TUGAS AKHIR – KS141501

RANCANG BANGUN BACK-END "SIAP": SISTEM INFORMASI ASPIRASI DAN PENGADUAN MASYARAKAT BERBASIS WEB DENGAN MENGGUNAKAN METODE MICROSERVICE SPRINGBOOT

DEDY PUJI JAYANTO
NRP 5213 100 041

Dosen Pembimbing

Dr.Eng. Febriliyan Samopa, S.Kom., M.Kom.
Hatma Suryotrisongko, S.Kom., M.Eng

JURUSAN SISTEM INFORMASI
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember
Surabaya 2017

FINAL PROJECT – KS141501

APPLICATION BACKEND DEVELOPMENT OF "SIAP" : "SISTEM INFORMASI ASPIRASI DAN PENGADUAN MASYARAKAT" APPLICATION BASED ON WEB USING MICROSERVICE SPRINGBOOT METHOD

DEDY PUJI JAYANTO
NRP 5213 100 041

Supervisors

Dr.Eng. Febriliyan Samopa, S.Kom., M.Kom.
Hatma Suryotrisongko, S.Kom., M.Eng

INFORMATION SYSTEM DEPARTMENT
Faculty of Information Technology
Institut Teknologi Sepuluh Nopember
Surabaya 2017

TUGAS AKHIR – KS141501

RANCANG BANGUN BACK-END "SIAP": SISTEM INFORMASI ASPIRASI DAN PENGADUAN MASYARAKAT BERBASIS WEB DENGAN MENGGUNAKAN METODE MICROSERVICE SPRINGBOOT

DEDY PUJI JAYANTO
NRP 5213 100 041

Dosen Pembimbing

Dr.Eng. Febriliyan Samopa, S.Kom., M.Kom.
Hatma Suryotrisongko, S.Kom., M.Eng

JURUSAN SISTEM INFORMASI
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember
Surabaya 2017

FINAL PROJECT – KS141501

APPLICATION BACKEND DEVELOPMENT OF "SIAP" : "SISTEM INFORMASI ASPIRASI DAN PENGADUAN MASYARAKAT" APPLICATION BASED ON WEB USING MICROSERVICE SPRINGBOOT METHOD

DEDY PUJI JAYANTO
NRP 5213 100 041

Supervisors

Dr.Eng. Febriliyan Samopa, S.Kom., M.Kom.
Hatma Suryotrisongko, S.Kom., M.Eng

INFORMATION SYSTEM DEPARTMENT
Faculty of Information Technology
Institut Teknologi Sepuluh Nopember
Surabaya 2017

LEMBAR PENGESAHAN

RANCANG BANGUN BACK-END "SIAP": SISTEM INFORMASI ASPIRASI DAN PENGADUAN MASYARAKAT BERBASIS WEB DENGAN MENGGUNAKAN METODE MICROSERVICE SPRINGBOOT

TUGAS AKHIR

Disusun Untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Departemen Sistem Informasi
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember

Oleh:

DEDY PUJI JAYANTO
NRP. 5213 100 041

Surabaya, Juli 2017

KEPALA DEPARTEMEN SISTEM INFORMASI

Dr. Ir. Aris Tjahyanto, M.Kom
NIP. 198303062012121001

LEMBAR PERSETUJUAN

RANCANG BANGUN BACK-END "SIAP": SISTEM INFORMASI ASPIRASI DAN PENGADUAN MASYARAKAT BERBASIS WEB DENGAN MENGGUNAKAN METODE MICROSERVICE SPRINGBOOT

TUGAS AKHIR

Disusun Untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Departemen Sistem Informasi
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember

Oleh:

DEDY PUJI JAYANTO
NRP. 5213 100 041

Surabaya, Juli 2017

Disetujui Tim Penguji : Tanggal Ujian : 05 Juli 2017
Periode Wisuda: September 2017

Dr.Eng. Febriliyan Samopa, S.Kom., M.Kom. (Pembimbing I)

Hatma Suryotrisongko, S.Kom., M.Eng (Pembimbing II)

Ir. Aris Tjahyanto, M.Kom (Penguji I)

Nisfu Asrul Sani, S.Kom, M.Sc (Penguji II)

RANCANG BANGUN BACK-END "SIAP": SISTEM INFORMASI ASPIRASI DAN PENGADUAN MASYARAKAT BERBASIS WEB DENGAN MENGGUNAKAN METODE MICROSERVICE SPRINGBOOT

Nama Mahasiswa : DEDY PUJI JAYANTO

NRP : 5213 100 041

Departemen : SISTEM INFORMASI FTIF-ITS

Dosen Pembimbing 1 : Dr.Eng. Febriliyan Samopa, S.Kom., M.Kom.

Dosen Pembimbing 2 : Hatma Suryotrisongko, S.Kom., M.Eng

ABSTRAK

E-government merupakan salah satu bentuk implementasi ICT (information and communication technologies) di bidang kepemerintahan dalam meningkatkan layanan kepada masyarakat oleh pemerintahan/sektor publik. Sebagai contoh, Smart City, layanan perijinan online, layanan pengaduan masyarakat, dll. Inpres no. 3 tahun 2003 menganjurkan pemerintah untuk menerapkan E-government dalam sistem kepemerintahan. Selain itu sudah banyak masyarakat yang paham dalam mengakses internet. Oleh karena itu E-government menjadi salah satu cara yang efektif dan dapat menjangkau banyak kalangan masyarakat dalam menghubungkan antara masyarakat dengan pemerintah

sehingga tercipta kerjasama yang baik dalam membangun daerahnya.

Tujuan dari Tugas Akhir ini adalah mengembangkan aplikasi SIAP: Sistem Informasi Aspirasi dan Pengaduan Masyarakat yang kemudian bisa dimanfaatkan oleh semua kabupaten/kota di Indonesia. Dengan konsep e-government as a Service, kontribusi hasil penelitian ini adalah memungkinkan pemerintahan kabupaten/kota se-Indonesia untuk bisa dengan mudah mengimplementasikan layanan Aspirasi dan Pengaduan Masyarakat berbasis web serta dengan biaya investasi yang rendah dan memanfaatkan sumber daya manusia yang ada, karena infrastruktur aplikasi, keamanan, penanganan error, update program dilakukan oleh pihak tim peneliti selaku penyedia layanan e-government as a Service SIAP.

Metode yang digunakan di Tugas Akhir ini adalah menggunakan arsitektur Microservices Springboot. Penggunaan microservice adalah dengan membagi-bagi fungsionalitas aplikasi menjadi banyak bagian atau banyak layanan yang kecil (micro) yang saling berhubungan satu sama lain sehingga menjadi satu kesatuan bisnis proses aplikasi.

Kata kunci : e-government, microservices, aplikasi Aspirasi dan Pengaduan masyarakat, spring boot

APPLICATION BACKEND DEVELOPMENT OF "SIAP: SISTEM INFORMASI ASPIRASI DAN PENGADUAN MASYARAKAT"APPLICATION BASED ON WEB USING MICROSERVICE SPRINGBOOT METHOD

Name : DEDY PUJI JAYANTO

NRP : 5213 100 041

Departement : INFORMATION SYSTEM FTIF-ITS

Supervisor 1 : Dr.Eng. Febriliyan Samopa, S.Kom., M.Kom.

Supervisor 2 : Hatma Suryotrisongko, S.Kom., M.Eng

ABSTRACT

E-government is one form of implementation of ICT (information and communication technologies) in the field of governance in improving services to the public by government or public sector. For example, Smart City, online licensing services, community complaint services, etc. Presidential Instruction no. 3 of 2003 recommend the government to implement e-government in the governance system. In addition there are many people who understand in accessing the internet. Therefore E-government becomes one effective way that can reach whole people in connecting between society and government so that can create good cooperation for developing the area.

The purpose of this Final Project is to develop the application SIAP: "Sistem Informasi Aspirasi dan Pengaduan" which then can be utilized by all districts / cities in Indonesia. With the concept of e-government as a Service, the contribution of this research result is to enable the district / city government in Indonesia to be able to easily implement web-based Aspirations and Complaints service as well as with low investment cost and utilize existing human resources, Application, security, error handling, program update done by the research team as the service provider of the e-government as a Service SIAP.

The method used in this Final Project is to use Springboot Microservices architecture. The use of microservice is to divide the application functionality into many parts or many micro services (micro) based on business process and services are interconnected each other that become a single application with complete business process.

Keywords: e-government, microservices, complaint services, spring boot.

KATA PENGANTAR

Bismillahirrohmanirrohim.

Alhamdulillahirabil'alamin, sejuta ucapan terimakasih kepada ALLAH SWT yang telah melimpahkan rahmat dan karunianya sehingga penulis dapat menyelesaikan tugas akhir penulis yang diberi judul **"RANCANG BANGUN BACK-END "SIAP": SISTEM INFORMASI ASPIRASI DAN PENGADUAN MASYARAKAT BERBASIS WEB DENGAN MENGGUNAKAN METODE MICROSERVICE SPRINGBOOT"** telah berhasil diselesaikan tepat waktu dan merupakan salah satu syarat kelulusan pada Departemen Sistem Informasi, Fakultas Teknologi Informasi Komunikasi (FTIK), Institut Teknologi Sepuluh Nopember.

Dalam pengerjaan tugas ini, tentunya penulis banyak mendapatkan dukungan dan bantuan dari berbagai belah pihak, untuk itu izinkan penulis menuliskan rasa terimakasih kepada semua yang berjasa yang membuat semua pengerjaan tugas akhir ini selesai:

1. Allah SWT yang telah menuntun dan memberikan rahmat dan hidayahnya sehingga dapat menyelesaikan tugas akhir ini tepat waktu.

2. Kedua orang tua yang selalu mendukung anaknya, Bapak Moh. Faisal dan Ibu Rumiyati, dan juga adik tercinta Dandi Yudit yang selalu memberikan dukungan dan doa yang tiada henti untuk penulis.

3. Bapak Dr.Eng. Febriliyan Samopa, S.Kom., M.Kom atau Bapak iyan, selaku dosen pembimbing satu yang telah membimbing dan mengarahkan penulis dalam menyelesaikan tugas akhir ini.

4. Bapak Hatma Suryotrisongko, S.Kom., M.Eng atau Bapak Hatma selaku pembimbing dua yang juga telah membimbing dan mengarahkan penulis dalam menyelesaikan tugas akhir ini.

5. Bapak Ir. Aris Tjahyanto, M.Kom selaku dosen peguji I, sekaligus ketua Departement Sistem Informasi FTIK – ITS yang telah memberikan masukan dan saran kepada penulis

6. Bapak Nisfu Asrul Sani, S.Kom, M.Sc selaku dosen penguji II yang telah memberikan masukan dan saran kepada penulis

7. Teman-teman BPH dan seluruh staff Badan Semi Otonom VIVAT PRESS BEM dua generasi yang memberikan dukungan penuh bagi penulis dalam menyelesaikan tugas akhir ini dan memberikan arti dalam berjuang.

8. Teman-teman BELTRANIS Sistem Informasi Angkatan 2013 Mia Eka Setyaningsih, Alimul Hakim, Harun Rizal, Hanif Rusdiyansah, Fikri Jember, Tayomi Dwi yang selalu memberikan dukungan dan selalu ada dan teman-teman angkatan 2013 BELTRANIS yang telah memberikan kenangan suka dan duka selama 4 tahun kuliah ini. SUKSES KOMPAK SELALU ya rek!.

9. Teman-teman Kontrakan Al-Qonaqun: Evan, Hamzah, Dwiki yang berjuang bersama menyelesaikan tugas akhir. Irvan, Utha, Baymaji, dan Haryo yang selalu memberikan semangat dengan caranya sendiri. Terimakasih telah hadir di tahun terakhir ini.

10. Penulis juga mengucapkan terimakasih kepada pihak-pihak lain yang tidak dapat disebutkan satu per satu yang membantu penulis dalam menyelesaikan tugas akhir ini.

Tugas akhir ini tentunya tidak luput dari kesalahan dan tidak sempurna, maka dari itu penulis mengharapkan kritik dan saran yang membangun dari pembaca sekalian. Terimakasih.

Surabaya, 23 Juni 2017

Penulis

Halaman ini sengaja dikosongkan

DAFTAR ISI

Halaman ini sengaja dikosongkan

DAFTAR GAMBAR

Halaman ini sengaja dikosongkan

DAFTAR TABEL

Halaman ini sengaja dikosongkan

BAB 1
PENDAHULUAN

Bab pertama akan menjelaskan tentang pendahuluan pengerjaan tugas akhir ini, yang meliputi latar belakang, rumusan permasalahan, batasan masalah, tujuan penelitian hingga manfaat yang diperoleh dari penelitian ini.

1.1 Latar Belakang Masalah

Perkembangan *information and communication technologies (ICT)* berkembang sangat pesat dan menyeluruh ke bagian-bagian kehidupan masyarakat. Pesatnya perkembangan *ICT* akan membuka peluang dan tantangan untuk menciptakan (*to create*), mengakses (*to access*), mengelola (*to process*), dan memanfaatkan (*to utilize*) informasi secara tepat dan akurat [**1**]. Penerapannya bisa dimanfaatkan di segala bidang termasuk juga bidang pemerintahan. Salah satu penerapannya di bidang pemerintahan yang biasa dikenal sebagai *E-government*.

