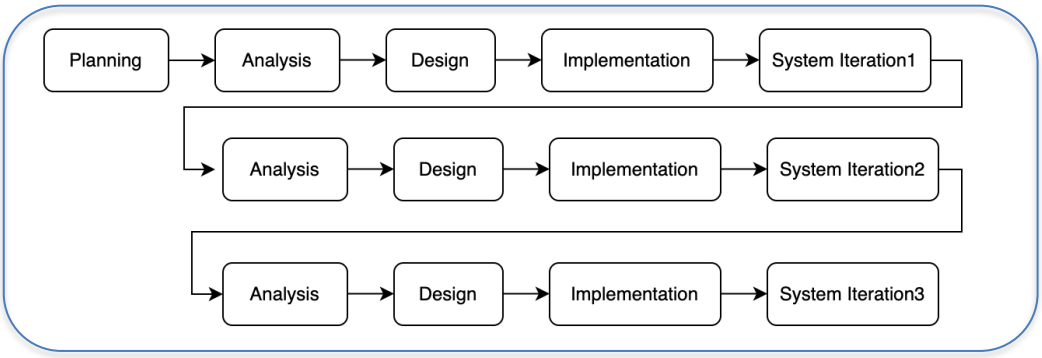
Kata Kunci: administrasi, *digitalisasi, iterative model, MEAN, Scrum*

I. Pendahuluan

Citra suatu organisasi tercermin dari baiknya penyusunan administrasi organisasi yang melibatkan kepemimpinan, kebijakan, dan hubungan antar manusia. Kegiatan administrasi mencakup kegiatan pengendalian informasi yaitu tulis menulis/mencatat, menggandakan, menyimpan, dimana kegiatannya bertransformasi menjadi lebih praktis dan transparansi pada era digitalisasi yang merambah lini kehidupan manusia. Kegiatan administrasi dalam suatu organisasi akan menjadi sumber informasi yang kelak dipergunakan dalam proses pengambilan keputusan, dimana komunikasi koordinasi dan kerjasama akan didukung secara lebih baik. Dalam organisasi perguruan tinggi, untuk menghasilkan lulusan perguruan tinggi yang baik dalam *hardskill* dan *softskill*, maka diperlukan kerjasama antar unsur untuk mengatasi kompleksitas administrasi yang ada dalam berbagai aspek [1]. Salah satunya adalah kompleksitas administrasi Tugas Akhir (TA) mahasiswa juga dihadapi oleh suatu program studi yang merupakan bagian dari perguruan tinggi dan menimbulkan beberapa permasalahan yaitu pertama, *human error* yang meliputi kesalahan dalam pencatatan, kesalahan dalam penyimpanan berkas, kesalahan dalam melakukan unggah berkas. Kedua adalah beragamnya *formatting* berkas, permasalahan ketiga adalah terjadinya komunikasi panjang dan koordinasi bertingkat antar unsur dalam perguruan tinggi yang berujung pada terganggunya waktu pengambilan keputusan. Solusi permasalahan adalah dengan pendigitalisasian proses administrasi TA mahasiswa, sehingga 4 (empat) manfaat yang dapat dipetik adalah percepatan koordinasi antar unsur yang terlibat dalam proses TA (mahasiswa-dosen-koordinator TA), peminimalisasian *human error* dalam pengarsipan TA, pemusatan pengarsipan dengan lebih terstandar dan penghematan biaya kertas.

Pendigitalisasian berkaitan erat dengan pengembangan perangkat lunak, yang merupakan pekerjaan yang tidak mudah dimana kemungkinan kegagalan sistem dapat terjadi sebagai dampak adanya tugas yang tidak lengkap dalam sistem. Penerapan metodologi dan model yang tepat dalam proses pembuatan perangkat lunak terbukti dapat membantu menjaga kualitas dan waktu pengembangan perangkat lunak [2][3][4]. Metodologi ini juga dapat membantu mengatasi berbagai kendala dalam pengembangan perangkat lunak, seperti *human error*, perbedaan kompetensi dari anggota tim, serta kondisi dari masing-masing anggota tim pembuat program dan juga kondisi dari lingkungan tempat kerja [2][3]. Fase-fase dalam metodologi dan model pembuatan perangkat lunak, antara lain akan membentuk manajemen tim untuk alokasi tugas kepada anggota tim, proses konfigurasi anggota tim, pengontrolan, serta mengatasi konflik yang mungkin terjadi [2][4]. Diperlukan adaptasi dengan pendekatan yang lebih baik untuk menciptakan sistem dengan perangkat lunak yang menjadi bagian dari kehidupan manusia [3]. Pergeseran dalam pengembangan perangkat lunak juga terjadi dewasa ini, yaitu dari metodologi tradisional menjadi metodologi yang *agile*. Salah satu metodologi pengembangan *Agile* yang adaptif adalah Scrum *framework*, dimana pada *framework* tersebut pengguna sistem dilibatkan dalam siklus pengembangan yang bertingkat dan berulang [5].

Tetapi, tidak banyak penelitian terkait dengan adaptasi aktual dan modifikasi sistem yang mengimplementasikan Scrum *framework* sesuai dalam kebutuhan dunia nyata. Meskipun hal tersebut merupakan solusi atas kelemahan metode dalam batasan kontekstual yang spesifik. Lebih dari setengah teknik Agile menggunakan Scrum dan kombinasi antara Scrum dengan teknik lainnya [6] sedangkan *Traditional Iterative* model merupakan salah satu model tradisional dalam pengembangan perangkat lunak [4], seperti dapat dilihat pada Gambar 1.



