Sistem Presensi Online Menggunakan Arsitektur Pengembangan Perangkat Lunak Model-View-Viewmodel

Adha Setiawan Wiyana[1*], M. Ihsan Alfani Putera[2], Sri Rahayu Natasia[3]

[1,2,3]Program Studi Sistem Informasi, Institut Teknologi Kalimantan, Balikpapan, Kalimantan Timur
Email: [1*]adhasetiawanwiyana@gmail.com, [2]ihsanalfani@lecturer.itk.ac.id, [3]natasia.ayu@lecturer.itk.ac.id

(Naskah masuk: 1 Agu 2021, direvisi: 18 Okt 2021, 29 Okt 2021, diterima: 2 Nov 2021)

Abstrak

PT. Lintasmaya Network Samarinda adalah perusahaan yang bergerak di bidang jasa dengan memberikan layanan *IT Support* kepada perusahaan lain. Sebagai perusahaan yang bergerak di bidang jasa, PT. Lintasmaya Network Samarinda perlu memastikan untuk dapat memberikan layanan terbaik kepada konsumen dengan memperhatikan kinerja karyawan. Agar hal tersebut dapat tercapai, maka solusi yang dipilih oleh PT. Lintamaya Network Samarinda adalah membangun sistem presensi berbasis aplikasi *mobile*. Hanya saja karena PT. Lintasmaya Network Samarinda tidak memiliki kemampuan dan pengalaman dalam mengembangkan sebuah aplikasi *mobile*, sistem presensi tersebut tidak dapat berjalan dengan baik. Berangkat dari masalah tersebut, maka diadakan penelitian pengembangan sistem presensi berbasis aplikasi *mobile* dengan menggunakan metode pengembangan perangkat lunak *Personal Extreme Programming* (PXP) dan arsitektur *Model-View-Viewodel* (MVVM). Metode pengembangan perangkat lunak PXP terdiri dari tujuh proses yaitu *requirement*, *planning*, *iteration initialization*, *design*, *implementation*, *system testing*, dan *retrospective*. Sedangkan aristektur *MVVM* adalah penulisan kode yang dibagi menjadi tiga bagian terdiri dari *view*, *viewmodel*, dan *model*. Adapun hal lain yang menjadi perhatian dalam penelitian ini adalah penerapan IMEI untuk mengenali perangkat, GPS untuk mendapatkan lokasi, dan JWT *Authentication* untuk keamanan fitur. Melalui penerapan tersebut, sistem dapat memberikan hal positif seperti data yang akurat dan penggunaan fitur yang terbatas hanya untuk pihak perusahaan.

Kata Kunci: *Android*, MVVM, PXP, Sistem Presensi.

Online Presence System Development Using Model-View-Viewmodel Software Development Architecture

Abstract

PT. Lintasmaya Network Samarinda is a company engaged in the service sector by providing IT Support services to other companies. As a company engaged in service, PT. Lintasmaya Network Samarinda needs to ensure that it can provide the best service to consumers by paying attention to employee performance. To achive that aim, the solution chosen by PT. Lintamaya Network Samarinda is building a mobile application-based presence system. It's just because PT. Lintasmaya Network Samarinda does not have the ability and experience in developing a mobile application, the presence system cannot run properly. Based on this problem, a research was conducted on the development of a mobile application-based presence system using the Personal Extreme Programming (PXP) software development method and the Model - View - Viewodel (MVVM) architecture. PXP software development method consists of seven processes, namely requirements, planning, iteration initialization, design, implementation, system testing, and retrospective. While the MVVM architecture is writing code which is divided into three parts consisting of view, viewmodel, and model. Other concerns in this research are the application of IMEI to identify the device, GPS to get location, and JWT Authentication for security features. Through that implementation, the system is now able to provide positive things such as accurate dat and, limited use of features only for the company.

Keywords: *Android, MVVM, PXP, Presence System.*

I. PENDAHULUAN

Perkembangan teknologi sudah menjadi hal yang sangat umum dan mudah untuk diamati. Perkembangan ini terjadi diberbagai macam bidang dan berbagai macam aspek, selaras dengan jalannya perkembangan ilmu pengetahuan. Hal ini dapat dibuktikan dengan bukti sejarah pada tahun 1750 hingga 1930 dimana mesin uap dan kereta api menjadi revolusi industri pada saat itu. Lalu dilanjutkan dengan 1870 hingga 1900 dimana listrik, alat komunikasi serta kimia dan minyak menjadi hal baru yang digemari oleh masyarakat dan industri dunia. Hingga akhirnya masuk pada tahun 1960 hingga sekarang dimana *internet* sudah menjadi kebutuhan sehari-hari untuk masyarakat dunia [1].

Pada tahun 2021 perkembangan teknologi kini sudah menjadi lebih pesat dengan hadirnya revolusi industri 4.0. Mengutip dari penelitian yang sudah dilakukan oleh Venti Eka pada tahun 2018, Kementrian Perindustrian sudah memiliki rencana agar Indonesia bisa mendapatkan untung dari revolusi industri 4.0. Adapun langkah tersebut dimulai dari mendorong sumber daya manusia lebih berkompeten dalam bidang *internet of things* (IoT) hingga memfasilitasi pengembangan *start up* di Indonesia [1].

Antusias dari revolusi industri 4.0 tidak hanya terjadi pada sisi pemerintah saja. Pada dunia industri sudah banyak perusahaan/instansi/lembaga/organisasi yang mau melalukan investasi pada bidang teknologi, khususnya pada sisi perkembangan perangkat lunak. Salah satu contohnya penelitian yang dilakukan oleh Panji Rachmat untuk membuat sistem presensi berbasis *Android*, dimana pada hasil penelitian tersebut implementasi sistem presensi berbasis *Android* pada Universitas Islam Riau memberikan dampak baik karena sistem dapat mempermudah proses pendataan kehadiran [2].