E-government pada dasarnya terdiri dari penggunaan teknologi komunikasi elektronik seperti internet, dalam meningkatkan dan memajukan akses warga terhadap pelayanan publik [**2**]. Penerapan *E-governmen*t memberikan efisiensi dan kecepatan pengelolaan pada sistem administrasi laporan, serta transparansi dari proses-proses yang terjadi pada administrasi pemerintahan. Melalui hal tersebut maka muncul suatu aspek yang disebut *good governance* yang secara sederhana bisa kita artikan sebagai tata kelola proses yang baik.

Pada Inpres No.3 tahun 2003 tentang kebijakan dan strategi pengembangan *e-government*, pemerintah harus bisa memanfaatkan ICT untuk meningkatkan efisiensi, efektifitas, transparansi dan akuntabilitas pemerintah [**3**]. Jenis layanan *E-government* yang bisa diterapkan salah satunya adalah layanan interaksi. Layanan interaksi memungkinkan pemerintah melakukan komunikasi dua arah misalnya berupa pengaduan atau konsultasi [**4**].

Jakarta, Yogyakarta, Provinsi Sulawesi Selatan, serta Provinsi Sumatera Utara telah mempunyai *website* yang memberikan informasi untuk melakukan pengaduan di *website* resmi pemerintahannya. Penerapan layanan interaksi *E-government* mulai banyak pengembangannya di zaman yang serba informasi ini.

Jumlah penduduk Indonesia yang mempunyai akses terhadap internet sudah sangat tinggi pada masa ini. Terdapat 132,7 juta orang yang sudah bisa mengakses internet pada awal tahun 2017 berdasarkan statistika perusahaan riset "We Are Social" [**5**] dan diperkirakan terdapat 100 juta orang yang sudah memiliki perangkat genggam di tangannya pada tahun 2016 lalu. Layanan interaksi *E-government* melalui dunia internet dan yang bisa diakses melalui perangkat genggam akan menjadi salah satu solusi yang efektif untuk menghubungkan masyarakat dengan pemerintahannya dan menjangkau seluruh masyarakatnya sehingga tercipta kerja sama yang sinergi dalam pembangunan daerahnya masing-masing.

Penerapan *E-government* di Indonesia tidaklah mudah. Banyak tantangan yang perlu dihadapi agar penerapannya bisa mengalami keberhasilan. Isu ekonomis mengenai biaya pengembangan dan operasi aplikasi *e-government*, isu-isu teknis seperti masalah keamanan, privasi, dan interoperabilitas sistem serta isu sumber daya manusia yang kurang mempunyai kapabilitas dalam mengelolahnya. Isu infrastruktur juga menjadi masalah tersendiri. Terkadang masih banyak pemerintahan yang tidak bisa merevolusi sistem pemerintahannya karena tidak memiliki alat-alat infrastruktur yang memadai karena pengadaannya dianggap terlalu mahal.

Perkembangan *penelitian* teknologi yang semakin canggih menghadirkan beberapa alternatif jawaban atas permasalahan penerapan *E-government*. Teknologi *cloud* menjadi salah satu pilihan jawaban. Penggunaan *cloud* hari ini semakin banyak. Salah satu model dalam penggunaan *cloud* adalah *Software as*

a Service (SaaS). Model ini memungkinkan konsumen dapat menggunakan aplikasi yang ada di *cloud* secara *online* oleh penyedia yang bisa diakses diberbagai macam perangkat. Salah satu penggunaan SaaS adalah aplikasi Google Docs yang memungkinkan kita mengedit dokumen secara *online* dengan menggunakan perangkat komputer maupun perangkat *smartphone*. Tetapi implementasi SaaS pada bidang *E-government* masih jarang ditemukan.

Selain teknologi *cloud*, muncul tren baru dikalangan peneliti/praktisi *Software Architect*, yaitu *microservices*. Secara sederhana hal ini merupakan *software* atau sistem informasi dirancang untuk terdistribusi dan memberikan layanan spesifik dan terfokus. Istilah *microservices* telah mulai diperkenalkan sejak tahun 2011 ("*Microservices*," n.d.), namun pola desain arsitektur *software* atau sistem informasi ini menjadi semakin banyak diteliti dan digunakan industri setelah munculnya teknologi *container-based virtualization*, yaitu Docker di akhir tahun 2014 [**6**]. Pentingnya *microservices* dalam pengembangan sebuah aplikasi adalah untuk mempercepat permintaan terhadap layanan pada aplikasi.

Penggunaan teknologi *microservices* dapat memberikan beberapa kelebihan kepada sistem *E-government*. Konsep modularitas pada *microservices* memungkinkan pengelolaan *service-service* yang ada pada suatu aplikasi secara terpisah-pisah. Dampaknya apabila ingin melakukan pengembangan pada satu *service* tertentu maka tidak akan menggangu *service* lain. Pengembangan kapasitas suatu *service* bisa dibedakan antar *service* lain. Sehingga *resource* yang digunakan secara tepat. Selain itu pengembangan antar *service* bisa dikembangkan dengan bahasa pemrograman yang berbeda-beda [**7**]. Hal ini memberikan keuntungan kepada tim pengembang sehingga bisa bekerja secara cepat dan tepat.

Penerapan *microservices* juga menjamin kecepatan dalam pelayanan layanan, kecepatan dalam menangani permasalahan

yang terjadi dalam sistem, dan juga kemudahan dalam menyesuaikan kapasitas layanan sesuai dengan permintaan pengguna terhadap layanan [**8**].

Berdasarkan karakteristik *cloud computing* dan *microservices*, dua teknologi tersebut bisa menjawab isu-isu dalam penerapan *e-government*. Oleh karena itu, tujuan dari rencana penelitian tugas akhir ini adalah mengembangkan layanan *e-government* "SIAP: Sistem Informasi Aspirasi dan Pengaduan Masyarakat berbasis web" yang kemudian bisa dimanfaatkan oleh semua kabupaten/kota di Indonesia. Berdasarkan teknologi *cloud* SaaS maka muncul konsep *E-government* as a *Service*, yaitu memungkinkan pemerintahan kabupaten/kota se-Indonesia untuk bisa dengan mudah mengimplementasikan layanan *E-government* Aspirasi dan Pengaduan Masyarakat berbasis web, karena infrastruktur aplikasi, keamanan, penanganan *error*, *update* program dilakukan oleh pihak tim peneliti selaku penyedia layanan *E-government as a Service*. Oleh karena itu diharapkan implementasi aplikasi *E-government* ini dapat mengurangi beban biaya pemerintahan dalam pembelanjaan alat-alat untuk membangun infrastruktur *E-government*.

Pengembangan aplikasi *microservices* "SIAP" akan menggunakan *framework* Spring boot. *Framework* Spring boot berasal dari keluarga Spring yang sudah populer digunakan dalam membangun aplikasi sejenis. Penggunaan *framework* Spring boot memberikan beberapa keuntungan terutama bagi yang pemula menggunakan *framework* tersebut. Selain itu Spring boot akan memberikan konfigurasi-konfigurasi yang dibutuhkan untuk aplikasi yang akan di *deploy* ke *cloud*. Spring boot juga mendukung pengembangan *microservices* karena mendukung REST *service* sebagai salah satu komponen penting *microservices*.

Penelitian tugas akhir ini diharapkan menjadi salah satu aplikasi e-government untuk pelaporan yang bisa digunakan oleh seluruh kota/kabupate se-Indonesia dengan mudah serta

dengan biaya infrastruktur yang rendah, dan memanfaatkan sumber daya manusia yang sudah ada.

1.2 Rumusan Masalah

Perumusan masalah untuk penelitian tugas akhir ini adalah sebagai berikut:

1. Bagaimana arsitektur aplikasi layanan aspirasi dan pengaduan masyarakat sebagai *software as a service* dengan arsitektur *microservice* untuk *e-government*?
2. Bagaimana merancang dan membangun aplikasi *e-government* layanan aspirasi dan pengaduan masyarakat dengan menggunakan metode *microservice* spring boot?

1.3 Batasan Masalah

Batasan masalah untuk penelitian tugas akhir ini adalah :

1. Prototipe dikembangkan dengan metode Spring boot *microservices*.
2. Fokus tugas akhir ini lebih kepada *back-end* aplikasi.
3. Berakhirnya tugas akhir adalah terbentuknya aplikasi SIAP yang telah memenuhi kebutuhan yang disepakati, telah melewati pengujian yang juga telah disepakati sebelumnya dan tidak atau belum termasuk penerapannya dalam pemerintah kabupaten/kota/provinsi.

1.4 Tujuan Penelitian

Tujuan dari penelitian tugas akhir ini adalah untuk bisa mengembangkan aplikasi SIAP: Sistem Informasi layanan Aspirasi dan Pengaduan Masyarakat berbasis web yang kemudian bisa dimanfaatkan oleh semua kabupaten/kota di Indonesia.

1.5 Manfaat Penelitian

Manfaat penelitian tugas akhir yang diusulkan ini adalah :

1. Mendukung proyek penelitian lab IKTI yang didanai oleh Hibah Lab PUPT 2017.

2. Pengembangan aplikasi layanan Aspirasi dan Pengaduan Masyarakat berbasis web yang kemudian bisa dimanfaatkan oleh semua kabupaten/kota di Indonesia.

1.6 Relevansi

Tugas Akhir ini disusun untuk memenuhi salah satu syarat kelulusan sebagai Sarjana Komputer di Jurusan Sistem Informasi, Institut Teknologi Sepuluh Nopember . Tugas akhir ini juga sesuai dengan topik bidang penelitian unggulan ITS (ICT), serta sesuai dengan *roadmap* lab IKTI JSI ITS. Visi / target jangka panjang dari *roadmap* penelitian lab IKTI JSI ITS yang terkait dengan tugas akhir ini adalah untuk mengembangkan *Resilient Information Systems* untuk Kebencanaan (*Disaster*) dan *E-government*.

Penerapan sistem *E-government* tidak hanya berpusat pada pembuatan sistem informasinya saja, melainkan juga membutuhkan sumberdaya manusia dan infrastruktur untuk dapat mengoperasikan sistem tersebut dengan baik dan dapat mengatasi berbagai macam gangguan/permasalahan yang muncul. Pentingnya penerapan *E-government* pada masa pemerintahan saat ini, menuntut adanya pemecahan masalah yang tepat atas dari isu-isu penerapan *E-government* di Indonesia, dimana kondisi kekiniannya adalah memiliki sumberdaya manusia dan infrastruktur yang kurang memadai. Sehingga penting bagi lab IKTI untuk turut mengembangkan penelitian untuk bisa membuat Desain Arsitektur *Software*: *Resilient Information Systems* untuk kebencanaan dan *e-government*.

1.6.1 Roadmap Rencana Penelitian Lab IKTI (Infrastruktur dan Keamanan Teknologi Informasi)

Isu strategis yang menjadi basis dan fokus penelitian laboratorium Infrastruktur Sistem dan Teknologi Informasi Jurusan Sistem Informasi, Fakultas Teknologi Informasi (FTIF) tahun 2015-2020 adalah Infrastruktur Teknologi Informasi (fokus pada *middle layer* ke bawah dari OSI *Layer*), Energi, *Military Defense*, Perilaku Pengguna TI (lebih spesifik

adalah internet), Transportasi dan Logistik, Kesehatan, Efisiensi melalui *monitoring* konsumsi pemakaian listrik, Peningkatan pemanfaatan potensi laut melalui pengembangan teknologi pada bidang kemaritiman dan Desain Arsitektur *Software* : *Resilient Information Systems*.

Tabel 1.1. Roadmap Rencana Penelitian Lab IKTI

Isu-Isu strategis: Desain *Arsitektur Software : Resilient Information Systems* untuk kebencanaan dan *e-government*
Konsep Pemikiran: ICT, kebencanaan dan *e-government* adalah topik penelitian unggulan ITS. Peran teknologi informasi dalam situasi mitigasi maupun penanganan ketika terjadi/pasca bencana, terbukti penting dan sangat signifikan. Seperti pada bencana tsunami Jepang 2011, gempa bumi di Jogja 2006, dll. Di masa lalu, menjadi responsif dalam kasus gangguan yang tidak terencana sangat menyulitkan manajemen. Untuk IT, hal tersebut bahkan lebih menantang: Sistem yang dikembangkan dirancang hanya untuk memenuhi sifat yang telah ditetapkan, dan hanya menawarkan satu set fungsionalitas terprogram untuk penanganan eksepsi. *Resilience* meliputi reaksi atas gangguan di luar lingkup yang sudah dikenal sebelumnya. *Software*/sistem informasi dianggap resilient jika memiliki kemampuan menyesuaikan kebutuhan baru yang belum eksplisit ada di dalam desain IT sebelumnya. *e-government* adalah implementasi ICT untuk layanan publik pada pemerintahan. Sebagai contoh, layanan perijinan *online*, layanan aspirasi/suara warga berbasis aplikasi, dll. Implementasi sebuah sistem *e-government* tidak hanya sekedar butuh kode program/*database* saja, melainkan juga membutuhkan sumberdaya manusia serta infrastruktur untuk dapat mengoperasionalkan sistem tersebut dengan tingkat ketersediaan yang baik dan dapat

mengatasi berbagai macam gangguan/permasalahan yang muncul. Kesadaran akan pentingnya ketersediaan layanan publik yang berbasis *online* dan aplikasi akhir-akhir ini, menuntut adanya terbosan dalam solusi inovasi *e-government* yang tepat untuk kondisi di Indonesia, dimana tidak semua memiliki sumberdaya manusia dan infrastruktur yang memadai untuk menerapkan *best practice e-government*.