Gambar 1. *Traditional Iterative Model* Source: [4]

Sistem administrasi digital TA berbasis web pada penelitian ini, menyederhanakan koordinasi antar pihak yang terkait dalam proses TA dan pengarsipan terpusat dengan menggabungkan metodologi *Traditional Iterative* model dengan Scrum model. Upaya penggabungan metodologi dengan maksud agar perangkat lunak yang dihasilkan dapat lebih adaptif dalam durasi waktu yang lebih cepat, namun dengan biaya yang terjangkau dan pengendalian

dapat dilakukan pada setiap tahap secara transparansi dalam sebuah tim kerja kecil dan tanggap terhadap adanya perubahan dalam bentuk apapun. Durasi kecepatan proses pengembangan perangkat lunak didukung oleh teknologi Google yang mengembangkan mesin V8 JavaScript yang mampu mengkompile dan mengeksekusi sumber kode JavaScript, menangani pengalokasian memori dan keakuratan manajemen memori yang cukup penting dalam performansi tinggi V8. Beragam *framework* JavaSript lebih sederhana dan berfokus pada teknologi *client-side*, dimulai dari pengembangan *backend* hingga *frontend*, sehingga terjadi upaya pengurangan ketergantungan terhadap teknologi *server-side* pada pengembangan website. *MEAN* terdiri dari komponen *MongoDB, Express, Angular, dan NodeJS* merupakan kombinasi dari beberapa *open-source framework* dengan gaya pengembangan web deklaratif menggunakan mesin V8 JavaScript yang secara efektif memanfaatkan kelebihan ekosistem paket Node sebagai ekosistem terbesar dalam kumpulan *open source* [7]. Angular merupakan salah satu *open-source* JavaScript *framework* yang didukung oleh Google dalam pengembangan *Single Page Application*, sedangkan NodeJS sebagai *server-side cross platform* yang dibangun pada Google Chrome JavaScript Engine berbentuk asynchonous, single-threaded, non-blocking I/O, Express termasuk pengkodean ringan yang didukung multi platform dan pengembang perangkat lunak. Pengembangan web deklaratif merupakan pendekatan yang berfokus pada struktur dan elemen aplikasi dengan mengekspresikan apa yang dilakukukan sistem dalam menyelesaikan permasalahan.

Pemanfaatan modern teknologi pengembangan web *MEAN stack* mengakomodir pendigitalisasian administrasi TA sebagai sebuah SPA (*Single-Page Application*) *fullstack* yang dimana untuk pengembangan sisi *backend* terdiri dari MongoDB sebagai manajemen basis data, Express sebagai *Hypertext Transfer Protocol* (HTTP) server *framework* fleksibel yang melayani aplikasi AngularJS di sisi *frontend*, dan Node yang menggunakan bahasa pemrograman JavaScript baik dari sisi *client-side* maupun *server-side* [7]. SPA merupakan suatu *stand alone application* berbasis web, atau dapat disebut sebagai sebuah halaman web sebagai tampilan aplikasinya dengan efisiensi beban kerja *server* dan *client* dengan tetap menyediakan layanan yang mudah digunakan, user friendly, realtime [8]. Hal ini dimungkikan karena SPA merupakan modular dan *responsive architecture*, dimana setiap komponennya saling responsif satu sama lain [9].

II. Metodologi Penelitian

Penelitian mengadopsi lima fase metodologi pengembangan perangkat lunak yang terdiri dari [4]:

a. *Planning*

Tahap *planning* merupakan tahap awal dalam pengembangan perangkat lunak yang meliputi penentuan masalah yang akan diatasi, termasuk pembentukan tim dengan sumber daya manusia terbatas dan penentuan kebutuhan durasi waktu pengembangan, serta pengumpulan kebutuhan data primer dan data sekunder yang digunakan pada tahap analisa. Bentuk data primer diperoleh melalui teknik observasi, wawancara dan diskusi dengan koordinator TA yang bertugas di program studi, dengan mahasiswa yang mengambil TA, dan dengan dosen pembimbing TA maupun dosen penguji TA. Bentuk data sekunder yang dikumpulkan berupa Standard Operasional Prosedur (SOP) TA yang berlaku, panduan TA yang digunakan, formulir penilaian dan contoh bentuk hasil TA yang dikumpulkan oleh mahasiswa.

b. *Analysis*

Bagian yang dilakukan dari tahap *Analysis* adalah *brainstorming* antara anggota tim untuk membahas lebih dalam tentang masalah yang ditentukan pada tahap planning, termasuk strategi penyelesaian masalahnya. Tim pengembang juga berkoordinasi dengan pihak program studi guna menganalisa kebutuhan perangkat lunak berbasis web yang akan dibangun, merumuskan kebutuhan berdasar proses sistem TA yang berjalan, kondisi lingkungan dan kebutuhan pengguna yang akan mengakses perangkat lunak tersebut baik selaku mahasiswa peserta TA, dosen pembimbing, dosen penguji serta koordinator TA. Pada bagian *analysis* inilah artefak dari model Scrum mulai nampak, karena pada tahap ini setiap anggota tim akan dibentuk untuk mempunyai pemahaman yang sama akan perangkat lunak yang akan dikembangkan.

c. *Design*

Tahap *Design* berisi abstraksi dari analisa yang telah dilakukan dan meliputi pembuatan diagram yang menjelaskan tentang solusi yang diusulkan dari tahap analisa. Diagram yang akan dibuat antara lain *Use Case* diagram dan *Activity* diagram. Schema terkait basis data non relational Dalam tahap *design* juga akan

memuat rancangan basis data dan penggambaran tampilan *frontend* perangkat lunak berbasis web yang akan dibuat. Artefak Scrum secara nyata akan mulai nampak pada tahap *design* seperti terlihat pada Gambar 5 sampai dengan Gambar 8, yang merupakan kelanjutan dari tahap *analysis*.