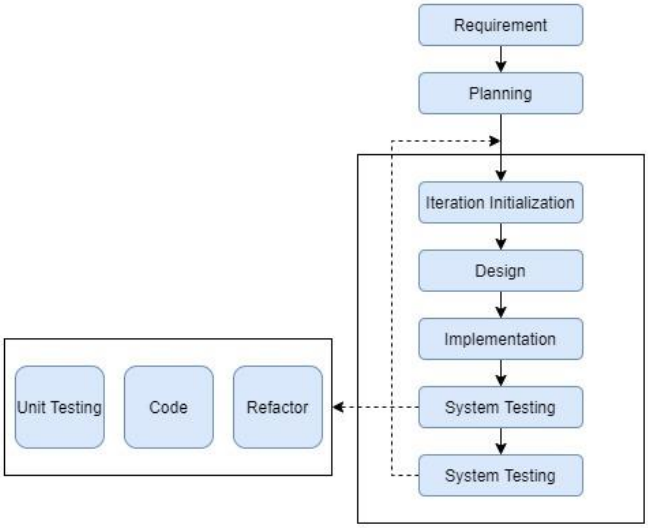
Hasil dari penelitian di Universitas Islam Riau tersebut tidak jauh berbeda dengan penelitian di beberapa tempat. Jika dikelompokan ke dalam dua *device*, penelitian sistem presensi pada *device Android* pernah dilakukan di dua perusahaan. Pertama oleh Ardian Permana untuk perusahaan PT. Pelabuhan Indonesia III [3]. Kedua dilakukan oleh Arafat Febriandirza pada PT. Duta Hidayah [4]. Sedangkan pada *web*, penelitian sistem presensi pernah dilakukan oleh Subiantoro pada kantor Kecamatan Purwodadi [5]. Penelitian-penelitian tersebut mendapatkan hasil yang baik di mana sistem kehadiran dapat mempercepat proses pendataan kehadiran masuk dan pulang karyawan. Selain itu adapun keuntungan lain yang didapat seperti akses informasi yang mudah, resiko kesalahan yang kecil, dan lainnya [5].

Berangkat dari kondisi yang telah dijelaskan di atas, maka hal serupa telah dilakukan oleh perusahaan Lintasmaya Network Samarinda pada sistem kehadiran mereka. PT. Lintasmaya Network sendiri adalah perusahaan yang bergerak di bidang jasa dengan memberikan layanan *IT Support* kepada perusahaan lain. Perusahaan ini berada di kota Samarinda, provinsi Kalimantan Timur. Karena PT. Lintasmaya Network Samarinda bergerak pada bidang jasa, maka kinerja dari sumber daya manusia sudah menjadi aspek yang perlu mendapat perhatian khusus dari perusahaan tersebut [6].

Bergeraknya Lintasmaya Network pada bidang jasa dan hadirnya tema penelitian untuk sistem presensi karyawan, membuat PT. Lintasmaya Network terdorong untuk dapat mengembangkan sistem presensi dengan dua fokus utama. Fokus pertama adalah keamanan dan fokus kedua adalah *maintainability*, dimana sistem mudah untuk dirawat dan dikembangkan oleh perusahaan. Hadirnya kedua fokus ini akan menjadi pembeda dari penelitian yang sudah ada, sekaligus kunci agar PT. Lintasmaya Network dapat mengawasi kinerja karyawan sehingga dapat berkontribusi penuh kepada perusahaan [6].

Adapun implementasi teknologi dan metode yang digunakan agar kedua fokus tercapai adalah *PXP* yang merupakan bentuk lain *Extreme Programming* (*XP*) dimana *PXP* dibuat khusus untuk pengembangan sistem oleh satu orang [7]. Arsitektur *MVVM* yang merupakan salah *model* desain pengembangan perangkat lunak yang membuat *user interface* terpisah dengan *back end* dari sistem [8]. Sedangkan pada fokus keamanan implementasi akan berfokus pada penggunaan IMEI perangkat, GPS, dan JWT *Authentication*.

II. METODOLOGI PENELITIAN



Gambar 1. Alur Penelitian dari Sistem Presensi PT. Lintasmaya Network

Implementasi metode pengembangan perangkat lunak PXP didasarkan pada penelitian tahun 2018 yang telah dilakukan oleh Gita Indah. Hasil dari implementasi metode PXP pada penelitian teresbut berjalan dengan baik, karena proses metode *PXP* memberikan keuntungan berupa mudahnya proses pengembangan sistem. Ketika terjadi perubahan terhadap proses *requirement*, peneliti dapat cepat menyesuaikan perubahan tersebut pada tahap implementasi, sehingga peneliti dapat dengan cepat menghadirkan sistem informasi bagi perpustakaan [7].

Melalui penelitian yang sudah pernah dilakukan tersebut, maka implementasi PXP pada sistem presensi dianggap ideal. Metode PXP adalah metode turunan dari *Extreme*

Progromming (XP) yang mana XP sendiri adalah salah satu model dari metode *Agile*. Perbedaan metode PXP dan XP ada pada penyesuaian cara kerja, yang mana metode PXP dibuat khusus untuk satu orang. Melalui penyesuaian tersebut maka implementasi metode PXP terdiri dari tujuh proses utama seperti terlihat pada Gambar 1. Proses pertama akan dimulai dari *requirement* lalu dilanjutkan dengan *planning*, *design*, *implementation*, *system testing* hingga proses terakhir yaitu *retrospective* [7].

A. *Requirement*

Tahapan pertama dari PXP pada sistem kehadiran adalah tahap *requirement.* Tahap ini dikerjakaan dengan tujuan untuk mendapatkan kebutuhan dasar dari sistem yang dikembangkan ke depannya, mulai dari apa saja fitur yang dibuat dan bagaimana detail dari fitur yang ada pada sistem tersebut. Selain mendapatkan semua kebutuhan dasar seperti yang telah disebutkan sebelumnya, tujuan lain dari tahapan ini adalah menilai apa saja sekiranya masalah yang bisa dihadapi, berapa lama waktu yang dibutuhkan, serta alat bantu saja yang diperlukan untuk membuat sistem tersebut. Tahapan *requirement* dikerjakan dengan menggunakan dua cara. Pertama adalah sesi wawancara kepada mitra dan kedua, studi literatur untuk melihat detail dari kebutuhan sistem [7].

B. *Planning*

Tahapan kedua adalah tahapan *planning.* Tahapan ini dikerjakan dengan memperhatikan hasil dari sesi wawancara pada tahap *requirement.* Selain dari memperhatikan hasil wawancara, proses lain yang dapat dikerjakan adalah memperhatikan bagaimana sistem yang ada sekarang sudah berjalan. Melalui dua proses tersebut maka hasil dari tahap *planning* adalah rancangan sistem baru pada tiap fitur dengan memperhatikan fokus-fokus yang diinginkan mitra. Apabila rancangan sudah sesuai dengan fokus yang disebutkan sebelumnya maka proses ini dapat dilanjutkan menuju tahapan *Iteration Initialization* dan begitu pula sebaliknya. Pada penelitian dari sistem presensi PT. Lintasmaya Network, hasil dari tahap *planning* dibuat dalam bentuk *flowchart* agar mudah dipahami. Melalui proses inilah proses pengembangan menjadi ideal karena proses pengembangan didokumentasikan sehingga bisa dijadikan pembelajaran ke depannya [7].