Sehingga penting bagi lab IKTI untuk turut mengembangkan penelitian untuk bisa membuat Desain *Arsitektur Software : Resilient Information Systems* untuk kebencanaan dan *e-government*, dengan potensi *output* berupa paten dan teknologi tepat guna.

Pemecahan Masalah:

- Diperlukan penelitian pada *state of the art* tentang *software* arsitektur terdistribusi.
- Diperlukan penelitian pada *state of the art* tentang *scalability* pada sistem informasi.
- Diperlukan penelitian pada *state of the art* tentang *reliability* pada sistem informasi.
- Diperlukan penelitian pada *state of the art* tentang *maintainability* pada sistem informasi.
- Diperlukan penelitian pada *state of the art* tentang *availability* pada sistem informasi.

Topik yang Diperlukan:

- Kajian/pengembangan pada topik *Cloud Infrastruktur*
- Kajian/pengembangan pada topik sistem terdistribusi
- Kajian/pengembangan pada topik *Microservicess*
- Kajian/pengembangan pada topik *Docker-container virtualization*

BAB 2
TINJAUAN PUSTAKA

Bab kedua akan menjelaskan mengenai penelitian sebelumnya dan dasar teori yang dijadikan acuan atau landasan dalam pengerjaan tugas akhir ini. Landasan teori akan memberikan gambaran secara umum dari landasan penjabaran tugas akhir ini.

2.1 Penelitian Sebelumnya

Dalam proses pengerjaan penelitian tugas akhir ini, dilakukan pencarian penelitian-penelitian yang sudah dilakukan untuk menjelaskan posisi penelitian ini. Penelitian-penelitian sebelumnya mengenai arsitektur *microservices* juga bisa dijadikan sebagai referensi yang dapat membantu proses pengerjaan penelitian tugas akhir ini.

Pembuatan Sistem Microservicess dengan Node.js: (Purnama & Yatini, 2016) mengembangkan aplikasi pengelolaan skripsi di STMIK AKAKOM Yogyakarta dengan tujuan untuk menghindari adanya kesamaan topik atau judul skripsi yang selama ini sering terjadi kasus plagiarisme. Pengembangan aplikasi menggunakan arsitektur *microservices* dengan alasan kemudahan dalam pengembangan aplikasi tersebut. Ketika terjadi kasus penambahan suatu *service* baru maka tidak lagi membuat ulang aplikasi tetapi hanya tinggal menambahkan sehingga memunculkan nilai bisnis yaitu penghematan waktu dalam pengembangan aplikasi.

Platform as a Service dan Cloud Computing: (Alim & Cancer, 2016) mempunyai maksud agar fokus pengembangan *software* lebih dalam dari pada harus memikirkan mengenai biaya perawatan dan konfigurasi infrastruktur dengan menggunakan teknologi *cloud computing* layanan *Platform as a Service*. Melalui bantuan *cloud computing* tadi, pengembang tidak perlu lagi memikirkan biaya pembelian perangkat keras, konfigurasi jaringan, konfigurasi keamanan, dan perawatan sumber daya. Biaya dapat dialihkan ke sesuatu yang lain misalnya pengembangan aplikasi.

E-government* : *Microservicess* / *Software as a Service
Dua buah publikasi penelitian yang paling relevan dengan proposal penelitian ini adalah (Hole, 2016) dan (Marijn Janssen & Anton Joha, 2011).

(Hole, 2016) [**9**] berargumen mengenai apakah sebaiknya sistem *e-government* dibangun dalam satu *platform* sistem yang mencakup *scope* besar/nasional, atau sistem *e-government* sebaiknya dibuat terdesentralisasi menjadi tugas dan tanggung-jawab masing-masing kota/kabupaten. Hole (2016) menyarankan bahwa *public cloud* menjadi solusi tepat untuk membuat sistem *e-government* yang *reliable* dan terpusat

Kemudian, pada paper (Janssen & Joha, 2011) [**10**], dipaparkan bahwa penggunaan model *Software as a Service* pada sektor publik masih sangat jarang dan belum tersentuh. Walaupun *SaaS* pada sektor publik / *e-government* menjanjikan banyak keuntungan, seperti *cost-saving* dan efektivitas, namun tantangan-tantangan yang dihadapi juga sangat berat seperti kualitas, keamanan, *privacy*, dan juga termasuk kebutuhan untuk bisa *custom*/memiliki perbedaan pada sistem di daerah satu dengan daerah lainnya.

Pada infrastruktur *cloud* yang akan dibangun, aplikasi *e-government* "SIAP: Sistem Informasi layanan Aspirasi dan Pengaduan Masyarakat berbasis WEB" disusun dengan arsitektur *microservices*, sehingga sebuah layanan *e-government* pada satu tempat bisa berbeda dari tempat lain, karena aplikasi dapat di susun dari berbagai *microservices* yang berbeda-beda sesuai dengan kebutuhan pada kabupaten/kota/propinsi. Dengan dua pendekatan metode/teknologi terkini tersebut, seluruh kota/kabupaten/propinsi se-Indonesia dapat mudah mengimplementasikan berbagai *best-practise* layanan Aspirasi dan Pengaduan Masyarakat berbasis WEB dengan investasi biaya, infrastruktur dan sumberdaya manusia yang rendah.

Dengan paparan beberapa publikasi penelitian selama 5 tahun terakhir terkait *Microservices* diatas, diharapkan mampu menjelaskan posisi penelitian ini dan kontribusinya yaitu implementasi *microservices* untuk studi kasus *e-government*: aplikasi layanan Aspirasi dan Pengaduan Masyarakat.

Building Microservice: Designing Fine-Grained Systems.

Sebuah publikasi yang berupa buku oleh [**11**] yang berjudul Building Microservice: Designing Fine-Grained Systems menjelaskan mengenai teori microservice dan beberapa praktik dalam membangun aplikasi berjenis microservice.

Pada Bab 3 yang berjudul How to Model Services menjelaskan bagaimana menyusun microservice dari sebuah aplikasi. Dalam penjelasannya untuk memulai membangun sebuah aplikasi microservice, pengembang harus tau terlebih dahulu tujuan besar yang ingin dicapai dari pembangunan aplikasi.

Dalam membangun service maka perlu diketahui konsep dari *loose coupling* dan *high cohesion.* Penjelasan dalam buku secara sederhana menjelaskan bahwa loose coupling adalah keadaan dimana ketika ingin mengubah suatu fungsi atau bagian dari aplikasi maka seharusnya tidak membutuhkan pengubahan fungsi di bagian yang lain. Konsep microservice ini adalah bisa dilakukan perubahan pada suatu service dan bisa dilakukan deploy tanpa menganggu bagian lain dari aplikasi. Sedangkan konsep high cohesion adalah fungsi dengan behavior yang berhubungan harus dijadikan dalam satu lingkungan sedangkan behavior yang lain harus ditempatkan dalam lingkungan yang lain. Sehingga harus dicari batasan dalam menentukan lingkungan sehingga bisa mengelompokkan fungsi-fungsi dengan behavior yang terkait.

Sehingga dalam menyusun aplikasi microservice maka hal selanjutnya adalah menentukan batasan lingkungan atau dalam buku disebut dengan istilah batasan konteks. Secara sederhana menentukan batasan konteks dari aplikasi bisa melihat dari proses bisnis aplikasi yang kemudian bisa dikelompokkan berdasarkan fungsional dari pengguna misal departemen finansial mengurus pembayaran, dan departemen warehouse mengurus pemesanan konsumen. Fungsional tersebut nantinya akan dijadikan sebuah satu modul. Sehingga modul tersebut menjadi batasan konteks untuk membuat microservice-microservice yang sesuai dengan tujuan modul tersebut dibuat. Sehingga microservice-microservice yang sudah dibuat akan menerapkan konsep *loose coupling* antara modul yang lain dan *high cohesion* microservice yang saling berhubungan dengan modul yang dibuat. Akhirnya bisa ditentukan bagian mana yang nantinya akan melakukan penulisan data ke database dan pembacaan data dari database.

2.2 Dasar Teori

A. Software as a Services (SaaS)

Teknologi *cloud computing* sudah tidak asing lagi pada perkembangan teknologi modern saat ini. *Cloud computing* merupakan suatu model yang memungkinkan akses secara bebas tempat, mudah, sesuai kebutuhan terhadap *server*, penyimpanan, aplikasi, dan layanan dengan upaya manajemen sumber daya yang minimal [**12**]. Pada teknologi *cloud computing* terdapat 3 layanan yang tersedia:

1. Platform as a Service
2. Software as a Service
3. Infrastructure as a Service

Penggunaan teknologi *cloud computing* memberikan beberapa kelebihan terhadap penggunanya. Pengguna bebas menggunakan teknologi *cloud* sesuai dengan kebutuhan dan kemampuan pengguna untuk menyewa layanan *cloud*. Biaya penyewaan tersebut juga menjadi hal yang menguntungkan

karena pengguna tidak lagi dibingungkan dengan masalah perawatan infrastruktur, dan keamanan misalnya. Pengguna bebas menggunakan layanan *cloud computing* dimana saja dan kapan saja dengan terhubung ke jaringan internet.

Software as a Service (SaaS) merupakan suatu aplikasi yang dibangun dan kemudian diletakkan dalam sistem *cloud* oleh *vendor* aplikasi tersebut. Pengguna mengakses aplikasi tersebut menggunakan jaringan internet. Berbeda dengan aplikasi yang biasanya harus dipasang dan dikonfigurasi pada perangkat komputer pribadi yang menggunakan *resource* komputer atau *server* tersebut, *SaaS* dimiliki oleh *vendor* dan berjalan pada komputer atau *server vendor* [**13**].

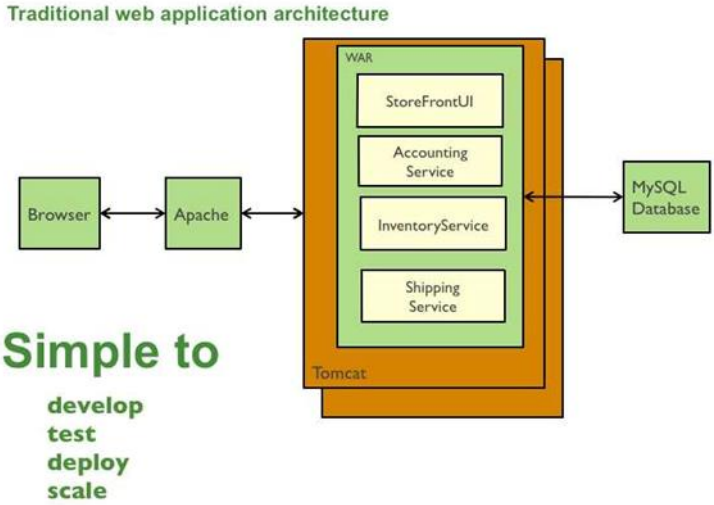
Permintaan terhadap layanan *SaaS* bersifat dinamis. Pada suatu waktu permintaan layanan bisa menjadi sangat banyak dan *vendor* harus memenuhi permintaan tersebut. Oleh karena itu dibutuhkan sebuah *Service Level Agremeent (SLA)* yaitu sebuah perjanjian atau *vendor* dan pengguna dalam pemenuhan layanan. Selain itu dibutuhkan terobosan sistem teknologi yang dapat mengelola *resources* dari *SaaS* agar memenuhi permintaan layanan.

B. Microservices

Ada banyak solusi yang diusulkan oleh para peneliti untuk dapat meningkatkan kualitas *software*/sistem informasi dari sisi *non-functional requirements* (*scalability*, *reliability*, *maintainability* dan *availability*). Mulai dari penggunaan teknik *mirrorring*/DRC/*cloud*/sampai dengan pendekatan algoritma untuk dapat mengembalikan performa sistem setelah terjadi serangan [**14**].

Salah satu solusi yang dapat digunakan adalah teknologi *Microservices*. *Microservices* menjadi salah satu teknologi yang sering dibicarakan dalam pengembangan aplikasi modern ini. Sebelum mengetahui *microservices*, kita harus paham salah satu arsitektur aplikasi yang sekarang masih banyak digunakan yaitu arsitektur *monolithic*.