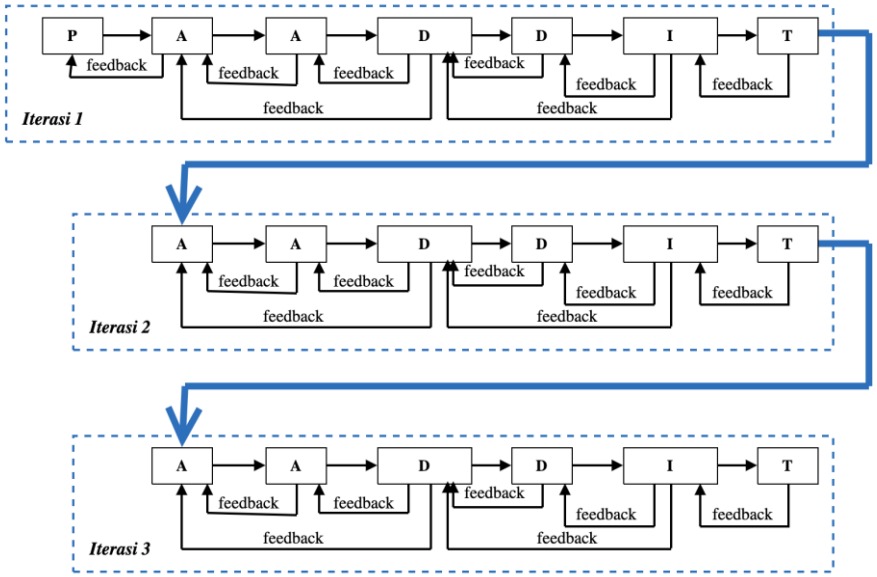
d. *Implementation*

Implementation dalam penelitian menggunakan pengembangan web berteknologi *MEAN stack* untuk menghasilkan website yang deklaratif. *MEAN stack* model mendayagunakan beberapa JavaScript *framework* baik pada sisi bagian *backend* hingga sisi *frontend*. Pada tahap implementation ini, maka tim pengembang akan mulai dapat memperlihatkan hasil nyata dari perangkat lunak yang dikembangkan. Artefak Scrum berupa perangkat lunak yang dikembangkan juga akan mulai muncul pada bagian ini. Bentuk artefak Scrum berupa perangkat lunak dapat dilihat pada Gambar 12, Gambar 13, dan Gambar 14.

e. *Testing*

Tahap testing merupakan tahap untuk memastikan bahwa perangkat lunak yang dibuat dapat berfungsi dengan baik. Penelitian menggunakan *blackbox testing* untuk memastikan apakah perangkat lunak yang dihasilkan dapat berjalan sesuai dengan fungsinya tanpa perlu mengetahui *code* program [10][11]. Dalam *blackbox* ini, pengujian dilakukan untuk setiap *test case* yang dihasilkan dari deskripsi eksternal perangkat lunak [12].

Selain lima fase dalam metodologi pengembangan sistem, penelitian ini juga menggunakan kombinasi model pengembangan sistem, yaitu *Traditional Iterative* model dengan Scrum model. Kombinasi dari kedua model ini merupakan salah satu keunikan dalam pembuatan perangkat lunak administrasi digital TA, mengingat sejauh pengetahuan penulis, belum ada penelitian yang mencoba menggabungkan kedua model ini. Ide dasar kombinasi kedua model ini adalah supaya dapat saling mengurangi kelemahan masing-masing model, sambil terus memaksimalkan kelebihan dari setiap model. Gambar 2 menggambarkan keterkaitan metodologi dan model pengembangan perangkat lunak yang digunakan pada penelitian ini.



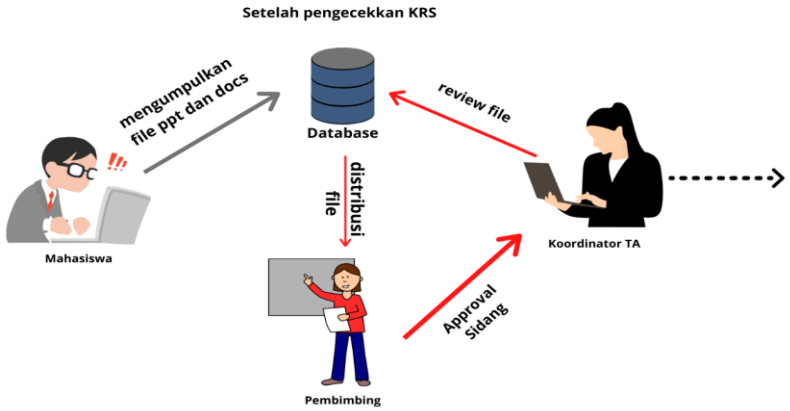
Keterangan:
P= fase Planning,
A= fase Analysis,
D= fase Design,
I = fase Implementation,
T= fase Testing yang menjadi bagian dari maintenance.

Gambar 2. Konsep Permodelan Yang Diusulkan

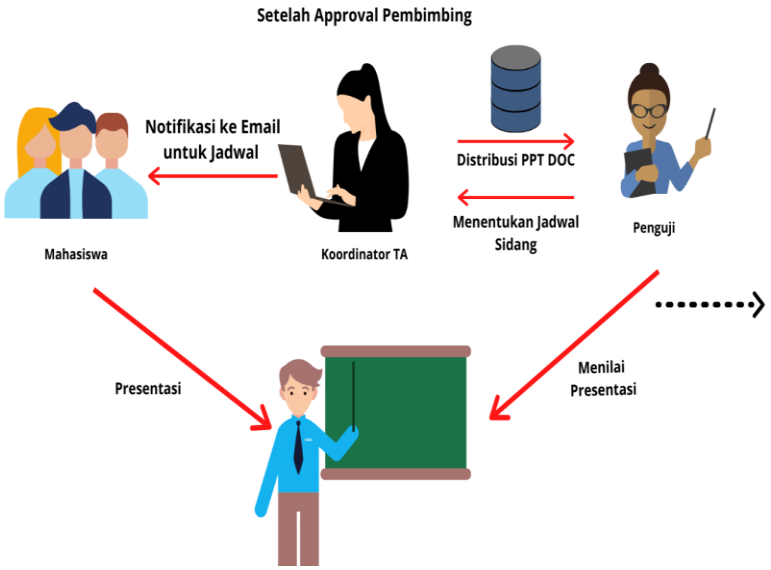
Dalam setiap iterasi, fase-fase yang diperlukan akan tetap dikerjakan. Proses iterasi dilakukan sampai durasi waktu selesai atau perangkat lunak dianggap sudah berfungsi dengan baik sesuai dengan perencanaan awal. Umpanbalik yang fleksibel dikomunikasikan secara regular disiapkan pada tiap fase yang dilakukan oleh setiap blok iterasi. Hasil testing dari satu blok iterasi akan menjadi masukan untuk proses analisa pada blok iterasi berikutnya.