C. *Iteration Initialization*

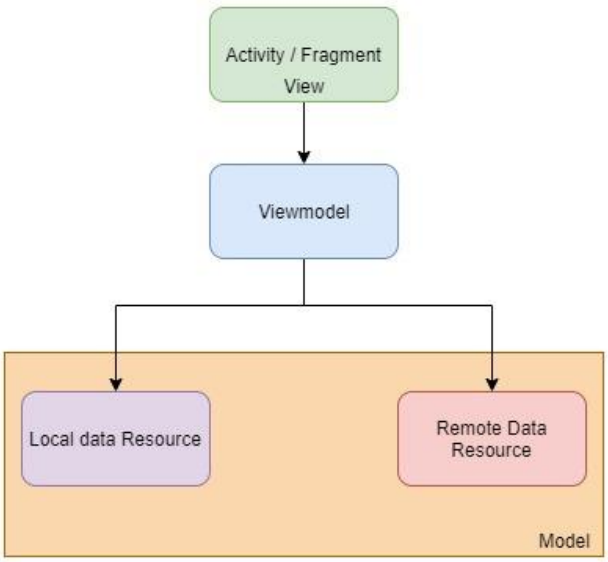
Tahap ketiga PXP pada sistem presensi adalah *Iteration Initialization.* Tahap ini dikerjakan dengan tujuan untuk membuat detail dari jalan proses pengembangan tiap iterasi. Selain membuat detail pengerjaan, tahapan ketiga ini juga merupakan tahapan untuk memastikan berapa lama tiap iterasi berjalan pada masing-masing proses. Hal yang perlu diperhatikan pada tahapan ini di antaranya adalah prioritas, tingkat kesulitan, dan alur sistem. Berdasarkan dari penjelasan di atas maka hasil dari implementasi metode *iteration initialization* dibuat dalam bentuk tabel dengan detail kegiatan terurut dilengkapi dengan estimasi waktu pengerjaan [7].

D. *Design*

Tahap keempat PXP pada sistem kehadiran adalah *design.* Hasil dari tahap ini adalah sebuah gambaran yang akan dilihat oleh *user*. Tahap ini dikerjakan dengan menggunakan alat bantu/*tool* seperti *Adobe* XD, *Figma*, dan *Balsamiq*. Pada penelitian ini, proses *design* dikerjakan dengan menggunakan figma, karena pemakaian yang mudah. Selain dari memberikan gambaran tampilan kepada *user*, tahap *design* memberikan keuntungan lain seperti membantu peneliti melihat data apa saja yang akan ditampilkan. Serta membantu proses implementasi kode berjalan lebih cepat karena *design* yang dibutuhkan sudah dikerjakan pada proses terpisah [7].

E. *Implementation*

Tahap kelima pada metode pengembangan PXP adalah *implementation.* Tahap ini adalah proses pengembangan sistem dengan menyesuaikan rancangan yang sudah dibuat [7]. Pada tahap implementasim peneliti menggunakan dua alat bantu yaitu bahasa pemrograman *Kotlin* dan *IDE Android Studio*. Agar proses implementasi berjalan maksimal, maka pada proses inilah peneliti menerapkan arsitektur MVVM dengan memperhatikan fokus keamanan dan *maintainability*.



Gambar 2. *Design Pattern Model-View-Viewmodel*

Sama seperti implementasi PXP, pemilihan implementasi arsitektur MVVM pada sistem kehadiran didasarkan dari penelitian lain. Implementasi MVVM sudah pernah dilakukan untuk membangun sistem yang membuat diagnosis kondisi pendengaran seseorang. Implementasi tersebut dilakukan pada tahun 2019 oleh Sheikh Waseem. Hasil dari implementasi tersebut menyatakan implementasi MVVM bisa menjadi solusi untuk pengembangan sistem dalam jangka waktu panjang karena penulisan kode yang terbagi menjadi tiga bagian sehingga proses pengembangan menjadi lebih mudah [8].

Berdasarkan Gambar 2, implementasi MVVM dibagi menjadi tiga bagian. Bagian pertama adalah model yang bertanggung jawab untuk menjalankan API dan mengatur

kesediaan data pada *repository*. Bagian kedua adalah *viewmodel* yang bertanggung jawab untuk memperhatikan ketersediaan data dan memberikan respon dari API. Bagian ketiga adalah *view* yang bertanggung jawab untuk mengatur tampilan yang akan dilihat oleh *user*. Melalui penulisan kode secara terpisah seperti penejalasan yang telah disampaikan, membuat sistem menjadi lebih mudah untuk dikembangkan dan dirawat jika terjadi hal yang tidak diinginkan[8].

Adapun proses implementasi dilanjutkan dengan menerapkan tiga metode keamanan yang didasarkan pada penelitian lain. Metode pertama adalah penggunaan IMEI untuk mengetahui *device* yang digunakan, sehingga ketika karyawan ingin mengganti *device*, karyawan tersebut harus melapor kepada perusahaan. Penggunaan IMEI pada sistem kehadiran dianggap ideal, karena kombinasi angka yang acak serta beragam dan dapat digabungkan dengan metode keamanan lain, yang dilakukan pada penelitian 2018 oleh Pavan Sai [9].

Metode selanjutnya adalah metode untuk melacak lokasi karyawan dengan menggunakan GPS. Metode ini dianggap efektif karena sudah diterapkan pada penelitian lain seperti penelitian *AbsenLoc* pada tahun 2021 oleh Jemy Agung. Pada penelitian tersebut, hasil implementasi pada sistem presensi dianggap efektif dan sudah dapat memberikan manfaat yaitu proses pendataan kehadiran yang singkat dan mudah [10].

Metode keamanan terakhir pada sistem presensi PT. Lintasmaya Network adalah JWT *Authentication*. Metode ini bekerja dengan memberikan sebuah token kepada masing-masing *user*, sehingga semua fitur pada sistem kehadiran benar-benar hanya digunakan oleh pihak yang mimiliki kepentingan. Implementasi dari JWT *Authentication* dianggap cukup efektif karena sudah terbukti dapat menuntaskan masalah interoperabilitas, sehingga sistem dapat bertukar informasi satu sama lain [11].

F. *System Testing*

Bagian keenam dari implementasi PXP pada sistem kehadiran adalah *system testing*. Proses uji sistem dilakukan dengan tujuan untuk melihat bagaimana sistem berjalan dengan skenario kejadian tertentu. Proses ini akan berjalan dengan menggunakan metode uji sistem *black box*. Adapun implementasi metode uji sistem *black box* dikerjakan pada tiap iterasi agar hasil yang didapatkan menjadi lebih teliti [7].