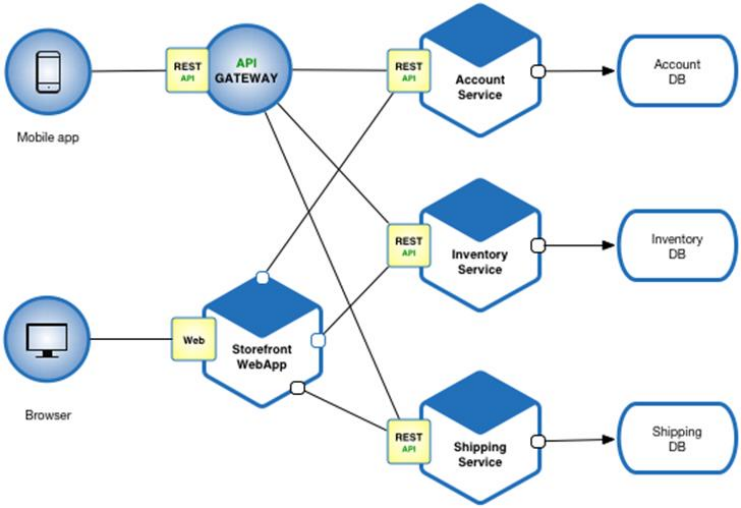
Arsitektur monolithic merupakan arsitektur aplikasi terpusat dan teknologi yang seragam [**15**]. Pengembangan aplikasi menggunakan arsitektur ini menggunakan bahasa *programming* yang sama dan teknologi yang serupa. Aplikasi terbungkus dalam satu *package* besar, dimana ketika terjadi perubahan pada salah satu bagian aplikasi maka bisa berdampak pada aplikasi secara keseluruhan seperti yang ditunjukkan oleh Gambar 2.1 Salah satu aplikasi arsitektur *monolithic* adalah wordpress.



Gambar 2.1. Gambar Arsitektur Microservice

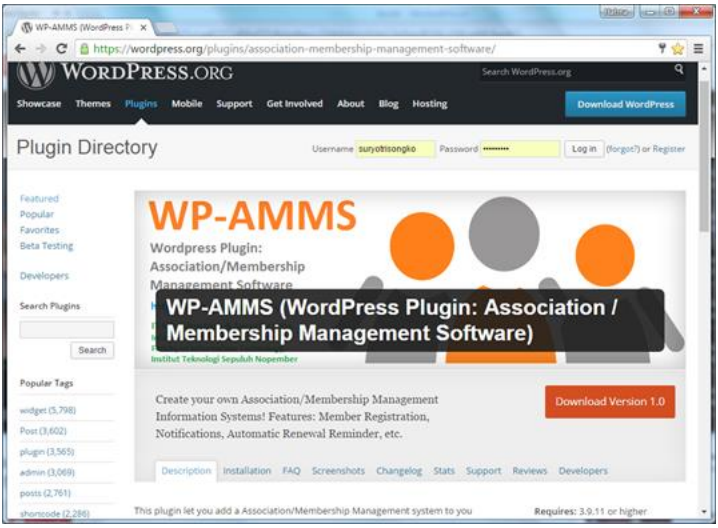
Arsitektur *microservices* secara sederhana adalah suatu arsitektur dengan memecah aplikasi menjadi bagian-bagian fungsi kecil yang spesifik. Sudut pandang paradigma *microservices*, sebuah konsep kunci dalam *resilient engineering* adalah penyekatan. Jika salah satu komponen dari sistem gagal, kegagalan tersebut tidak akan *cascade* / memberikan pengaruh ke kinerja komponen yang lain. Kita dapat mengisolasi masalah dan sisanya dari komponen-komponen sistem yang lain dapat terus bekerja.

Fungsi-fungsi yang terdapat pada arsitektur bisa dibuat dengan menggunakan teknologi *stack* yang sesuai kebutuhan tiap fungsi dan bisa berbeda satu sama lain. Setiap fungsi bisa dibangun oleh satu tim tersendiri dengan *code base* tersendiri dan bisa dilakukan testing secara *independent* [**16**]. Gambaran arsitektur *microservices* bisa dilihat pada Gambar 2.2.

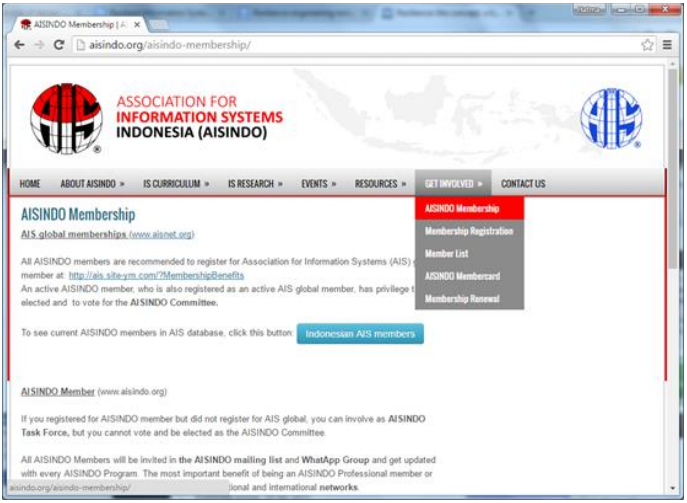


Gambar 2.2. Arsitektur Microservice

Metode *Microservices* di pada penilitian tugas akhir ini, sudah mulai diteliti di lab IKTI sejak tahun 2016 melalui hibah penelitian pemula untuk mengembangkan *software* manajemen asosiasi pada AISINDO (Asosiasi Sistem Informasi Indonesia) berbasis *Microservices* (Gambar 2.3 dan 2.4).



Gambar 2.3. Software hasil penelitian lab IKTI tahun 2015: Open Source manajemen asosiasi/keanggotaan



Gambar 2.4. Software hasil penelitian lab IKTI 2015 telah digunakan untuk manajemen anggota AISINDO

C. Spring Boot

Spring boot merupakan salah satu produk dari komunitas Spring yang bergerak dalam menyediakan *framework* pengembangan aplikasi. Spring boot dikembangkan oleh Pivotal dan diperkenalkan pada tahun 2014. Spring boot dibuat untuk membuat proses pengembangan aplikasi spring agar lebih sederhana. *Framework* ini dikembangkan berdasarkan permintaan dari SPR-988 [**17**]yaitu dengan perbaikan '*containerless'* arsitektur aplikasi web yang artinya pengembangan *container* yang lebih ringan.

Spring *framework* mempunyai beberapa kelebihan. Spring telah mendukung MVC dan menyediakan fitur *RESTful Web Service*. Koneksi *database* juga telah disediakan dalam paket spring. Spring *framework* juga mendukung *Dependency injection* yaitu kemudahan konfigurasi *dependency* dalam aplikasi sehingga dalam proses pengembangan aplikasi menjadi lebih mudah. Setiap spring *framework* juga mendukung *Aspect Object Programming (AOP).* AOP secara sederhana menyediakan fungsi yang dibutuhkan untuk pengembangan aplikasi namun tidak berhubungan dengan fungsi yang ada di aplikasi. Jadi sebagai pengembang aplikasi bisa fokus kepada *core business* dari aplikasi tersebut sedangkan fungsi yang lain seperti masalah keamanan, *logging*, *caching*, *resource pooling*, dan manajemen kinerja aplikasi dibantu dengan AOP [**18**] yang dinamakan teknik *cross-cutting concerns*.

Spring boot sebagai salah satu *framework* spring sudah memiliki beberapa kelebihan yang disebutkan diatas. Kelebihan tambahan yang ada dalam spring boot adalah sudah terdapat sebuah *server* tomcat sehingga hanya perlu dijalankan saja secara langsung [**19**]. Selain itu spring boot juga mendukung maven. Sebuah tool yang akan memudahkan pengembangan aplikasi dengan membuat folder-folder dan *file-file* yang dibutuhkan secara otomatis.

D. Sistem Informasi Aspirasi dan Pengaduan Masyarakat

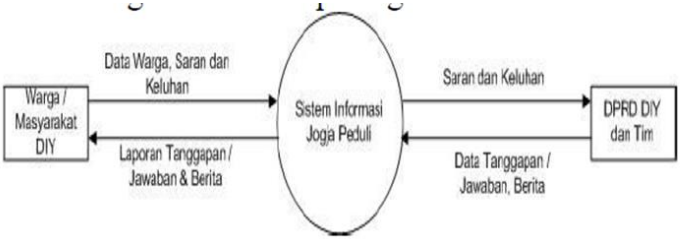
Sistem informasi adalah sebuah kombinasi dari penggunaan teknologi informasi dan aktivitas yang menggunakan teknologi tersebut untuk membantu proses kerja dan manajemen. Penerapan sistem informasi sudah banyak digunakan dalam berbagai bidang. Sistem informasi layanan aspirasi dan pengaduan masyarakat merupakan salah satu jenis produk sistem informasi yang digunakan dalam bidang kepemerintahan. Implementasi jenis ini merupakan implementasi *e-government*.

E-government adalah penyampaian informasi dan layanan dari suatu lembaga pemerintahan terhadap masyarakat, pelaku bisnis dan industri, dan lembaga pemerintahan lainnya melalui penggunaan teknologi informasi dan komunikasi untuk mewujudkan pelaksanaan pemerintahan yang efektif dan efisien, layanan yang lebih baik dan nyaman, mencakup jangkuan yang lebih luas, serta menjamin transparasi dan akuntabilitas [**20**]. Manfaat penggunaan *e-government* adalah meningkatkan kualitas layanan publik kepada masyarakat dan juga perbaikan proses kerja pemerintahan yang lebih efisien dan efektif.

Layanan *e-government* dibagi menjadi beberapa macam layanan. Layanan publikasi, interaksi, dan transaksi. Layanan publikasi merupakan layanan yang sederhana yang memungkinkan komunikasi satu arah dari pemerintah. Sistem informasi aspirasi dan pengaduan masyarakat merupakan bentuk layanan *E-government* interaksi. Layanan ini memungkinkan terjadinya komunikasi dua arah antara pemerintah dan masyarakat. Sedangkan layanan *e-government* transaksi merupakan jenis layanan yang memungkinkan perpindahan uang atau data dari masyarakat kepada pemerintah dan sebaliknya.

Sistem informasi pengaduan masyarakat sudah mulai banyak dikembangkan. Proses bisnis aplikasi dibuat sesuai

dengan analisa kondisi lingkungan yang dibuat oleh *developer*. Aplikasi "Jogja Peduli" oleh (Yanto, 2013) [**21**] untuk penjaringan aspirasi publik terhadap infrastruktur sarana dan prasarana jalan dalam perkotaan Daerah Istimewa Yogyakarta memiliki proses bisnis sebagai yang digambarkan oleh gambar 2.5 dibawah ini.



Gambar 2.5. Diagram konteks sistem informasi daerah terpadu DIY

Warga DIY mencari informasi serta memberi saran dan keluhan melalui *smartphone* pada sistem dan mendapat laporan tanggapan atau jawaban dan berita dari sistem yang ada. DPRD DIY dan tim ahli yang sudah dibentuk untuk menanggapi saran dan keluhan mendapatkan berita laporan tentang saran dan keluhan beserta tanggapan dari tim. Warga DIY mendaftarkan diri ke dalam sistem menyesuaikan dengan KTP sehingga bisa ditanggapi oleh pemerintah DIY. Terdapat bagian berita informasi terbaru seputar DIY di sistem informasi bagi pengunjung. Laporan dari warga akan dikembalikan lagi ke warga sebagai balasan terhadap laporan tersebut.

Proses bisnis aplikasi yang sama juga terdapat pada proyek tesis pembuatan aplikasi sistem informasi pengaduan publik pemerintah kota Bandung oleh (Imawati, 2011) [**22**] yang digambarkan oleh gambar 2.6 dibawah ini

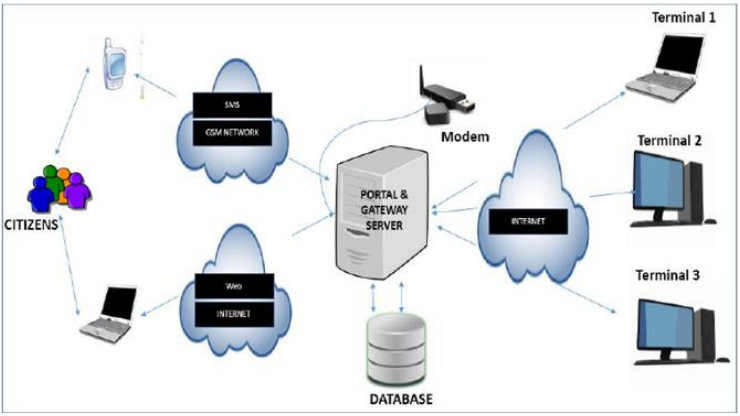
Gambar 2.6. Proses bisnis aplikasi pengaduan Bandung

Kesamaan terletak pada terjadinya komunikasi dua arah, yaitu warga yang melaporkan ke pemerintah daerah dan laporannya akan dikembalikan lagi sebagai timbal balik komunikasi. Proses pengelolaan aduan pada sistem informasi pengaduan publik pemerintah kota Bandung terjadi di suatu *admin*. Laporan akan diteruskan ke satuan kerja perangkat daerah (SKPD) berdasarkan tujuan laporan. SKPD yang akan mengambil langkah dalam menangani aduan berdasarkan laporan tersebut. Proses ini juga melewati walikota sebagai pemberi instruksi.

Membangun sebuah layanan pengaduan memang tidak cukup hanya melewati satu instansi saja. Perlu kerjasama antar semua instansi dan lembaga pemerintah dalam menciptakan pelayanan aduan yang baik. Sehingga proses bisnis yang terjadi dalam aplikasi akan melibatkan banyak SKPD atau instansi pemerintah lainnya sebagai pemeran penting dalam penanganan masalah. Proses bisnis aplikasi aduan seperti tersebut banyak dilakukan seperti dalam aplikasi aduan rakyat *online* Denpasar 2014 [**23**], aplikasi pelaporan warga berbasis *web* dan sms dearah Gihosa, Burundi [**2**], aplikasi LAPOR (lapor.go.id) [**24**], Suara warga Kota Kediri [**25**] dan mungkin masih banyak lagi. Mesikpun media yang digunakan dari tiap aplikasi berbeda (sms, web, aplikasi android), tetapi tujuan aplikasi tetap sama.