III. Hasil dan Pembahasan

Pembahasan akan dimulai dengan melihat pada proses bisnis yang ada saat ini dan menjadi landasan bagi dilakukannya sistem digitalisasi administrasi yang diusulkan. Pihak Koordinator TA melakukan penentuan dosen pembimbing dan dosen penguji secara manual sehingga subjektivitas mempengaruhi penentuan ini. Proses bisnis yang digambarkan pada Gambar 3 merupakan kondisi yang terjadi sebelum sidang TA, dimana mahasiswa mengumpulkan berkas TA dalam file berformat dokumen ppt dan docs dalam media *database*, selanjutnya Koordinator TA melakukan proses *review* file-file yang dikirimkan oleh mahasiswa tersebut. Peran dosen pembimbing adalah menerima distribusi file, dan setelah proses bimbingan antara dosen pembimbing dan mahasiswa terjadi, maka pembimbing menyetujui mahasiswa mengikuti sidang TA. Adapun Gambar 4 menggambarkan proses bisnis menjelang sidang TA, dimana Koordinator Ta akan menginformasikan mahasiswa tentang jadwal sidang dan mendistribusikan file yang telah dikumpulkan ke dosen penguji. Penguji akan menilai presentasi yang dipaparkan oleh mahasiswa peserta sidang TA. Hanya mahasiswa yang sudah mengumpulkan berkas lengkap yang diverifikasi oleh koordinator TA dan sudah mengisi KRS TA sebagai syarat administrasi yang akan mengikuti proses bisnis ini.



Gambar 3. Proses Bisnis Sebelum Sidang TA



Gambar 4. Proses Bisnis Menjelang Sidang TA

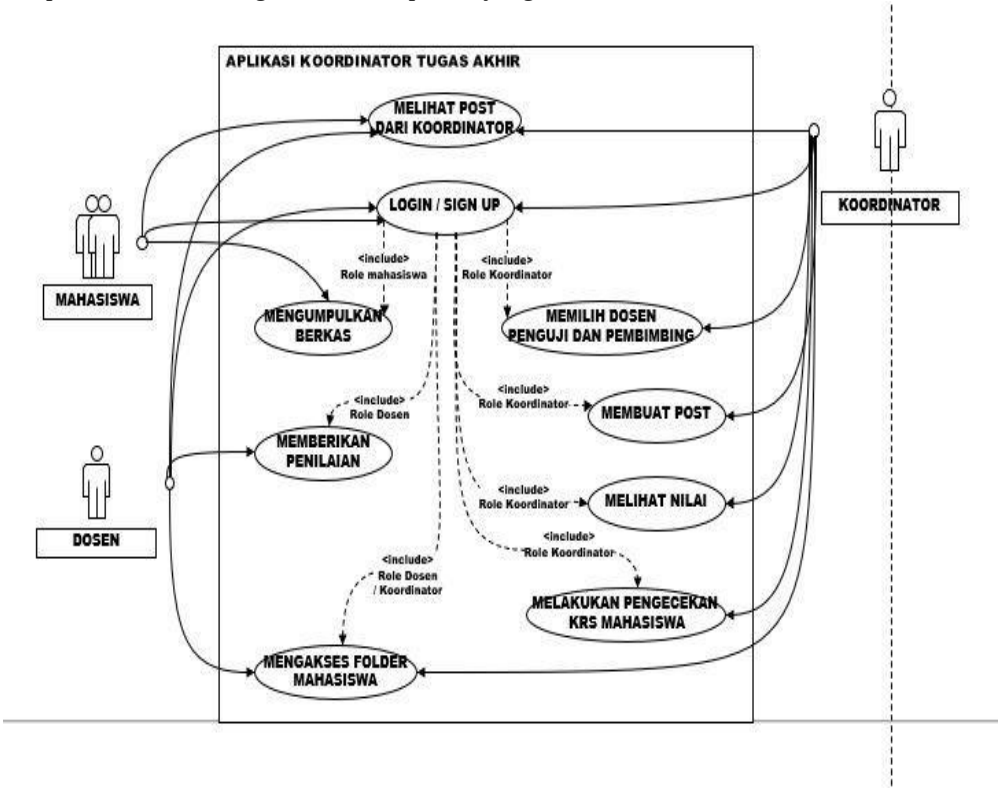
Proses bisnis setelah sidang TA ditunjukkan pada Gambar 5, dimana Koordinator TA akan mengumpulkan nilai para mahasiswa peserta sidang baik dari dosen pembimbing, maupun dosen penguji TA.

Gambar 5. Proses Bisnis Setelah Sidang TA

Permasalahan yang ada pada proses bisnis saat ini adalah karena alurnya yang panjang dan membutuhkan verifikasi dari beberapa pihak, dimana seluruh proses verifikasi masih dilakukan semi manual melalui koordinator TA. Hal ini dapat menimbulkan delay yang cukup lama serta cukup rentan akan *human error*, apalagi pada saat jumlah mahasiswa TA yang cukup banyak. Oleh karena itulah diperlukan sistem digitalisasi administrasi TA yang dapat memotong proses antrian yang bisa panjang dan mempermudah verifikasi dengan tetap memastikan keamanan berkas TA. Untuk memperjelas alur dari sistem digitalisasi administrasi TA yang diusulkan, maka dibuatlah *Use Case* diagram, *Activity* diagram, serta *Schema* dari sistem yang dibuat.

Use Case Diagram

Use case diagram mencerminkan hasil analisis atas penggambaran kebutuhan sistem, dan merupakan diagram kebutuhan fungsional yang cukup populer mempengaruhi keefektifan pengembangan aktivtias lain dalam sistem [13]. Aktor pada *use case diagram* dalam sistem digitalisasi administrasi TA terbagi menjadi 3 (tiga) *role* pengguna yaitu *user*, admin (koordinator), dan admin kedua (Dosen Pembimbing dan Dosen Penguji), digambarkan pada Gambar 6 dengan hak akses proses yang berbeda-beda.