G. *Restrospective*

Tahap terakhir pada PXP adalah tahap *restrospective*. Tahap ini bertujuan untuk melihat kesesuaian proses implementasi dari dengan rancangan yang sudah dibuat. Tahap ini dilakukan pada di semua iterasi. Apabila sudah sesuai maka proses iterasi tersebut akan dianggap selesai dan peneliti dapat berlanjut mengerjakan iterasi pada fitur-fitur lain [7].

III. HASIL DAN PEMBAHASAN

Berikut adalah detail dari implementasi metode pengembangan PXP dan aristektur MVVM. Agar hasil dapat sesuai dengan fokus yang diinginkan mitra, maka hal lain yang harus diperhatikan adalah penggunaan metode *get* IMEI, *get location* & JWT *Authentication*.

A. *Requirement*

Berikut adalah hasil proses *requirement* dengan melakukan dua proses. Proses adalah wawancara kepada mitra untuk mendapatkan kebutuhan dasar dari sistem kehadiran. Proses kedua adalah studi literatur untuk mendapatkan detail dari tiap fitur yang ada pada sistem kehadiran.

Tabel 1. Daftar Tabel Fitur Kehadiran

Tabel Fitur			
No	**Kode Fitur**	**Nama Fitur**	**Deskripsi Fitur**
1	FU-01	*Log in / Sign in*	Sistem keamanan dengan memastikan bahwa hanya dapat digunakan oleh pihak tertentu.
2	FU-02	*Admin contact*	Menghubungi admin melalui *send email*, sebagai langkah untuk menyampaikan masalah jika akun *user* bermasalah.
3	FU-03	*Presence*	Pencatatan kehadiran tanggal, waktu dan kegiatan karyawan.
4	FU-04	*Absence*	Pencatatan waktu dan tanggal selesai bekerja.
5	FU-05	*Track record*	Menampilkan data detail kinerja karyawan dalam bentuk waktu, tanggal dan kegiatan.
6	FU-06	*Profile*	Menampilkan data diri *user*.
7	FU-07	*Password change*	Sistem keamanan dengan memastikan bahwa hanya dapat digunakan oleh pihak tertentu.

Berdasarkan Tabel 1, proses pengembangan sistem presensi memiliki tujuh buah fitur. Tiap fitur dikerjakan dalam iterasi terpisah. Setiap iterasi memiliki detail yang berbeda dengan iterasi lain. Jika pengerjaan iterasi sudah sesuai dengan yang diinginkan mitra maka proses pengembangan dapat dilanjutkan menuju iterasi lain. Jika yang terjadi sebaliknya maka iterasi akan disesuaikan dengan yang diinginkan mitra. Adapun detail dari tiap fitur dapat dilihat dari salah satu tabel pada iterasi fitur seperti berikut

.

Tabel 2. Detail Fitur *Log In* Pada Sistem Presensi

Kode Fitur	**Nama Fitur**	**Deskripsi Fitur**	**Detail Fitur**
FU-01	*Log in*	Sebuah fitur yang memungkinkan *user* untuk mengakses sebuah sistem dengan	*Input email, password* & IMEI
			Input IMEI dilakukan secara otomatis

	mengenali *user* melalui input identitas tertentu.	Akses data penggunaan dengan *email, password* & IMEI
		Generate token untuk menjadi akses *user* kepada fitur lain
		Menyimpan respon dari pencarian data *user*

Berdasarkan Tabel 2 rancangan pada fitur *log in* akan memiliki lima detail. Masing-masing detail dibuat dengan memperhatikan fokus yang sudah diinginkan mitra. Pada fitur *log in*, fokus yang diperhatikan adalah fokus keamanan. Fokus ini dapat diwujudkan dengan menambahkan masukan data secara otomatis dengan menggunakan *get* IMEI sehingga sistem dapat mengenali *device* apa yang digunakan oleh *user*. Keuntungan dari penggunaan metode *get* IMEI adalah bisa memberikan fokus lain, yaitu fokus akurasi data sehingga dapat meminimalkan kemungkinan masukan data yang diberikan salah.

Tabel 3. Detail Fitur Admin *Contact* Pada Sistem Presensi

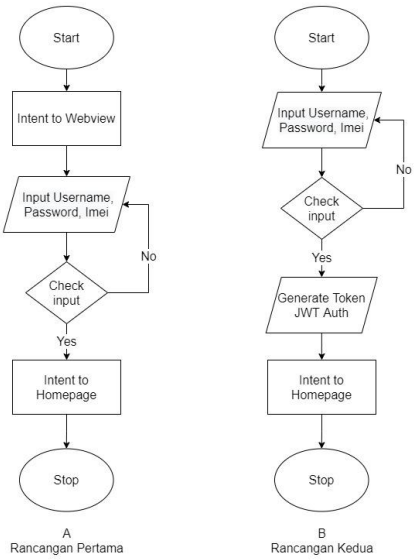
Kode Fitur	Nama Fitur	Deskripsi Fitur	Detail Fitur
FU-02	*Admin Contact*	Sebuah fitur yang dapat membantu *user* ketika terjadi kendala pada akun	Akses halaman pengiriman *email* dengan sebuah *button*

Berdasarkan Tabel 3 agar fokus *maintainability* dapat bisa menjadi lebih maksimal, maka peneliti dapat membuat sebuah fitur sederhana. Fitur tersebut nantinya akan berguna untuk menghubungkan *user* dan *admin* ketika terjadi masalah sehingga kedepannya mitra memiliki refrensi dan data apa saja yang harus diperbaiki. Selain itu juga memiliki referensi dan data bagaimana sistem bisa dikembangkan menjadi lebih baik berdasarkan dari masukan yang diberikan oleh *user*.

B. *Planning*

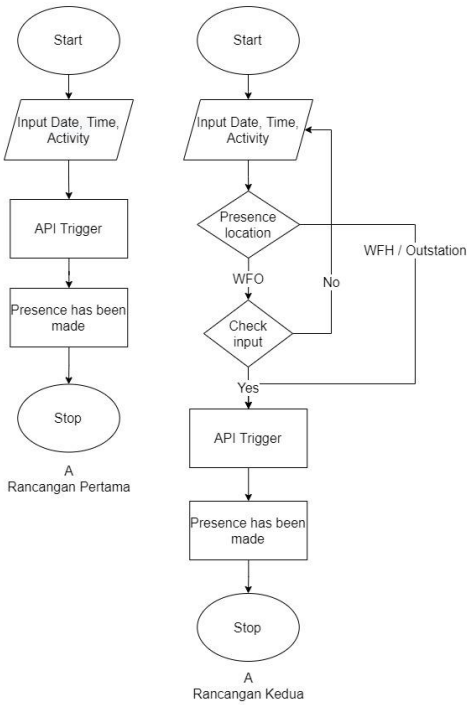
Proses selanjutnya pada implementasi PXP adalah proses *planning*. Proses ini dikerjakan dengan memperhatikan hasil dari proses *requirement*. Selain memperhatikan proses pertama, hal lain yang dapat menjadi acuan ketika mengerjakan proses *planning* adalah melihat dari sistem yang sudah ada serta memperhatikan metode yang sekiranya dapat memenuhi fokus pengembangan yang diinginkan mitra.