Dalam menjalankan proses bisnis aplikasi, maka diperlukan arsitektur sistem yang tepat dalam menopang

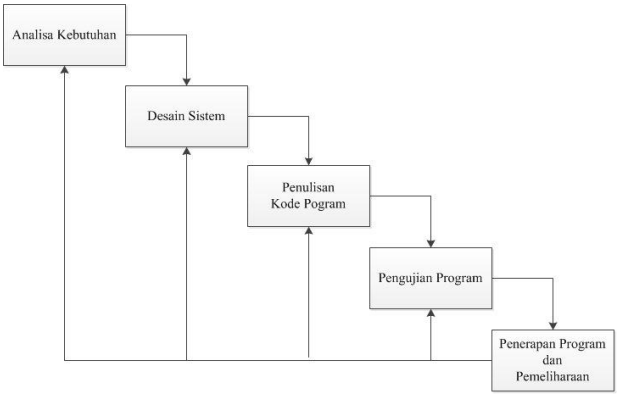
jalannya aplikasi. Aplikasi *e-governement* pengaduan dan aspirasi dipasang dan dikelola sendiri oleh pemerintah setempat. Arsitektur sistem aplikasi *e-government* pengaduan dan aspirasi bisa digambarkan dengan gambar 2.7 dibawah ini



Gambar 2.7. Arsitektur sistem aplikasi pengaduan publik

E. Model Rekayasa Perangkat Lunak Waterfall

Pengembangan perangkat lunak memerlukan suatu model atau metodologi dalam proses pengembangannya. Hal ini disebut juga sebagai Software Development Life Cycle (SDLC). Model atau metodologi tersebut digunakan sebagai petunjuk untuk mengelola bagaimana seharunya perangkat lunak dikembangkan [**26**]. Salah satu model dalam SDLC adalah waterfall. Model waterfall digambarkan secara sederhana seperti gambar 2.8 dibawah ini



Gambar 2.8. Model Waterfall

Metode waterfall merupakan metode dengan pendekatan alur hidup perangkat lunak secara sekuensial dan terurut [**27**] dimulai dari analisa kebutuhan, desain, tahapan implementasi desain (pengkodean), pengujian, dan tahapan pendukung lain yang dikerjakan secara bertahap. Pada pengembangan aplikasi SIAP ini akan digunakan metode waterfall.

F. Unified Modeling Language (UML)

UML adalah sebuah bahasa yang telah menjadi standart dalam industri untuk visualisasi, merancang, dan mendokumentasikan sistem perangkat lunak [**28**]. Pengembangan sistem perangkat lunak lebih baik dilihat dari beberapa sudut pandang untuk mendapatkan gambaran perangkat lunak secara menyeluruh. Oleh karena itu UML menyediakan 9 jenis diagram atau model untuk menggambarkan beberapa sudut pandang sistem perangkat lunak. Berikut adalah daftar 9 diagram tersebut

1. Class Diagram
2. Object Diagram
3. Usecase Diagram
4. Sequence Diagram
5. Collaboration Diagram
6. Statechart Diagram

7. Activity Diagram
8. Component Diagram
9. Deployment Diagram

Dalam membuat UML dibutuhkan tools atau perangkat lunak yang mendukung bahasa UML. Salah satu pengembangan perangkat lunak yang menuntut menggunakan UML dalam proses pengerjaannya adalah ketika membangun perangkat lunak berbasis objek atau Object Oriented Programming (OOP). Penggunaan diagram tidak semuanya harus dibuat. Penggunaan diagram disesuaikan dengan keadaan kebutuhan pengembangan.

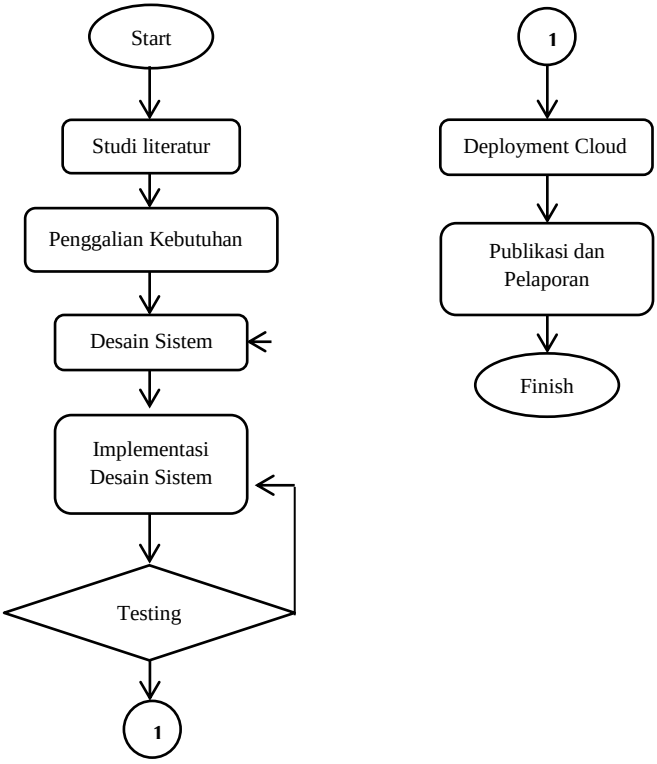
G. Blackbox Testing

Salah satu metode pengujian pada pengembangan aplikasi yang dilakukan adalah blackbox testing. Blackbox testing adalah pengujian yang dilakukan untuk memeriksa apakah fungsional aplikasi berjalan dengan baik dan benar tanpa mengetahui proses yang terjadi di dalam aplikasi [**29**]. Pengujian blackbox dilakukan dengan membuat suatu test case yang berupa test input dan output yang diharapkan dari fungsional aplikasi. Pengujian bisa dilakukan pada aplikasi yang tidak menggunakan algoritma atau tingkat granulitas yang rendah [**30**] sehingga tidak memakan waktu yang banyak [**31**].

Halaman ini sengaja dikosongkan

BAB 3
METODOLOGI

Metodologi merupakan tahapan-tahapan dalam penyelesaian permasalahan pada tugas akhir ini. Metodologi dapat digunakan sebagai panduan pengerjaan tugas akhir agar lebih sistematis, teratur dan terarah. Pembuatan metodologi berdasarkan model rekaya perangkat lunak waterfall. Penggunaan waterfall dikarenakan tahapan pengerjaan yang bertahap dan sistematis. Tahapan-tahapan pengerjaan tugas akhir ini dapat dilihat pada bagan berikut ini:



1. Studi Literatur

Pada tahapan ini dilakukan pengumpulan dan pengkajian pustaka tentang konsep dan metode pengerjaan yang digunakan untuk menyelesaikan permasalahan yang diangkat pada tugas akhir ini. Konsep dan metode yang akan digunakan adalah *microservices*, *docker container*, dan *software as a service* model.

2. Penggalian Kebutuhan

Pada tahap ini akan dilakukan penggalian kebutuhan mengenai sistem informasi yang akan dibuat. Penggalian kebutuhan nantinya akan menjadi *functional requirement* dan *non functional requirement*.

3. Desain Sistem

Pada tahap ini dilakukan perancangan dan desain aplikasi SIAP. Perancagan dan desain menggunakan *Unified Model Language* (UML).

4. Implementasi Desain Sistem

Implementasi desain sistem merupakan tahapan pembuatan kode aplikasi dan kode *database* aplikasi berdasarkan desain yang telah disepakati. Kode aplikasi dibuat dengan menggunakan *framework* Spring.

5. Testing

Pengujian yang akan dilakukan adalah menggunakan pengujian blackbox testing untuk mengetahui jalannya fungsionalitas dalam aplikasi. Blackbox testing juga dilakukan untuk memastikan aplikasi berjalan dengan baik.

6. Deployment Cloud

Aplikasi yang sudah selesai dibangun kemudian akan diletakkan pada server cloud.

7. Publikasi dan Pelaporan

Publikasi dan pelaporan merupakan tahapan akhir dari penelitian ini yang bertujuan untuk mendokumentasikan proses penyelesaian penelitian tugas akhir secara terperinci.

BAB 4
ANALISA KEBUTUHAN DAN PERANCANGAN

Pada bab ini akan dibahas mengenai hal-hal yang terkait dengan kebutuhan dan perancangan aplikasi SIAP.

4.1 Gambaran Umum Sistem

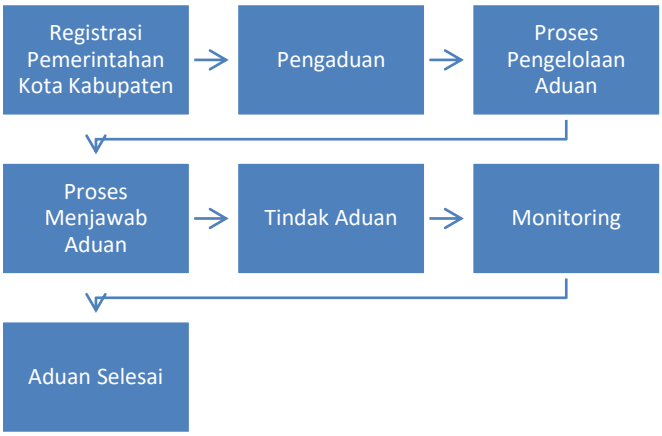
Aplikasi ini bernama "SIAP" yang memiliki kepanjangan "Sistem Informasi Aspirasi dan Pengaduan Masyarakat". Aplikasi SIAP memiliki fungsi secara umum untuk menyampaikan laporan aspirasi, aduan, maupun keluhan masyarakat kepada pemerintah setempat. Sehingga tujuan secara jangka panjang adalah terjadinya komunikasi dua arah antara pemerintah dan masyarakatnya dalam membangun daerahnya terutama dalam membangun fasilitas dan pelayanan daerah yang lebih baik.

Aplikasi SIAP berbasis *web*. Itu berarti proses penyampaian laporan aduan kepada pemerintah melalui media aplikasi web SIAP. Hanya warga yang memiliki nomor identitias yang sesuai dengan daerahnya yang bisa melapor yaitu melalui menggunakan nomor induk kependudukan atau NIK. Penyampaian laporan aduan melalui media web mengharuskan pelapor melakukan input data NIK untuk validasi. Setiap laporan akan memiliki nomor tiket laporan aduan sehingga bisa dilacak status laporannya.

Laporan aduan yang dikirim akan masuk ke dalam sistem aplikasi yang kemudian akan divalidasi dan dikelola oleh pengelola pusat aplikasi yang bisa dinamakan *command center*. Pengelolah laporan aduan di *command center* bisa disebut sebagai Pool. Laporan aduan yang sudah dikelola akan diteruskan ke SKPD terkait dalam pemecahan masalah yang dilaporkan. Laporan aduan akan dikembalikan lagi kepada pengirim sebagai balasan bahwa laporannya telah ditanggapi.

4.2 Gambaran Proses bisnis

Aplikasi SIAP: Sistem informasi aspirasi dan pengaduan masyarakat berbasis web mempunyai proses bisnis yang digambarkan dengan gambar 4.1



Gambar 4.1. Proses Binis Aplikasi SIAP

Aplikasi SIAP dioperasikan dan dikelola dalam suatu unit khusus atau tim tertentu. Dalam beberapa kasus di daerah misalnya Bandung [**32**], aplikasi sejenis ini dioperasikan dan dikelola dalam *command center* sehingga kerjanya lebih maksimal. Dengan menggunakan *command center* maka terdapat suatu pusat terintegrasi baik pusat data dan pusat sistem jalannya proses aduan dan aspirasi warga. Selain itu data aduan yang masuk dalam *command center* bisa dikelolah lebih lanjut dan bisa dipresentasikan untuk kepentingan pemerintah. Hal ini yang menjadi pembeda dari beberapa aplikasi pengaduan yang sejenis.

Aplikasi SIAP merupakan aplikasi berbasis cloud. Pemerintah daerah bisa menyewa layanan aplikasi SIAP sesuai dengan membayar sejumlah uang kepada penyedia aplikasi. Administrator yang ditugaskan pemerintah dalam

mengoperasikan aplikasi SIAP yang telah disewa bisa melakukan upload data kependudukan. Hal ini dibutuhkan untuk memastikan domisili warga yang melapor sesuai dengan daerahnya. Jika tidak sesuai maka laporan akan ditolak.

Penyampaian pengaduan melalui media web. Penyampaian pengaduan melalui media web membutuhkan beberapa data untuk validasi pelapor. Data yang dibutuhkan untuk validasi tersebut adalah data nomor induk kependudukan atau NIK. Warga yang melapor harus memiliki NIK yang valid dan sesuai dengan daerah domisili berdasarkan NIK pada kartu tanda penduduk warga.

Proses pengelolaan aduan dimulai dengan tahapan seleksi aduan. Semua aduan dan aspirasi yang diterima tidak semua ditangani. Terdapat tahapan seleksi laporan oleh sistem. Seleksi yang akan dilakukan yaitu seleksi terhadap konten aduan apabila konten diluar konteks atau aduan yang hanya iseng maka laporan tidak ditanggapi.