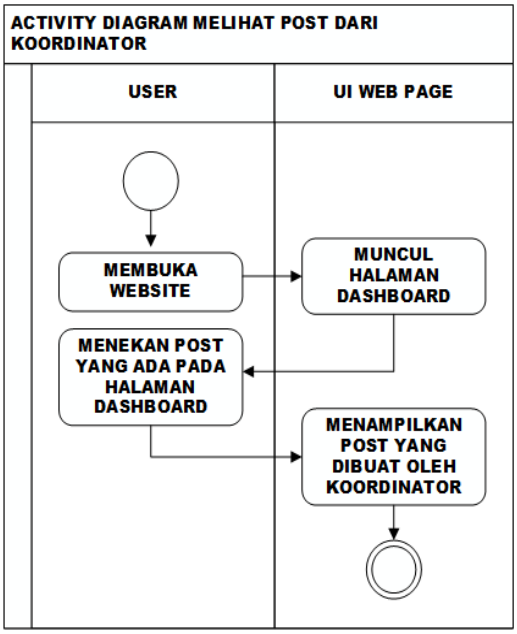


Gambar 6. *Use Case Diagram Sistem Administrasi TA*

DOI: http://dx.doi.org/10.22441/fifo.2022.v14i1.002

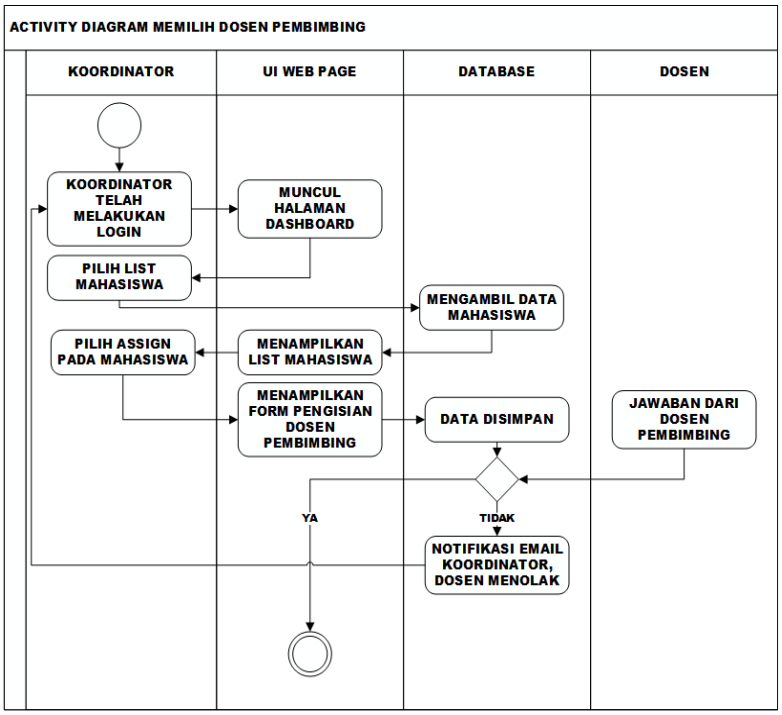
Activity Diagram

Activity diagram merupakan representasi perilaku sistem dan paling banyak dipergunakan dalam rangkaian diagram *Unified Modeling Language* (UML) sebagai standar bahasa dalam sistem permodelan dan perancangan perangkat lunak.Dalam *activity diagram*, maka sebuah *node* aktivitas dapat berupa sebuah *node* objek, sebuah *node* aksi, ataupun sebuah *node* kendali [14]. Aktivitas pertama pengguna pada sistem digitalisasi administrasi TA adalah validasi login untuk memasuki sistem. Sistem juga memfasilitasi perilaku aktivitas *signup* pengguna, apabila pengguna masih belum terdata dalam repositori yang telah disediakan. Aktivitas pengguna untuk melihat pesan notifikasi dari Koordinator TA digambarkan pada Gambar 7. Koordinator TA juga dapat melakukan posting informasi penting untuk seluruh mahasiswa melalui sistem ini.

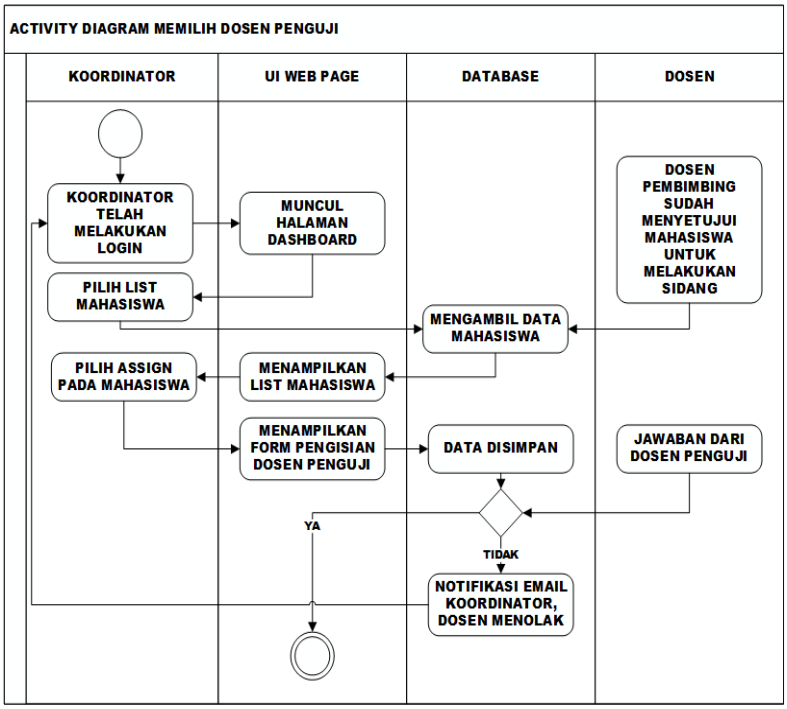


Gambar 7. *Activity Diagram* Post Koordinator

Seluruh dokumen mahasiswa akan disimpan dalam folder masing-masing sesuai nama mahasiswa peserta TA. Dosen penguji TA, dosen pembimbing TA, dan juga koordinator TA dapat mengakses langsung dokumen yang diperlukan pada setiap folder mahasiswa tersebut. Komunikasi dan mekanisme proses pemilihan dosen penguji dan dosen pembimbing dapat dilakukan melalui sistem seperti yang digambarkan pada Gambar 8 dan Gambar 9. Dosen penguji dan pembimbing juga dapat melakukan input penilaian terkait TA melalui sistem.