Berdasarkan Gambar 3 agar fokus keamanan dari fitur *log in* dapat berjalan lebih baik, maka pada bagian B proses *log in* perlu dibuat metode untuk mengamankan API yang ada pada sistem kehadiran. Metode pengamanan ini dilakukan dengan menggunakan JWT *Authentcation*. Hadirnya metode pengamanan ini akan memberikan keuntungan untuk bisa melakukan hal seperti pemisahan otoritas antar akun seperti *admin* dan *user* sehingga sistem benar-benar digunakan oleh pihak yang memiliki kepentingan saja.



Gambar 3. Rancangan Baru Fitur *Log In* Pada Sistem Presensi

Gambar 3 bagian A adalah *flowchart* dari fitur *log in* yang telah dimplementasikan dan berjalan sekarang. Pada fitur yang sedang berjalan sistem kehadiran menjadi lebih sulit untuk dikembangkan karena *model* tampilan yang menggunakan teknologi *webview* dan bukan tampilan *native* dari sistem. Kerugian lain dari rancangan ini juga terletak pada sisi keamanan, karena sistem hanya dapat mengenali *user* tanpa bisa mengenali *device* yang digunakan oleh *user*.



Gambar 4. Rencana Baru Fitur *Presence* Pada Sistem Presensi

Berdasarkan Gambar 4 bagian B metode yang perlu diperhatikan selama proses implementasi adalah metode *get location.* Metode ini memberikan keuntungan kepada sistem agar bisa memastikan lokasi keberadaan *user*. Selain dari pada itu keuntungan lain dari penggunaan metode *get location* adalah peneliti dapat membuat fungsi kecil yang bisa meningkatkan keamanan seperti perhitungan jarak dan membuat pilihan untuk presensi di kantor dan di luar kantor.

Pada Gambar 4 bagian A adalah rancangan dari fitur *presence* yang kini digunakan. Proses pada rancangan ini mimiliki banyak kekurangan. Dimulai dari tidak adanya metode keamanan seperti pencatatan lokasi serta tidak dilengkapi dengan metode untuk melakukan kehadiran dengan metode yang berbeda sehingga akurasi data yang didapatkan oleh mitra tidak detail.

C. *Iteration Initialization*

Proses selanjutnya dari implementasi *model* pengermbangan PXP adalah *iteration intialization.* Proses ini dikerjakan dengan menentukan detail kegiatan yang akan dilakukan pada masing-masing iterasi. Agar proses pengerjaan iterasi dapat berjalan lebih efektif, maka selain membuat detail dari apa saja yang dikerjakan, peneliti juga perlu membuat urutan setiap detail kerjaan lengkap dengan estimasi waktu pengerjaan iterasi.

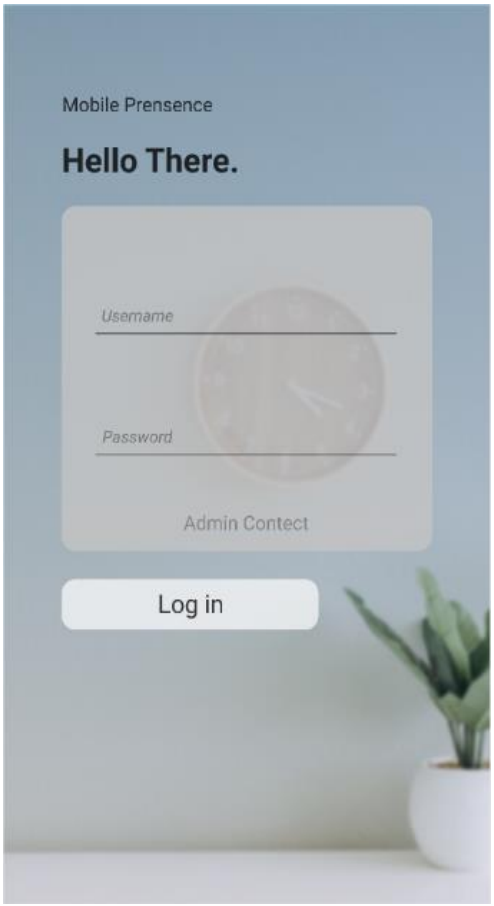
Tabel 4. Iterasi Fitur *Log In* Pada Sistem Presensi

Log in			
No	**Kode *Story***	**Deskripsi**	**Estimasi**
1	LO-01	Wawancara sistem terdahulu kepada mitra terkait sistem yang sudah ada	1 Hari
2	LO-02	Studi literatur	1 Hari
3	LO-03	Membuat model *As-Is* pada model *login* terbaru	1 Hari
4	LO-04	Konfirmasi rancangan model *As-Is* kepada mitra	1 Hari
5	LO-05	Membuat tampilan *Mock up*	2 Hari
6	LO-06	Membuat API dengan memperhatikan metode JWT *Authentication* dan pengujian API	6 Hari
7	LO-07	Implementasi *Login* dengan menggunakan metode *get* IMEI dan arsitektur MVVM	7 Hari
8	LO-08	Pengujian Sistem *Log in*	1 Hari

Berdasarkan Tabel 4, hasil tabel *iteration initialization* didapatkan dengan memperhatikan rancangan dari kebutuhan sistem pada proses pertama dan kedua. Karena setiap sistem memiliki kebutuhan dan detail yang berbeda maka proses implementasi pada tiap iterasi akan berbeda. Jika dalam proses perancangan, fitur tersebut tidak membutuhkan data dari basis data, maka proses pengerjaan dapat berjalan lebih singkat. Proses pengerjaan yang singkat ini terjadi karena peneliti tidak perlu membuat API untuk fitur tersebut sehingga fitur dapat berjalan tanpa koneksi internet nantinya.

D. *Design*

Proses keempat pada metode PXP adalah *design*. Seperti yang telah dijelaskan sebelumnya, proses ini dikerjakan dengan menggunakan sebuah alat bantu berupa *software*. Pada penelitian ini hasil proses *design* dikerjakan dengan menggunakan *software* bernama *Figma*. Berikut adalah salah satu hasil dari proses *design* pada iterasi *log in* dengan menggunakan *Figma*.