Aduan dan aspirasi yang valid dan siap untuk ditangani akan dilakukan klasifikasi sebelum memulai tindak lanjut aduan. Klasifikasi dilakukan oleh pool. Pool berada di *command center*. Pool bertugas untuk mengelompokkan aduan sesuai dengan tema atau badan dan Satuan Kerja Perangkat Daerah (SKPD) yang ditujukan pengadu. Aduan yang telah diklasifikasi akan dikirim dari *command center* menuju SKPD dengan prosedur yang telah disepakati.

Pada SKPD, aduan dan aspirasi yang telah sampai di SKPD akan dikelola lebih lanjut oleh SKPD terkait. Proses pengelolaan data aduan berupa melihat aduan, melakukan perubahan status aduan, menyaring aduan, dan menghapus aduan. Aduan yang tidak sesuai dengan SKPD maka akan dikembalikan lagi kepada pool.

Tindak aduan merupakan tindak lanjut nyata dari aduan yang diterima SKPD. Tindakan ini berupa tindak nyata di

lapangan atau jawaban pertanyaan yang ada di dalam sistem SIAP.

Proses selanjutnya adalah monitoring status penanganan aduan. Monitoring bisa dilakukan oleh masyarakat sebagai pihak pengadu dalam melihat status aduan yang dikirim apakah sudah ditangani atau belum. Monitoring juga dilakukan oleh kepala daerah sebagai pengawas yang dapat melihat semua aduan. Jika penanganan aduan lambat maka kepala daerah bisa memberikan teguran kepada pihak yang bertanggung jawab.

Jika laporan aduan benar-benar ditanggapi atau ditangani di lapangan, maka warga sebagai pihak pengadu bisa mengubah status aduan menjadi selesai untuk memberitahu pemerintahan bahwa aduan tersebut telah sepenuhnya ditangani.

4.1 Analisa Penggalian Kebutuhan

Tahapan sebelum pengkodean aplikasi SIAP adalah dilakukan analisa penggalian kebutuhan sistem. Analisa ini berguna untuk mendapatkan gambaran lebih detail mengenai aplikasi SIAP. Analisa penggalian kebutuhan meliputi analisa kebutuhan aktor, analisa kebutuhan fungsional, analisa kebutuhan microservice, dan analisa kebutuhan non fungsional aplikasi.

Analisa penggalian kebutuhan aplikasi SIAP dilakukan dengan melihat beberapa aplikasi sejenis. Pengamatan pertama yang dilakukan adalah dengan melihat aplikasi SURGA (Suara warga) kota Kediri. Penggalian kebutuhan dilakukan dengan melihat dokumen perancangan aplikasi berbasis web tersebut. Pengamatan selanjutnya yang dilakukan adalah dengan melihat aplikasi pelaporan di web seperti aplikasi LAPOR di situsnya www.lapor.go.id. Selain itu pengamatan dilakukan dari studi literatur sehingga menemukan beberapa informasi terkait kebutuhan aplikasi e-government pelaporan publik masyarakat. Selain itu juga dilakukan pengamatan dengan

studi literatur terhadap aplikasi software as a service dan aplikasi penyewaan yang mencatat informasi dari konsumen aplikasi tersebut. Dari semua proses tersebut akhirnya didapatkan beberapa kebutuhan aplikasi SIAP.

4.3.1 Analisa Kebutuhan Aktor

Aktor adalah pengguna yang akan berhubungan dengan jalannya proses bisnis aplikasi. Berikut adalah tabel 4.1 dibawah ini yang akan menjelaskan beberapa user dalam proses bisnis aplikasi SIAP.

Tabel 4.1. Analisa Kebutuhan Aktor

No	Aktor	Deskripsi	Sumber
1	Administrator	Warga atau masyarakat adalah orang-orang yang bertempat tinggal atau berdomisili di suatu daerah tertentu yang mempunyai hak untuk menggunakan aplikasi sesuai dengan domisili daerahnya.	SURGA (Suara Warga) Kota Kediri
2	Administrator (Admin)	Administrator atau admin adalah orang-orang yang merupakan perwakilan dari pemerintah daerah tertentu dalam mengelola aplikasi agar bisa digunakan sesuai dengan kepentingan daerahnya. Administrator bekerja dalam tim di command center	SURGA (Suara Warga) Kota Kediri

3	Pool	Pool adalah orang-orang yang merupakan perwakilan dari pemerintah daerah tertentu dalam mengelola laporan aduan. Pool akan melakukan validasi laporan aduan, klasifikasi sehingga laporan bisa ditanggapi dengan tepat.	SURGA (Suara Warga) Kota Kediri
4	Satuan Kerja Perangkat Daerah (SKPD)	SKPD merupakan perangkat daerah dalam menjalankan fungsi eksekutif agar pemerintahan berjalan dengan baik. Pada aplikasi ini SKPD diwaklikan oleh satu atau lebih orang untuk terlibat dalam proses bisnis aplikasi SIAP. SKPD akan bertindak dalam menyelesaikan masalah yang dilaporkan warga.	SURGA (Suara Warga) Kota Kediri
5	Vendor	Vendor adalah penyedia aplikasi yang bertugas dalam menjaga ketersediaan aplikasi sehingga bisa digunakan sesuai dengan kesepakatan.	
6	Supervisor	Supervisor atau Kepala pemerintahan	SURGA (Suara

	bisa berupa walikota, bupati, dan sejenisnya yang akan memantau laporan aduan dan melihat statistika aduan.	Warga) Kota Kediri

4.3.2 Analisa Kebutuhan Fungsional

Berdasarkan proses bisnis dan analisa aktor yang telah dibahas sebelumnya maka perlu dilakukan analisa kebutuhan fungsional untuk aplikasi SIAP. Kebutuhan fungsional akan dijelaskan pada tabel 2 dibawah ini

4.3.2.1 Kebutuhan Fungsional Administrator

Kebutuhan fungsional administrator adalah seperti pada tabel 4.2 dibawah ini

Tabel 4.2. Kebutuhan Fungsional Administrator

IDKF	Deskripsi	Sumber
KF1	Melakukan registrasi Customer	Niaga Hoster[**33**]
KF2	Login	SURGA (Suara Warga) Kota Kediri
KF3	Manajemen data kependudukan	Perancangan dan pembuatan aplikasi Sistem informasi desa (SIMDA) Desa Ngemplak Sukoharjo [**34**]
KF4	Manajemen Daftar Departement	SURGA (Suara Warga) Kota Kediri
KF5	Manajemen user petugas	SURGA (Suara

		Warga) Kota Kediri
KF6	Manajemen kategori aduan	SURGA (Suara Warga) Kota Kediri
KF7	Melihat tagihan	Niaga Hoster [**33**]
KF8	Melihat Profile	Niaga Hoster [**33**]
KF9	Logout	SURGA (Suara Warga) Kota Kediri
KF10	Reset Password	SURGA (Suara Warga) Kota Kediri

4.3.2.2 Kebutuhan Fungsional Pool

Kebutuhan fungsional pool adalah seperti pada tabel 4.3 dibawah ini

Tabel 4.3. Kebutuhan Fungsional Pool

IDKF	Deskripsi	Sumber
KF11	Login	SURGA (Suara Warga) Kota Kediri
KF12	Logout	SURGA (Suara Warga) Kota Kediri
KF13	Menerima aduan	SURGA (Suara Warga) Kota Kediri
KF14	Mengelompokkan aduan	SURGA (Suara Warga) Kota Kediri

KF15	Mengirim aduan ke SKPD	SURGA (Suara Warga) Kota Kediri

4.3.2.3 Kebutuhan Fungsional SKPD

Kebutuhan fungsional SKPD adalah seperti pada tabel 4.4 dibawah ini

Tabel 4.4. Kebutuhan Fungsional SKPD

IDKF	Deskripsi	Sumber
KF16	Login	SURGA (Suara Warga) Kota Kediri
KF17	Logout	SURGA (Suara Warga) Kota Kediri
KF18	Menjawab aduan	SURGA (Suara Warga) Kota Kediri
KF19	Mengembalikan aduan	SURGA (Suara Warga) Kota Kediri

4.3.2.4 Kebutuhan Fungsional Supervisor

Kebutuhan fungsional supervisor adalah seperti pada tabel dibawah ini

Tabel 4.5. Kebutuhan Fungsional Supervisor

IDKF	Deskripsi	Sumber
KF20	Login	SURGA (Suara Warga) Kota

		Kediri
KF21	Logout	SURGA (Suara Warga) Kota Kediri
KF22	Rekap Aduan Belum Selesai	SURGA (Suara Warga) Kota Kediri
KF23	Rekap Aduan Telah Selesai	SURGA (Suara Warga) Kota Kediri

4.3.2.5 Kebutuhan Fungsional Warga

Kebutuhan fungsional warga adalah seperti pada tabel dibawah ini

Tabel 4.6. Kebutuhan Fungsional Warga

IDKF	Deskripsi	Sumber
KF24	Melaporkan aduan	SURGA (Suara Warga) Kota Kediri
KF25	Melakukan verifikasi NIK	SURGA (Suara Warga) Kota Kediri
KF26	Menerima nomor tiket aduan	SURGA (Suara Warga) Kota Kediri
KF27	Melihat status aduan	SURGA (Suara Warga) Kota Kediri
KF28	Konfirmasi Aduan Selesai	Sistem

		Informasi Pengaduan Masyarakat Pada Dewan Pers Indonesia [**35**]

4.3.2.6 Kebutuhan Fungsional Vendor

Kebutuhan fungsional vendor adalah seperti pada tabel dibawah ini

Tabel 4.7. Kebutuhan Fungsional Vendor

IDKF	Deskripsi	Sumber
KF29	Login	SURGA (Suara Warga) Kota Kediri
KF30	Logout	SURGA (Suara Warga) Kota Kediri
KF31	Melihat Penyewa	Rancang Bangun Sistem Aplikasi Penyewaan Lapangan Futsal Berbasis Web [**36**]
KF32	Manajemen daftar paket	Rancang Bangun Sistem Informasi Akademik Mahasiswa

		Berbasis Web Pada "Akbid Griya Husada" Surabaya [**37**]

4.3.3 Analisa Kebutuhan Microservice

Dalam melakukan analisa kebutuhan fungsional, tahapan yang digunakan adalah menggunakan tahapan yang diadopsi dari [**11**] yaitu setelah mentukan kebutuhan fungsional maka tiap kebutuhan fungsional tersebut akan menjadi batasan konteks untuk menentukan microservice yang saling berhubungan dengan kebutuhan fungisonal tersebut. Pembagian kebutuhan microservice akan dibagi berdasarkan user yang ada di aplikasi SIAP.

4.3.3.1 Pemecahan Microservice User Administrator

Kebutuhan fungsional pada administrator bisa dibecah menjadi beberapa microservice seperti pada tabel 4.8 dibawah ini

Tabel 4.8. Pemecahan Microservice Administrator

Kebutuhan Fungsional	Kebutuhan Microservice
Melakukan registrasi Customer	Membuat customer
Login	Login customer
Manajemen data kependudukan	Mengambil data penduduk
	Upload data penduduk
	Menghapus data penduduk
Manajemen Daftar Departement	Mengambil data daftar departemen
	Membuat data departemen
	Menghapus data departemen
Manajemen user petugas	Membuat user petugas
	Mengambil data daftar user

	petugas
	Menghapus user petugas
Manajemen kategori aduan	Membuat data kategori
	Mengambil data daftar kategori
	Menghapus data kategori
Melihat tagihan	Menampilkan data tagihan
Melihat Profile	Menampilkan data customer
Reset Password	Mengambil data username customer
	Memperbarui password customer

4.3.3.2 Pemecahan Microservice User Pool

Kebutuhan fungsional pada user pool bisa dipecah dalam beberapa microservice yang mendukung yaitu seperti pada tabel 4.9 dibawah ini

Tabel 4.9. Pemecahan Microservice Pool

Kebutuhan Fungsional	Kebutuhan Microservice
Login	Login User Petugas
Menerima aduan	Memperbarui status aduan
Mengelompokkan aduan	Memperbarui nama kategori data aduan
Mengirim aduan ke SKPD terkait	Memperbarui nama departemen data aduan
	Memperbarui status aduan
	Mengirim aduan ke SKPD

4.3.3.3 Pemecahan Microservice User SKPD

Kebutuhan fungsional user SKPD dapat dipecah menjadi beberapa kebutuhan microservice yang mendukung yaitu seperti pada tabel 4.10 dibawah ini

Tabel 4.10. Pemecahan Microservice SKPD

Kebutuhan Fungsional	Kebutuhan Microservice

Login	Login User Petugas
Menjawab aduan	Memperbarui jawaban aduan
	Memperbarui status aduan
Mengembalikan aduan	Memperbarui nama departemen aduan
	Memperbarui status aduan
	Memperbarui note aduan

4.3.3.4 Pemecahan Microservice User Supervisor

Kebutuhan fungsional user supervisor dapat dipecah menjadi beberapa kebutuhan microservice yang mendukung yaitu seperti pada tabel 4.11 dibawah ini

Tabel 4.11. Pemecahan Microservice Supervisor

Kebutuhan Fungsional	Kebutuhan Microservice
Login	Login User Petugas
Rekap Aduan Belum Selesai	Mengambil data daftar aduan dengan status selesai
Rekap Aduan Telah Selesai	Mengambil data daftar aduan dengan semua status kecuali status selesai

4.3.3.5 Pemecahan Microservice User Warga

Kebutuhan fungsional user warga dapat dipecah menjadi beberapa kebutuhan microservice yang mendukung yaitu seperti pada tabel 4.12 dibawah ini

Tabel 4.12. Pemecahan Microservice Warga

Kebutuhan Fungsional	Kebutuhan Microservice
Melaporkan aduan	Membuat aduan
Melakukan verifikasi NIK	Mengambil data NIK penduduk
	Mengambil data nomor kota kabupaten customer

Melihat status aduan	Mengambil data aduan berdasarkan nomor tiket
Konfirmasi Aduan Selesai	Memperbarui status aduan menjadi selesai

4.3.3.6 Pemecahan Microservice User Vendor

Kebutuhan fungsional user vendor dapat dipecah menjadi beberapa kebutuhan microservice yang mendukung yaitu seperti pada tabel 4.13 dibawah ini

Tabel 4.13. Pemecahan Microservice Vendor

Kebutuhan Fungsional	Kebutuhan Microservice
Login	Login Vendor
Melihat Penyewa	Menampilkan data daftar customer
Manajemen daftar paket	Membuat data paket
	Menghapus data paket
	Mengambil data daftar paket

4.3.4 Analisa Kebutuhan Non Fungsional

Berikut adalah beberapa kebutuhan non fungsional yang dibutuhkan aplikasi yang akan dijabarkan dengan tabel 4.14 dibawah ini.