Gambar 8. *Activity Diagram* Pemilihan Dosen Pembimbing



Gambar 9. *Activity Diagram* Pemilihan Dosen Penguji

Schema

MongoDB merupakan salah satu manajemen sistem basis data non relational yang bersifat *open-source* menggunakan konsep dinamik *schema*, penyimpanan datanya dalam struktur JSON (sebuah list dari pasangan *key-value)* berbentuk dokumen yang beragam. MongoDB digunakan pada lingkungan JavaScript sisi *server* berfungsi menampung data melalui fungsionalitas dari NodeJS. Dipandang dari sisi performansi maka MongoDB lebih unggul dalam proses query dibanding MySQL [15]. Mongoose merupakan *Object Document Model* (ODM) yang memetakan objek pada basis data MongoDB dengan model JavaScriptnya, sehingga lingkungan

permodelannya cukup nyaman untuk penggunaan MongoDB[16]. *Schema* user, dosen dan koordinator pada Gambar 10 berfungsi untuk menyimpan data profil dari setiap *user*, dosen dan koordinator yang *login* maupun *sign up*. Nantinya data tersebut akan masuk ke dalam basis data MongoDB dalam bentuk *Schema*. Setiap melakukan *login* akan ada status yang akan menentukan *user* akan pindah ke *dashboard* yang dituju.

```
const mongoose = require("mongoose");
const uniqueValidator = require("mongoose-unique-validator");

const userSchema = mongoose.Schema({
  email: { type: String, required: true , unique: true},
  password: { type: String, required: true },
  nim: { type:String },
  status: { type: String },
  path: { type: String },
  krs: {type:String},
  judul:{type:String},
  dosen_pembimbing:{type:String},
  dosen_penguji:{type:String},
  dosen_penguji_2:{type:String},
  pending:{type:String}
});

userSchema.plugin(uniqueValidator);

module.exports = mongoose.model("user", userSchema);
```

Gambar 10. *Schema User*, Dosen, dan Koordinator

Gambar11 merupakan *Schema* untuk dosen dalam meng*upload* penilaiannya untuk mahasiswa yang dibimbing maupun diuji. Hasil penilaian dari dosen akan masuk ke *database*, maka koordinator akan mengetahui dan mengecek nilai mahasiswa beserta keterangan untuk membuktikan kejelasan nilai tersebut.

```
const mongoose = require("mongoose");

const nilai = mongoose.Schema({
  mhsEmail :{type:String},
  dosenEmail :{type:String},
  nilai :{type:String},
  keterangan : {type:String}
});

module.exports = mongoose.model("nilai", nilai);
```

Gambar 11. *Schema* Penilaian Dosen

Meskipun Koordinator TA memiliki wewenang untuk menentukan dosen pembimbing TA dan dosen penguji bagi mahasiswa peserta TA, tetapi koordinator juga tidak sepenuhnya memiliki wewenang memaksa dosen tersebut untuk menyetujuinya. Olehkarena itu, dosen difasilitasi rubrik '*answer'* dalam *Schema* terkait jawaban (menolak atau menerima) penujukkan dosen oleh Koordinator, seperti terlihat pada Gambar 12.

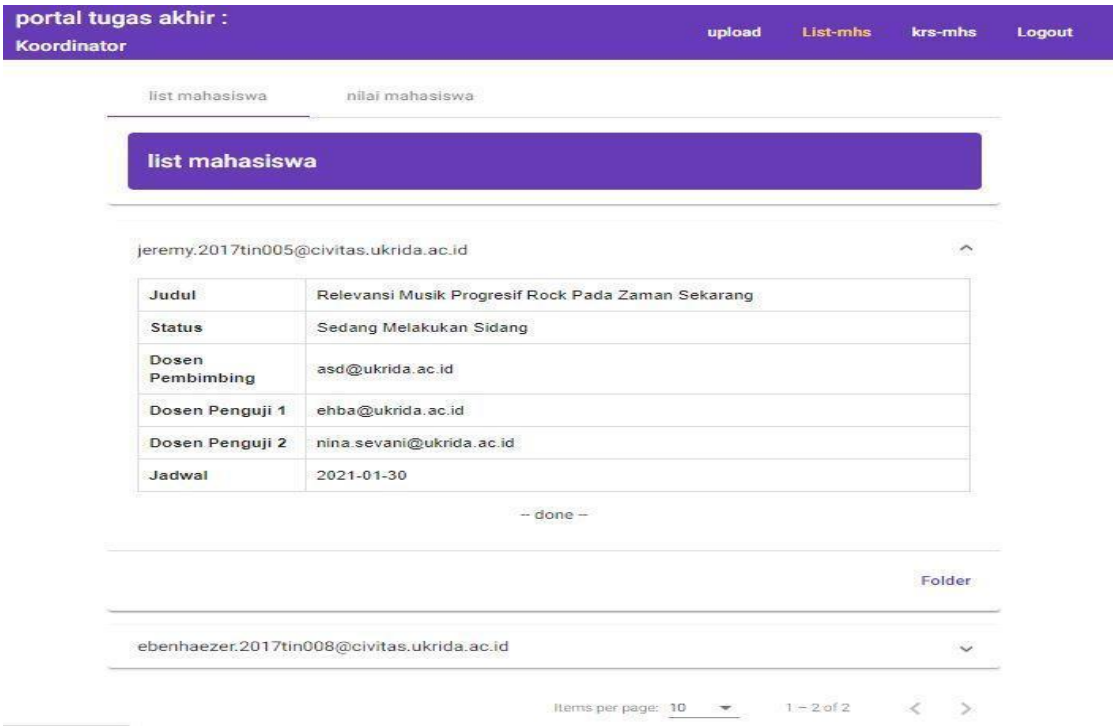
```
const mongoose = require("mongoose");

const assign = mongoose.Schema({
  mhsEmail :{type:String},
  dosenId :{type:String},
  status :{type:String},
  answer : {type:String},
  judul: {type:String}
});

module.exports = mongoose.model("Assign", assign);
```