Gambar 5. Desain Baru Fitur *Log In* Pada Sistem Presensi

Berdasarkan Gambar 5, tampilan dari fitur *log in* memiliki desain yang bersifat minimalis. Desain dari fitur *log in* telah disesuaikan dengan rancangan yang sudah dibuat sebelumnya, sehingga masukan data secara manual dari *user* hanya ada pada *email* dan *password.* Sedangkan pada *imei* masukan data tidak diperlukan karena proses masukan data *imei* terjadi secara otomatis.

E. *Implementation*

Proses selanjutnya dari metode pengembangan PXP adalah *implementation.* Proses ini dikerjakan dengan menggunakan arstitektur MVVM. Selain dari arsitektur MVVM, pada proses implementasi proses penulisan kode juga berfokus pada implementasi IMEI, lokasi *user*, dan JWT *Authentication*. Proses implementasi dikerjakan dengan menggunakan bahasa pemrograman *Kotlin* dengan IDE

Android Studio. Berikut adalah salah satu hasil dari implementasi MVVM pada iterasi *log in.*

Tabel 5. Implementasi Fitur *Log In* Pada Sistem Presensi

Layer	Kelas	Hasil
Model	*ApiService*	b
	LoginObject	object LoginObject { data class LoginResponse(val data_user: DataUser, val massage: String, val success: Boolean, val token: String) data class DataUser(val id_user: Int, val username: String, val email: String, val name: String, val division: String, val address: String, val role: String, val picture: String)}

	LoginRepository	//API Trigger
		fun login
		(email:String,
		password:String,
		imei:BigInteger):Single<LoginObject.LoginResponse>{
		return
		apiService.login(email, password, imei)}

Viewmodel	*LoginViewmodel*	private val
		loginResponse:MutableLiveData<UiState<LoginObject.LoginResponse>>
		= MutableLiveData()
		
		fun
		getLoginResponse():LiveData<UiState<LoginObject.LoginResponse>>{return loginResponse}
		
		fun login(email:
		String, password:
		String, imei:
		BigInteger){
		//UiStatus
		
		loginResponse.postValue(UiState.Loading(true))
		//API Running
		
		loginRepository.login(
		email, password, imei)
		.observeOn(schedulerProvider.ui())
		.subscribeOn(scheduler
		Provider.io())
		.subscribe({
		loginResponse.postValue(UiState.Success(it))
		},{loginResponse.postValue(UiState.Error(it))
		}).addTo(compositeDisposable)}

view	*LoginActivity*	//observasi respon API pada repository
		viewmodel.getLoginResponse().observe(this,
		Observer {
		when (it) {
		
		is UiState.Loading ->loading.show()
		
		is UiState.Success ->{loading.dismiss()
		toast("Welcome " +
		it.data.data_user.name
		)
		
		viewmodel.loginStatus(
		)
		
		viewmodel.savePict(it.data.data_user.picture
		)
		
		viewmodel.saveToken(it.data.token)
		
		viewmodel.saveResponse(
		it.data.data_user.name
		,
		it.data.data_user.id_user,
		it.data.data_user.division,
		it.data.data_user.role
		)
		
		startActivity<BottomNavActivity>()

```
finish()
                }
                is
UiState.Error -> {
if (it.throwable is
HttpException) {
if
(it.throwable.code()
== 404) {

binding.btnLogin.snack
bar(
Utils.getErrorMessage(

it.throwable.response(
)?.errorBody()
))
loading.dismiss()

} else {

binding.btnLogin.snack
bar(it.throwable.messa
ge())

}
}

Timber.tag("Error tag
-> ").e("Kesalahan
pada -> " + it)
                }
            }
        })
```

Berdasarkan Tabel 5 implementasi dari fitur *log in* dikerjakan dengan arsitektur MVVM agar nantinya mitra bisa dengan mudah merawat dan mengembangkan sistem kehadiran sehingga fokus *maintainability* dapat tercapai. Implementasi kode terbagi menjadi tiga bagian. Bagian pertama *model*, dimana bagian ini betugas untuk menjalankan API dengan kode `fun login` (masukan data dari *user*). Setelah API berjalan dan sistem mendapatkan respon, maka respon tersebut disimpan ke dalam sebuah objek yang memiliki dua *data class* dalam bentuk kode `data class loginResponse` (struktur data dari basis data) dan `data class DataUser` (struktur data dari basis data). Untuk memastikan data tersebut tersimpan dan tersedia dengan baik, pada bagian model peneliti perlu menambahkan *repository* untuk mengatur ketersediaan data dalam sistem dalam bentuk fungsi `fun login` (parameter masukan data *user*).

Bagian kedua dari arsitektur *MVVM* adalah *viewmodel*, bagian ini memiliki fungsi untuk memperhatikan ketersediaan data dari model. Pertama implementasi dari *viewmodel* dibuat dengan membuat sebuah variabel seperti `private val loginResponse : MutableLiveData<UiState <LoginObject.LoginResponse>> = MutableLive Data()` dengan tujuan untuk menampung ketersediaan data dari *repository*. Lalu dilanjutkan dengan membuat sebuah fungsi `fun getLoginResponse() : LiveData <UiState<LoginObject.LoginResponse>> = login Response` dengan tujuan untuk melihat ketersediaan data dari variabel `login`Response. Terakhir adalah membuat satu buah fungsi dengan kode `fun login` (parameter masukan data dari *user*){respon dari sistem terhadap ketersediaan data} dengan tujuan untuk menjalankan API sehingga sistem dapat memperoleh data yang diinginkan.

Bagian terakhir dari implementasi arsitektur MVVM pada sistem kehadiran adalah *view*. Bagian ini bertanggung jawab atas dua hal. Pertama adalah tampilan sistem yang dilihat oleh *user* dan kedua menjadi pemicu untuk menjalankan API yang ada. Agar API pada *backend* dapat berjalan, maka bagian *view* hanya perlu akses dari kelas *viewmodel*. Akses pertama adalah menjalankan fungsi untuk memperhatikan ketersediaan dengan cara `viewmodel.getLoginResponse(). observe(this, Observer` {respon sistem terhadap ketersediaan data}. Akses kedua adalah menjalankan API dengan kode `viewmodel.login` (masukan data dari *user*).

Selain dari mewujudkan fokus *maintainability* pada proses pengembangan sistem kehadiran. Fokus lain yang perlu dikerjakan pada sistem kehadiran adalah keamanan. Fokus ini dapat dicapai dan dimaksimalkan pada beberapa cara di beberapa bagian. Sehingga nantinya sistem bisa memberikan dampak positif kepada mitra.