Tabel 4.14. Kebutuhan Non Fungsional

Kebutuhan Non Fungsional (KNF)		
IDKNF	**Deskripsi**	**Tool**
KNF01	Aplikasi telah berjalan sesuai dengan kebutuhan yang telah dispesifikasikan	Pengujian dilakukan berdasarkan uji fungsionalitas sistem dengan menggunakan *Blackbox Testing*

4.4 Perancangan Sistem

Tahapan setelah melakukan analisa kebutuhan fungsional dan kebutuhan non fungsional adalah melakukan desain sistem. Kebutuhan fungsional harus di verifikasi terlebih dahulu sehingga tidak ada perubahan permintaan ketika tahapan pengembangan. Setelah disepakati maka dimulai tahap desain sistem untuk memberikan gambaran yang lebih lengkap mengenai aplikasi SIAP. Pada tahap ini dilakukan beberapa aktifitas yaitu membuat use case diagram, membuat use case story, membuat sequence diagram, perancangan *graphical user interface* dari aplikasi SIAP, dan menggambarkan arsitektur dari aplikasi.

4.4.1 Perancangan Use Case Aplikasi

Desain *use case* dibuat dari hasil analisa kebutuhan dan bentuk desain yang ada pada aplikasi SIAP. Use case ini menunjukan interaksi antara aktor atau penguna aplikai dengan aplikasi. Selain pembuatan desain, juga dilakukan pembuatan deskripsi dari masing-masing *use case* yang ada. Deskripsi use case (*Use Case Description*) berisikan detail *behavior* dari sebuah use case. Pada lampiran A terdapat desain use case aplikasi beserta deskripsi use case.

4.4.2 Perancangan Test Case

Test Case merupakan tahap pengujian pada setiap *use case* yang telah dibuat. Test Case dibuat setelah aplikasi selesai dibuat. Tujuan dari pembuatan *test case* adalah untuk mengetahui apakah sistem sudah berjalan sesuai dengan desain yang telah dibuat. Test Case yang dibuat berdasarkan metode pengujian black-box testing. Setiap test case mempunyai nomor pengujian dan skenario-skenario pengujian tertentu. Perancangan test case terdapat pada lampiran D.

BAB 5
IMPLEMENTASI DAN UJI COBA

Pada bab ini diuraikan hal-hal terkait implementasi dan uji coba Aplikasi SIAP. Teknologi yang diimplementasikan dalam sistem ini meliputi bahasa pemrograman java dengan *framework* Springboot sebagai *back-end* aplikasi SIAP, basis data MYSQL, dan menggunakan Bahasa javascript/typescript dengan framework Angular sebagai *front-end* aplikasi SIAP (opsional). Implementasi akan dipaparkan dengan menggunakan potongan gambar dari *user interface* dan potongan kode program. Uji coba sistem meliputi tes terhadap beberapa *usecase* yang telah dibuat pada sebelumnya.

5.1 Linkungan Implementasi

Aplikasi SIAP dikembangkan menggunakan perangkat keras laptop/PC. Spesifikasi perangkat keras yang digunakan dalam pengembangan aplikasi dapat dilhat pada tabel.

Perangkat lunak utama yang digunakan sebagai *editor* adalah Netbeans pada back-end dan Atom pada front-end. Web server menggunakan XAMPP 3.0.12, dengan basis data MYSQL. Tabel 5.1 dibawah menunjukkan spesifikasi perangkat keras yang digunakan pada tahap pengembangan dan tabel 5.2 menunjukkan perangkat lunak yang digunakan pada tahap pengembangan aplikasi SIAP

Tabel 5.1 Spesifikasi Perangkat Keras

Perangkat Keras	Spesifikasi	
PC/Laptop	Prosesor	AMD A10 CPU@2.5GHz
	RAM	6000 MB

Tabel 5.2 Analisa kebutuhan Perangkat Lunak

Perangkat Lunak / Tools	Versi
Sistem Operasi	Windows 10
Web Server	Apache 2.4.25 Tomcat 7.0.56
Basis Data	MYSQL 5.0.10
Bahasa Pemrograman	Java, Javascript, HTML5
Back-end Framework	Springboot
Front-end Framework	Angular v2
Editor	Netbeans Atom

5.2 Implementasi Desain Aplikasi SIAP

Aplikasi SIAP mengimplementasikan teknologi website untuk setiap fitur yang disediakan bagi setiap kategori user yang ada. Pada sub bab kali ini akan dijelaskan mengenai masing-masing implementasinya baik dibagian *front-end* maupun *back-end.*

5.2.1 Implementasi Registrasi Customer

Proses bisnis aplikasi SIAP dimulai dengan melakukan registrasi oleh pemerintahan. Pada tampilan registrasi pemerintahan menampilkan sebuah form seperti gambar 5.1 dibawah ini

Gambar 5.1. Tampilan Registrasi

Untuk membuat form registrasi maka perlu kode front-end pada angular seperti gambar potongan kode 5.2 berikut ini

```
ngOnInit() {
  this.getprovince();
  this.getpacketlist();
  this.register = this.fb.group({
    name: ['', [Validators.required]],
    idprovinsi: ['', [Validators.required]],
    idkokab: ['', [Validators.required]],
    address: ['', [Validators.required]],
    email: ['', [Validators.required]],
    phone: ['', [Validators.required]],
    idpacket: ['', [Validators.required]],
    username: ['', [Validators.required]],
    password: ['', [Validators.required]],
    role: ['ADMIN']
  });
}
```

Gambar 5.2. Potongan kode front-end registrasi

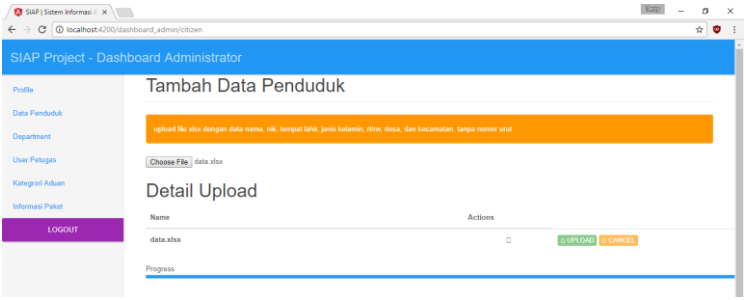
Kebutuhan fungsional registrasi membutuhkan microserive membuat customer baru dengan kode program java spring boot seperti pada potongan kode program 5.3 dibawah ini

```
@RestController
public class CustomerController {

	@Autowired
	private CustomerService customerservice;

	//register customer
	@RequestMapping(value="/customer/new", method=RequestMethod.POST)
	public Customer newCustomer (@RequestBody Customer customer){
	    customer.setPassword(new BCryptPasswordEncoder().encode(customer.getPassword()));
	        return customerservice.saveCustomer(customer);
	}
```

Gambar 5.3. Potongan kode microservice fungsional registrasi customer

5.2.2 Implementasi Manajemen Data Kependudukan

Implementasi pembuatan fungsi menunggah data kependudukan membutuhkan halaman front-end yang menampilkan sebuah tabel dan tombol untuk mengunggah seperti gambar 5.4 dibawah ini



Gambar 5.4. Tampilan Manajemen Data Kependudukan

Untuk membuat tampilan fungsi mengunggah dokumen data kependudukan maka dibutuhkan kode program front-end seperti potongan kode program 5.5 dibawah ini

```
public uploader:FileUploader = new FileUploader({url: URL});
public hasBaseDropZoneOver:boolean = false;
public hasAnotherDropZoneOver:boolean = false;

public fileOverBase(e:any):void {
  this.hasBaseDropZoneOver = e;
}
```

Gambar 5.5. Potongan kode front-end manajemen data kependudukan

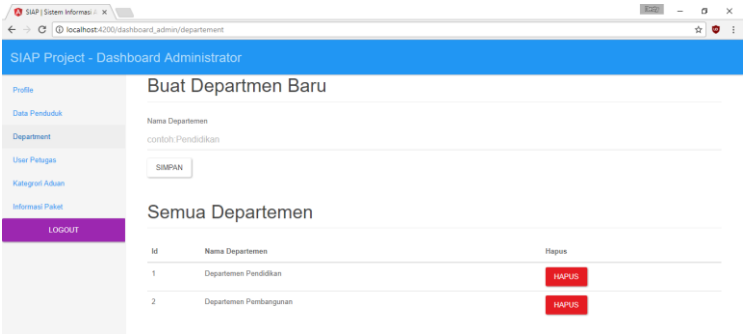
Manajemen data kependudukan membutuhkan microservice untuk membuat data kependudukan, menghapus data kependudukan, dan mengambil data daftar penduduk. Dalam membuat data penduduk metode yang digunakan adalah dengan mengunggah dokumen excel dengan kode program java springboot seperti pada potongan kode program 5.6 dibawah ini

```
public String uploadExcel (@RequestPart("file") MultipartFile file){
    try {
    int i = 0;
    //Creates a workbook object from the uploaded excelfile
    XSSFWorkbook workbook = new XSSFWorkbook(file.getInputStream());
    //Creates a worksheet object representing the first sheet
    XSSFSheet worksheet = workbook.getSheetAt(0);
    //Reads the data in excel file until last row is encountered
    while (i <= worksheet.getLastRowNum()) {
        //Creates an object for the Candidate  Model
        Citizen citizen=new Citizen();
        //Creates an object representing a single row in excel
        XSSFRow row = worksheet.getRow(i++);
        //Sets the Read data to the model class
        citizen.setNik(row.getCell(0).getStringCellValue());
        citizen.setName(row.getCell(1).getStringCellValue());
        citizen.setTempatttl(row.getCell(2).getStringCellValue());
        citizen.setSex(row.getCell(3).getStringCellValue());
        citizen.setAlamat(row.getCell(4).getStringCellValue());
        citizen.setRtrw(row.getCell(5).getStringCellValue());
        citizen.setDesa(row.getCell(6).getStringCellValue());
        citizen.setKecamatan(row.getCell(7).getStringCellValue());
        //Sends the model object to service layer for validation,
        //data processing and then to persist
        citizenservice.addcitizen(citizen);
    }
} catch (Exception e) {
    e.printStackTrace();
}

return "uploadSuccess";
}
```

Gambar 5.6. Potongan kode microservice fungsional manajemen data kependudukan

5.2.3 Implementasi Manajemen Daftar Departement

Implementasi manajemen daftar departemen membutuhkan suatu tabel dan form seperti pada gambar 5.7 dibawah ini



Gambar 5.7. Tampilan Manajemen Daftar Department

Untuk membuat fungsi tersebut maka diperlukan kode program pada front-end seperti pada gambar potongan kode program 5.8 dibawah ini

```
ngOnInit() {
    this.updatedept();
    this.f = this.fb.group({
    departmentname:['',[Validators.required]],
    idkokab:[this.idkokab]
  });
}
```

Gambar 5.8. Potongan kode front-end manajemen department

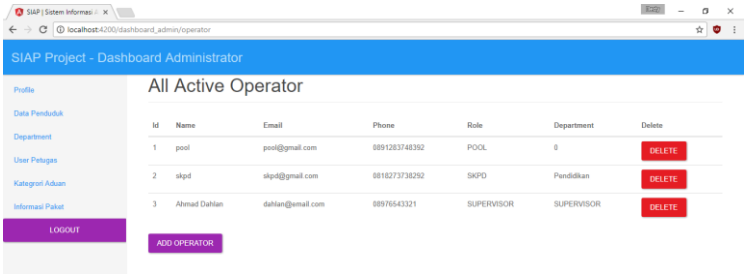
Manajemen daftar departemen membutuhkan *microservice* untuk membuat data departemen baru, menampilkan daftar departemen yang telah dibuat, dan menghapus data departemen. Berikut adalah gambar 5.9 potongan kode *microservice-microservice* tersebut

```
@Autowired
private DepartmentService departmentservice;

@RequestMapping(value="/new", method=RequestMethod.POST)
public Department addDepartment(@RequestBody Department department){
        return departmentservice.addDepartment(department);
}

@RequestMapping(value="/all/{idkokab}", method=RequestMethod.GET)
public List <Department> findAllDepartment(@PathVariable String idkokab){
        return departmentservice.findByidkokab(idkokab);
};

@RequestMapping(value="/delete/{departmentid}", method=RequestMethod.DELETE)
public void deleteDepartmentById (@PathVariable Long departmentid){
        departmentservice.deleteDepartmentById(departmentid);
};
```