Gambar 12. *Schema* Persetujuan Dosen Pembimbing dan Penguji

Implementasi sistem dilakukan menggunakan JavaScript yang mendasari pemrograman *MEAN Stack*. *MEAN Stack* akan memproses setiap lapisan, dimulai dari pengaksesan basis data, pembuatan RESTful API, pengelolaan routing, hingga implementasi *Single Page Application* (SPA). NodeJS memiliki komponen yang digunakan untk mengakses basis data Mongoose. Gambar 13 merupakan contoh tampilan pada halaman Koordinator, dimana terdapat data mahasiswa beserta nama pembimbing dan penguji yang sudah disetujui oleh dosen yang bersangkutan.

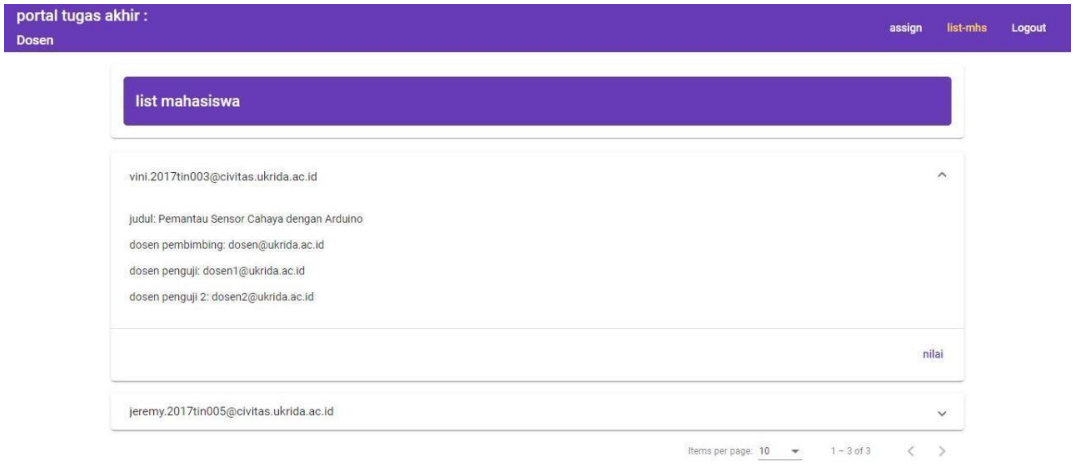


Gambar 13. Implementasi Halaman List Mahasiswa-Dosen Koordinator

Gambar 14. Notifikasi Penolakan Dosen

Dosen dapat memberikan penolakan atas bimbingan dan pengujian melalui aplikasi, dan secara otomatis notifikasi penolakan tersebut akan masuk dalam email yang dikelola oleh koordinator seperti pada Gambar14. Selanjutnya koordinator TA dapat mengelola kembali penentuan dosen pembimbing atau dosen penguji pada mahasiswa yang bersangkutan. Dosen juga dapat melihat list mahasiswa bimbingan seperti pada Gambar15.

DOI: http://dx.doi.org/10.22441/fifo.2022.v14i1.002



Gambar 15. Implementasi halaman list mahasiswa-dosen

Seluruh implementasi ini dibagi menjadi beberapa blok iterasi, berdasarkan *use case diagram* pada Gambar 6. Iterasi pertama adalah proses yang dilakukan oleh *actor* mahasiswa, yang meliputi proses *login/sign up*, melihat post dari koordinator, dan mengumpulkan berkas melalui web. *Login/Signup* harus dilakukan menggunakan email institusi. Saat berhasil *login*, maka mahasiswa akan dapat melihat berita yang diposting oleh koordinator TA. Mahasiswa juga baru dapat mengumpulkan berkas pada saat sudah berhasil login. Ketiga proses ini kemudian akan diuji menggunakan *blackbox*. Hasil pengujian akan dianalis dan menjadi dasar perancangan pada blok iterasi berikutnya, yaitu blok iterasi untuk *actor* koordinator TA. Apabila ada hasil implementasi pada blok pertama yang tidak sesuai, maka tidak akan dilanjutkan untuk blok iterasi kedua. Pada blok iterasi kedua ini, akan dirancang proses-proses untuk *actor* koordinator TA yang selaras dengan hasil pengujian pada blok iterasi sebelumnya. Seluruh proses pada blok iterasi kedua juga akan diuji sebelum masuk ke blok iterasi ketiga. Blok iterasi ketiga merupakan blok terakhir dalam sistem yang dikembangkan. Keterkaitan antara blok iterasi kedua dengan blok iterasi ketiga sama dengan yang dilakukan dari blok pertama ke blok kedua.

Setelah seluruh proses implementasi selesai dilakukan, sistem keseluruhan juga diuji menggunakan *blackbox testing* yang dilakukan oleh tim pengembang. Pengujian ini diperlukan untuk melihat apakah sistem yang dikembangkan sudah siap dipakai, serta sesuai dengan tujuan pembuatannya, dan hal ini dapat dilakukan dengan menyiapkan sejumlah *test case* untuk melihat validitas hasilnya [17]. Oleh karena itu, pengujian pada sistem ini difokuskan pada input dan output sistem, serta tidak menekankan pada proses kontrol yang dilakukan dalam sistem [10]. Tabel 1 berikut merupakan hasil pengujian untuk beberapa *test case*.