Tabel 6. Implementasi *Get* IMEI Pada Fokus Keamanan

Log in		
Layer	**Kelas**	**Hasil**

View	*Login*Activity	`//get imei`
		`var tm: TelephonyManager = getSystemService(Context.TELEPHONY_SERVICE) as TelephonyManager`
		
		`if(ActivityCompat.checkSelfPermission( this, Manifest.permission.READ_PHONE_STATE ) != PackageManager.PERMISSION_GRANTED ) { return }`
		
		`var data = tm.deviceId`

Berdasarkan Tabel 6 implementasi pada fitur *log in* tidak hanya terletak pada fokus *maintainability*. Implementasi pada fitur *log in* juga memperhatikan fokus keamanan dengan menambahkan metode tertentu agar sistem dapat membaca *device* yang digunakan oleh *user*. Fokus keamanan ini dapat

dikembangkan dengan cara membuat sebuah variabel dalam bentuk kode `var tm: TelephonyManager = getSystemService(Context.TELEPHONY_SERVICE) as TelephonyManager`. Variabel ini memiliki fungsi untuk menjadi sebuah metode yang bisa mengakses detail dari perangkat yang digunakan *user*. Setelah inisialisasi metode tersebut selesai proses selanjutnya adalah memanggil data yang diinginkan lalu disimpan dalam sebuah variabel lain dalam bentuk kode `var data = tm.deviceId`.

Tabel 7. Implementasi *Get Location* Pada Fokus Keamanan

Presence		
Layer	**Kelas**	**Hasil**
View	HomeFragment	`private fun buildLocationCallback() { locationCallback = object:LocationCallback() { override fun onLocationResult(p0: LocationResult) { super.onLocationResult(p0) val location = p0.locations.get(p0.locations.size - 1) lat = location.latitude lng = location.longitude }}}`

Pada Tabel 7 implementasi metode *get location* pada fokus keamanan diletakan pada fitur *presence*. Implementasi metode *get location* dilakukan pada *layer view* sehingga tidak mengganggu fungsi dari model dan *viewmodel*. Proses implementasi metode ini dimulai dari membuat sebuah fungsi kode *private* `fun buildLocationCallback(){}`. Sama seperti metode *get* IMEI sebelumnya metode pemanggilan disimpan dalam sebuah variabel `val location = p0.locations.get(p0.locations.size - 1)`. Ketika inisialisasi metode sudah selesai, maka lokasi dapat dipanggil dan bisa langsung disimpan dalam variabel lain. Untuk data *latitude* dapat disimpan dalam variabel `lat = location.latitude` dan *longitude* dapat disimpan `lng = location.longitude`.

Bagian terakhir pada fokus keamanan adalah implementasi dari metode JWT *Authentication*. Berbeda dengan metode *get imei* dan *get location*, implementasi pada JWT *Authentication* ada pada sisi *backend*. Proses implementasi dikerjakan dengan menggunakan bahasa pemrograman *Javasripct* pada *IDE Visual Studio Code*. Implementasi kode tersebut cukup sederhana karena hanya perlu membuat sebuah variabel yang memiliki sebuah metode. `var token = jwt.sign({rows}, config.secret);` metode tersebut akan menghasilkan sebuah token yang mana tiap *user* memiliki token yang berbeda-beda dan nantinya akan dijadikan sebagai masukan data ketika ingin menggunakan fitur lain pada sistem kehadiran.

F. *System Testing*

Proses selanjutnya dari implementasi model pengembangan *PXP* adalah *system testing*. Proses uji sistem dikerjakan dengan menyesuaikan detail kegiatan dari proses *iteration initialization*. Jika rancangan dari fitur tersebut memerlukan data dari basis data melalui *API*, maka uji sistem berjalan dua kali. Pertama untuk melihat kesesuaian *API* dan uji kedua adalah melihat hasil uji fitur secara utuh. Apabila sebaliknya maka proses uji sistem hanya akan berjalan sebanyak satu kali.

Tabel 8. Hasil Uji Sistem Pada Sistem Kehadiran

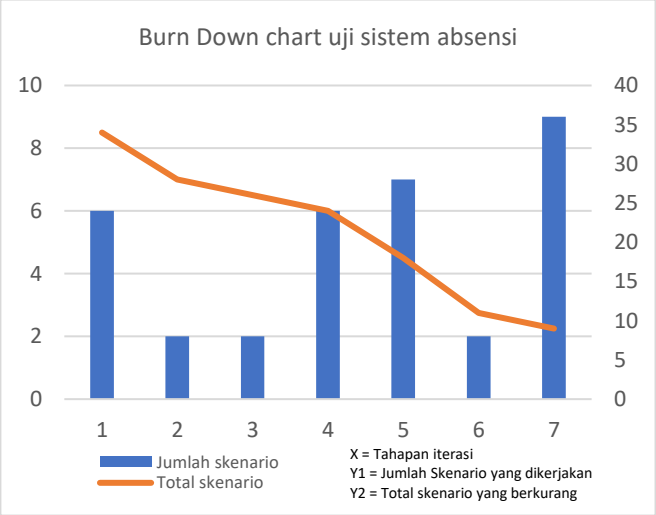
Kode Fitur	**Nama Fitur**	***Test***	**Hasil**
FU-01	*Log in*	Masukan *password* dan *username* lengkap dan benar	Terima
		Masukan data *username* kosong	Tolak
		Masukan data *password* kosong	Tolak
		Masukan data *username* salah	Tolak
		Masukan data *password* salah	Tolak
		Koneksi internet putus	Tolak
FU-02	*Admin Contact*	*Intent* dengan koneksi internet	Terima
		Intent tanpa koneksi internet	Terima
FU-06	*Profile*	Akses halaman dengan koneksi internet	Terima
		Akses halaman tanpa koneksi internet	Terima
FU-07	*Password Change*	Masukan data pertama kosong	Tolak
		Masukan data kedua kosong	Tolak
		Kedua *input* kosong	Tolak
		Kedua *input* tidak sesuai	Tolak
		Proses perubahan dijalankan tanpa koneksi *internet*	Tolak
		Semua *input* terisi dan ada koneksi internet	Terima
FU-03	*Presence*	Hadir kerja di kantor dengan masukan data lengkap	Terima
		Hadir kerja di luar kantor dengan masukan data tidak lengkap	Terima