Tabel 1. Hasil Uji *Blackbox*

Test Case	Deskripsi Pengujian	Hasil yang Diharapkan	Hasil	Kesimpulan
Halaman Login	Mengetikkan email, dan password tidak diisi atau kosong kemudian klik tombol Login	Sistem akan menolak dan menampilkan pesan “Password belum diisi”	Sesuai harapan	Berhasil
Halaman Post Koordinator	Mengetikkan Title , content dan tempat upload file diisi kemudian klik tombol submit	Sistem akan menyimpan data pindah ke halaman utama	Sesuai harapan	Berhasil
Halaman Pilih Dosen Pembimbing	Memilih dosen kemudian klik tombol submit	Sistem akan menyimpan data pindah ke halaman utama	Sesuai harapan	Berhasil
Halaman Mahasiswa Mengumpulkan Berkas	Mengumpulkan berkas file yang akan diuji dan dinilai disebuah kolom ‘choose file’	File akan otomatis masuk ke sebuah folder dan terdistribusi ke sistem koordinator	Sesuai harapan	Berhasil
Halaman Pengisian Nilai	Mengisi Form hasil penilaian Mahasiswa yang diuji	Dosen dapat mengisi penilaian di form tersebut, lalu nilai bisa masuk ke sistem Koordinator	Sesuai harapan	Berhasil

IV. Kesimpulan

Pendigitalisasian administrasi TA dengan pengembangan web berteknologi *MEAN (MongoDB, Express, Angular, dan NodeJS) stack* dengan menggunakan JavaScript pada sisi *backend* dan *frontend* mengakomodir penyederhanaan dan transparansi koordinasi bertingkat antar mahasiswa peserta TA, dosen pembimbing TA, dosen penguji TA dan koordinator TA. Dampak lain dari pendigitalisasian administrasi TA, maka keputusan yang tepat dapat diambil mengingat terpusatnya media penyimpanan data dan teroganisirnya keseragaman format dokumen.

Referensi

[1] Muhyadi, "Kajian Ilmu Administrasi," *Efisiensi, Kaji. Ilmu Adm.*, vol. XIV, no. 1, pp. 98–109, 2016.
[2] R. Jain and U. Suman, "A Project Management Framework for Global Software Development," *ACM SIGSOFT Softw. Eng. Notes*, vol. 43, no. 1, pp. 1–10, 2018, doi: 10.1145/3178315.3178329.
[3] R. Datta and H. U. Sharif, "An integrated approach to software engineering," *Int. J. Adv. Electron. Comput. Sci.*, vol. 7, no. 6, pp. 30–35, 2020, doi: 10.5860/choice.29-2748.
[4] S. Nugroho, S. H. Waluyo, and L. Hakim, "Comparative analysis of software development methods between parallel, V-shaped and iterative," *arXiv*, vol. 169, no. 11, pp. 7–11, 2017, doi: 10.5120/ijca2017914605.
[5] M. B. Legowo, B. Indiarto, and D. Prayitno, "Implementation of Scrum Work Framework in the Development of Quality Assurance Information System," *J. Penelit. Pos dan Inform.*, vol. 9, no. 2, p. 125, 2019, doi: 10.17933/jppi.2019.090204.
[6] M. Hron and N. Obwegeser, "Scrum in Practice: an Overview of Scrum Adaptations," *Proc. 51st Hawaii Int. Conf. Syst. Sci.*, pp. 5445–5454, 2018, doi: 10.24251/hicss.2018.679.
[7] N. Nirgudkar and P. Singh, "The MEAN Stack," *Int. Res. J. Eng. Technol.*, pp. 2395–56, 2017.
[8] P. L. Lokapitasari Belluano, "Pengembangan Single Page Application Pada Sistem Informasi Akademik," *Ilk. J. Ilm.*, vol. 10, no. 1, pp. 38–43, 2018, doi: 10.33096/ilkom.v10i1.204.38-43.
[9] V. Gavrilă, L. Băjenaru, and C. Dobre, "Modern single page application architecture: A case study," *Stud. Informatics Control*, vol. 28, no. 2, pp. 231–238, 2019, doi: 10.24846/v28i2y201911.
[10] T. Snadhika Jaya, "Pengujian Aplikasi dengan Metode Blackbox Testing Boundary Value Analysis (Studi Kasus: Kantor Digital Politeknik Negeri Lampung)," *J. Inform. J. Pengemb. IT*, vol. 03, no. 02, pp. 45–48, 2018.
[11] F. C. Ningrum, D. Suherman, S. Aryanti, H. A. Prasetya, and A. Saifudin, "Pengujian Black Box pada Aplikasi Perpustakaan Menggunakan Teknik Equivalence Partitions," *J. Inform. Univ. Pamulang*, vol. 4, no. 4, pp. 125–130, 2019, doi: 10.32493/jtsi.v3i3.5343.
[12] P. Ammann and J. Offutt, *Introduction to Software Testing*. Cambridge University Press, 2016.
[13] M. El-Attar, "Evaluating and empirically improving the visual syntax of use case diagrams," *J. Syst. Softw.*, vol. 156, pp. 136–163, 2019, doi: 10.1016/j.jss.2019.06.096.
[14] L. Lima, A. Tavares, and S. C. Nogueira, "A framework for verifying deadlock and nondeterminism in UML activity diagrams based on CSP," *Sci. Comput. Program.*, vol. 197, p. 102497, 2020, doi: 10.1016/j.scico.2020.102497.
[15] D. Damodaran B, S. Salim, and S. M. Vargese, "Performance Evaluation of MySQL and MongoDB Databases," *Int. J. Cybern. Informatics*, vol. 5, no. 2, pp. 387–394, 2016, doi: 10.5121/ijci.2016.5241.
[16] D. Bakwa D., E. Edim A., and O. Dantala O., "Simplifying Web Application Development Using - Mean Stack Technologies," *Int. J. Latest Res. Eng. Technol.*, vol. 04, no. 01, pp. 62–76, 2018.
[17] A. N. A. Thohari and A. E. Amalia, "Implementasi Test Driven Development Dalam Pengembangan Aplikasi Berbasis Web," *SITECH J. Sist. Inf. dan Teknol.*, vol. 1, no. 1, pp. 1–10, 2018, doi: 10.24176/sitech.v1i1.2255.