		Hadir kerja di kantor dengan masukan data tidak lengkap (lokasi tidak dipilih)	Tolak
		Hadir kerja di luar kantor dengan masukan data tidak lengkap (kegiatan tidak dimasukan)	Tolak
		Hadir kerja di kantor dengan data lengkap (tanpa ada koneksi internet)	Tolak
		Hadir kerja di luar kantor dengan data lengkap (tanpa ada koneksi internet)	Tolak
		Hadir kerja di kantor (jarak terlalu jauh)	Tolak
FU-04	*Absence*	*Absence* dengan koneksi internet	Terima
		Absence tanpa koneksi internet	Tolak
FU-05	*Track Record*	Memperoleh data dengan masukan data lengkap	Terima
		Memperoleh data dengan masukan data tidak lengkap	Terima
		Memperoleh data dengan masukan data lengkap (tanpa koneksi internet)	Tolak
		Memperoleh detail data dengan masukan data lengkap	Terima
		Memperoleh detail data dengan masukan tidak data lengkap	Tolak
		Memperoleh detail data dengan masukan data lengkap (tanpa koneksi internet)	Tolak
		Ubah data kegiatan dengan masukan data dan tanggal seusai hari	Terima
		Ubah data kegiatan dengan masukan data dan tanggal tidak sesuai hari	Tolak
		Ubah data kegiatan dengan masukan data dan tanggal (tanpa koneksi internet)	Terima

Berdasarkan dari Tabel 8 implementasi *system testing* pada sistem kehadiran dilakukan dengan menggunakan metode *black box*. Proses uji sistem dilakukan dengan melihat kondisi sistem ketika mendapatkan masukan data dan berjalan dalam dua kondisi berbeda. Kondisi pertama adalah masukan data yang sesuai serta memiliki koneksi internet dan kondisi kedua di mana sistem mendapat masukan data yang tidak sesuai dan tidak memiliki koneksi internet. Secara garis besar dari dua skenario kejadian di atas hasil didapat menunjukan sistem berjalan dengan baik dan berjalan sesuai rancangan. Apabila masukan data dan sistem berjalan sesuai maka sistem dapat berjalan atau bekerja dengan baik dan jika yang terjadi sebaliknya maka sistem tidak akan bekerja dengan baik.

Melalui dua skenario kejadian yang telah dijelaskan sebelumnya, maka hasil dari *system testing* terbagi menjadi dua. Iterasi yang dijalankan dengan model kejadian skenario pertama mendapatkan hasil "Terima" dan apabila yang terjadi pada iterasi tersebut adalah sebaliknya maka hasil menjadi "Tolak". Sehingga dari kedua hasil tersebut peneliti dapat menyimpulkan bahwa sistem berjalan dengan baik karena sistem memberikan respon sesuai dengan masing-masing skenario pada tiap iterasi.

G. *Retrospective*



Gambar 6. Hasil Total Uji Sistem Dari Tiap Iterasi

Berdasarkan Gambar 6 secara garis besar hasil implementasi pada sistem kehadiran dapat disimpulkan dengan membuat *burndown chart.* Nilai dari sumbu X adalah tahapan iterasi pada *system testing*, nilai dari sumbu Y1 adalah jumlah skenario yang diuji pada tiap iterasi, dan Y2 adalah sisa total skenario yang ada pada tiap iterasi. Melalui gambaran informasi yang ada pada *burndown chart* maka dapat dilihat bahwa semakin banyak uji skenario yang dilakukan mendapatkan hasil sesuai dari uji skenario, semakin berkurang pula total uji skenario yang harus dilakukan. Dengan demikian pada akhir implementasi sistem kehadiran nilai dari total uji skenario akan sama dengan nilai uji skenario pada *system testing.*

IV. KESIMPULAN DAN SARAN

A. Kesimpulan

Adapun kesimpulan yang dapat ditarik pada pengembangan sistem kehadiran *online* PT. Lintasmaya Network dengan memperhatikan arsitektur MVVM dan model pengembangan *Personal Extreme Programming* yaitu implementasi metode PXP dan arsitektur MVVM memungkinkan fokus *maintainability* tercapai karena proses

pengembangan yang terstruktur dan terdokumentasi dengan baik di tiap proses dan bagian sistem. Implementasi IMEI, *get location*, dan JWT *Authentication* memungkinkan sistem presensi menjadi lebih aman karena sistem kehadiran hanya dapat digunakan oleh yang memiliki kepentingan saja dengan hasil data yang akurat.

B. Saran

Terkait dengan saran yang dapat disampaikan oleh peneliti kepada pembaca dan mitra untuk pengembangan sistem kehadiran kedepannya adalah membuat sistem menjadi tersedia pada sistem operasi lain seperti iOS. Selain itu perlu dibuat sistem presensi yang memiliki tampilan lebih baik dengan memperhatikan UI dan UX dalam pengembangan berikutnya.

REFERENSI

[1] Eka, S.V. (2018). Strategi Indonesia Menghadapi Industri 4.0. *Info Singkat*, Vol. X(09), pp. 19-24.

[2] Setiawan, P.R. (2020). Aplikasi Absensi Online Berbasis Android. *IT Journal Research and Development*, Vol.5(1), pp. 63-71.

[3] Putera, A.P. (2020). *Rancang Bangun Aplikasi Absensi Online Berbasis Android Menggunakan Metode Deep Learning Pada PT. Pelabuhan Indonesia III (Persero)* [Skripsi]. Universitas 17 Agustus 1945 Surabaya.

[4] Febriandirza, A. (2020). Perancangan Aplikasi Absensi Online Dengan Menggunakan Bahasa Pemrograman Kotlin. *Jurnal Pseudocode*, Vol. 7(2), pp. 123-133.

[5] Subiantoro & Sardiarinto. (2018). Perancangan Sistem Absensi Pegawai Berbasis Web. *Jurnal Swabumi*, Vol. 6(2), pp. 184-189.

[6] Reza. (2020). *Sistem Absensi*. (A. Setiawan, Interviewer).

[7] Marthasari, G., Suharso, W. & Ardiyansyah, F. (2018). Personal Extreme Programming with MoSCoW Prioritization for Developing Library Information System. *Proceeding of the Electrical Engineering, Computer Science, and Informatics*, Vol. 5(51).

[8] Syeikh, W. & Syeikh, N. (2019). A Model-View-Viewmodel (MVVM) Application Framework For Hearing Impairment Diagnosis. *The Journal of Open Source Software*. DOI: 10.21105/joss.02016.

[9] Sai, P. (2018). Multi-Factor Authentication Using IMEI Encrypted Color QR Code. *International Journal of Pure and Mathematics*, Vol. 119(15), pp. 1713-1719.

[10] Pribadi, J.A. & Setiyawati, N. (2021). AbsenLoc: Aplikasi Absensi Mobile Berbasis Lokasi. *Jurnal Sistem dan Teknologi Informasi*, Vol. 9(1), pp. 33-40. DOI: 10.26418/justin.v9i1.41103.

[11] Gunawan, R. & Rahmatulloh, A. (2019). JSON Web Token (JWT) Untuk Authentication Pada Interoperabilitas Arsitektur Berbasis RESTful Web Service. *Jurnal Edukasi dan Penelitian Informatika*, Vol. 5(1), pp. 74-79.