Gambar 5.9. Potongan kode microservice fungsional manajemen daftar departemen

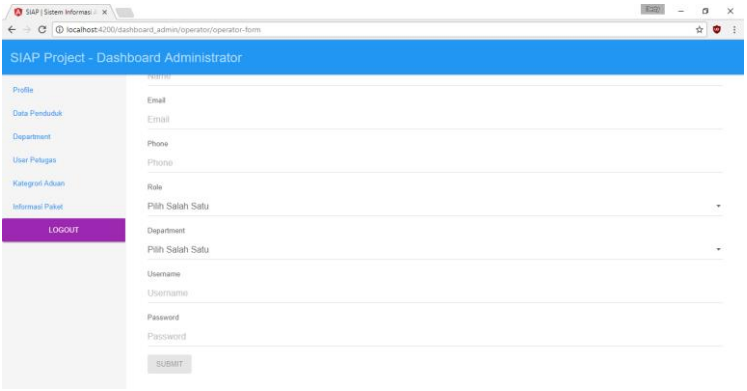
5.2.4 Implementasi Manajemen User Petugas

Implementasi manajemen user petugas membutuhkan sebuah tampilan tabel untuk menampilkan semua akun user petugas yang telah dibuat dan sebuah form untuk membuat data user petugas. Tampilan tabel unruk menampilkan user petugas tersebut digambarkan seperti gambar 5.10 dibawah ini



Gambar 5.10. Tampilan user petugas

Sedangkan tampilan untuk menambah user petugas yaitu sebuah form dapat digambarkan dengan gambar 5.11 dibawah ini



Gambar 5.11. Tampilan form user petugas

Untuk membuat fungsi tersebut maka dibutuhkan kode program front-end seperti pada gambar potongan kode program 5.12 dibawah ini

```
ngOnInit() {
    this.updatedept();
    this.f = this.fb.group({
    name:['',[Validators.required]],
    email:['',[Validators.required]],
    phone:['',[Validators.required]],
    role:['',[Validators.required]],
    department:['',[Validators.required]],
    idkokab:[this.idkokab],
    username:['',[Validators.required]],
    password:['',[Validators.required]],
  });
}
```

Gambar 5.12. Potongan kode front-end manajemen user petugas

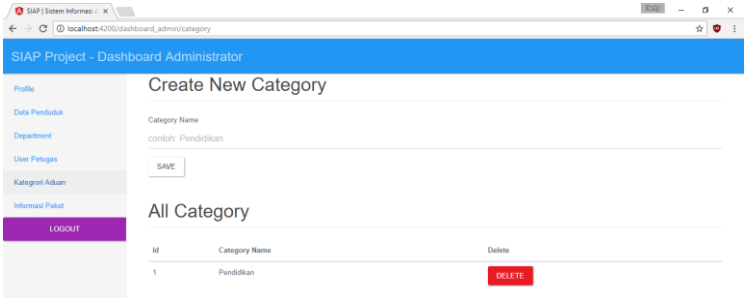
Manajemen user petugas membutuhkan microservice untuk membuat user petugas baru, mengambil data daftar user petugas yang telah dibuat, dan menghapus data user petugas. Berikut adalah salah satu gambar potongan kode program java springboot 5.13 microservice membuat user petugas baru

```
@Autowired
private OperatorService operatorservice;

//digunakan untuk registrasi operator baru
@RequestMapping(value="/rest/admin/operator/new", method=RequestMethod.POST)
public Operator addOperator (@RequestBody Operator operator){
        operator.setPassword(new BCryptPasswordEncoder().encode(operator.getPassword()));
        return operatorservice.addOperator(operator);
}
```

Gambar 5.13. Potongan kode microservice fungsional manajemen user petugas

5.2.5 Implementasi Manajemen Kategori Aduan

Dalam membuat fungsi manajemen kategoru aduan diperlukan input untuk menambah nama kategori dan sebuah tabel untuk menampilkan semua kategori yang telah dibuat seperti pada gambar 5.14 dibawah ini



Gambar 5.14. Tampilan Manajemen Kategori Aduan

Untuk membuat fungsi tersebut maka diperlukan kode program front-end seperti pada gambar potongan kode program 5.15 dibawah ini

```
ngOnInit() {
  this.updatecategory();
  this.f = this.fb.group({
  categoryname:['',[Validators.required]],
  idkokab:[this.idkokabe]
  });
}
```

Gambar 5.15. Potongan kode front-end manajemen kategori aduan

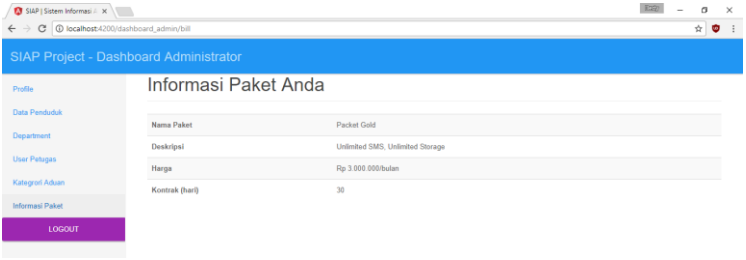
Fungsional manajemen kategori aduan membutuhkan microservice membuat kategori baru, mengambil daftar kategori yang telah dibuat, dan menghapus data kategori. Berikut adalah gambar potongan kode program 5.16 microservice pada fungsional manajemen kategori aduan

```
@Autowired
private CategoryService categoryservice;

@RequestMapping(value="/new", method=RequestMethod.POST)
public Category addCategory(@RequestBody Category category){
        return categoryservice.addCategory(category);
}

@RequestMapping(value="/all", method=RequestMethod.GET)
public List<Category> findAllCategory(){
        return categoryservice.findAllCategory();
}
```

Gambar 5.16. Potongan kode microservice fungsional manajemen kategori aduan

5.2.6 Implementasi Melihat Tagihan

Fungsi melihat tagihan membutuhkan suatu tabel untuk menampilkan tagihan atau paket yang dipilih oleh pemerintahan tersebut seperti pada gambar 5.17 dibawah ini



Gambar 5.17. Tampilan melihat tagihan

Dalam membuat tampilan seperti gambar diatas maka dibutuhkan kode pemrograman front-end seperti pada gambar potongan kode 5.18 dibawah ini

```
ngOnInit() {
  this.service.getonepacket(this.nopacket).subscribe(
    data => {
        this.packetdata = data;
    },
    error => console.log(this.errorMessage = <any>error)
  )
}
```

Gambar 5.18. Potongan kode front-end melihat tagihan

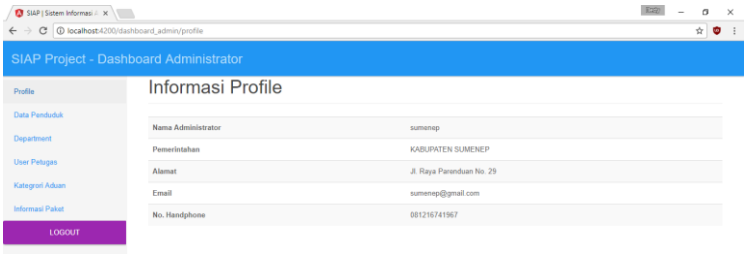
Fungsionalitas melihat tagihan membutuhkan microservice mengambil data tagihan sesuai dengan customer. Berikut adalah gambar potongan kode program 5.19 microservice fungsional melihat taghian

```
@RequestMapping(value="user-dev/packet/all", method=RequestMethod.GET)
public List<Packet> findAllPacketName(){
        return packetnameservice.findAllPacketName();

}

@RequestMapping(value="/rest/user-dev/packet/packet/{idpacket}", method=RequestMethod.GET)
public List<Packet> findById(@PathVariable Long idpacket){
        return packetnameservice.findByidpacket(idpacket);

}
```

Gambar 5.19. Potongan kode microservice fungsional melihat tagihan

5.2.7 Implementasi Melihat Profile

Fungsi melihat profile membutuhkan suatu tabel untuk menampilkan data profile pemerintahan pada saat melakukan registrasi. Tampilan fungsi melihat profile seperti pada gambar 5.20 dibawah ini



Gambar 5.20. Tampilan melihat profile

Untuk membuat fungsi melihat profile maka diperlukan kode program front-end seperti pada gambar potongan kode program 5.21 dibawah ini

```
kokabname(){
  this.service.getkokabname(this.gov).subscribe(
    data=> {
      this.governmentname = JSON.parse(JSON.stringify(data))._body;
    },
    err => err
  )
}
```

Gambar 5.21. Potongan kode front-end melihat profile

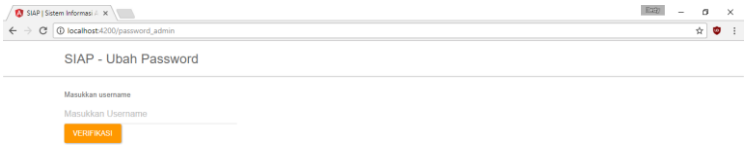
Fungsi melihat profile membutuhkan microservice mengambil data registrasi customer yang dikirim bersamaan dengan token pada saat login. Potongan kode microservice tersebut adalah seperti pada gambar 5.22 dibawah ini

```
    String pwd = customer.getPassword();

    if(!new BCryptPasswordEncoder().matches(password, pwd)){
        throw new ServletException("Invalid login. Please check your name and password.");
    }

    /*
    if (!password.equals(pwd)) {
        throw new ServletException("Invalid login. Please check your name and password.");
    } */

    jwtToken = Jwts.builder().setSubject(username).claim("roles", "administrator").setIssuedAt(new
            .signWith(SignatureAlgorithm.HS256, "ZGVkeXBlamk=").compact()+ ";" + customer.getName(

    return jwtToken;
}
```

Gambar 5.22. Potongan kode microservice fungsional melihat profile

5.2.8 Implementasi Reset Password

Untuk membuat fungsi reset password dibutuhkan halaman untuk melakukan verifikasi username dan halaman untuk mengubah password seperti pada gambar 5.23 dibawah ini



Gambar 5.23. Tampilan reset password

Untuk membuat tampilan reset password maka dibutuhkan kode pemrograman front-end seperti pada gambar potongan kode program 5.24 dibawah ini

```
resetPassword(password){
  let confirmmsg;
  let r = confirm("Yakin Reset Password");
  if (r == true) {
    this.service.resetPassword(this.id,password.value).subscribe(
      data=>{
        alert("Password Berhasil direset");
        this.route.navigate(['home']);
      }
    )
  } else {
    alert("Password tidak direset");
  }

}
```

Gambar 5.24. Potongan kode front-end reset password

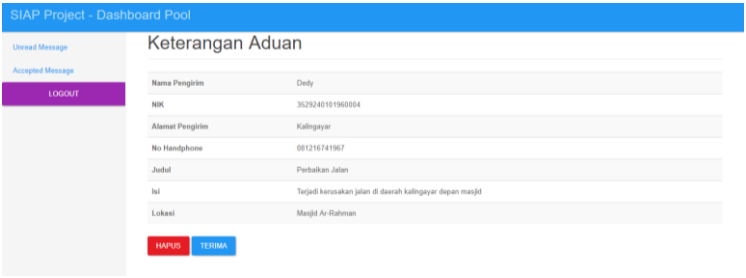
Fungsional reset password dibangun dengan microservice memperbarui password dengan melakukan pengecekan user menggunakan mengambil data username. Potongan kode microservice tersebut adalah seperti pada gambar 5.25 dibawah ini

```
//digunakan untuk reset password
@RequestMapping(value="/customer/reset/password/{id}", method=RequestMethod.PUT)
public void resetpassword (@PathVariable Long id, @RequestBody Customer customer){
    Customer update = customerservice.findByid(id);
    update.setPassword(new BCryptPasswordEncoder().encode(customer.getPassword()));
    customerservice.saveCustomer(update);
}
```

Gambar 5.25. Potongan kode microservice fungsional reset password

5.2.9 Implementasi Menerima Aduan

Fungsi menerima aduan dilakukan oleh pool sebagai petugas di command center untuk melihat semua aduan dan menerima aduan yang valid. Berikut adalah gambar 5.26 yang merupakan tampilan fungsi menerima aduan



Gambar 5.26. Tampilan menerima aduan

Untuk membuat fungsi menerima aduan maka dibutuhkan kode pemrograman front-end seperti pada gambar potongan kode program 5.27 dibawah ini

```
acceptcomplaint(){
  this.service.acceptComplaint(this.idcomplaint,"READ").subscribe(
    ()=>alert("Complaint accepted, check Accepted Message Menu")
  )
  this.router.navigate(["../../"], {relativeTo: this.route});
}
```

Gambar 5.27. Potongan kode front-end menerima aduan

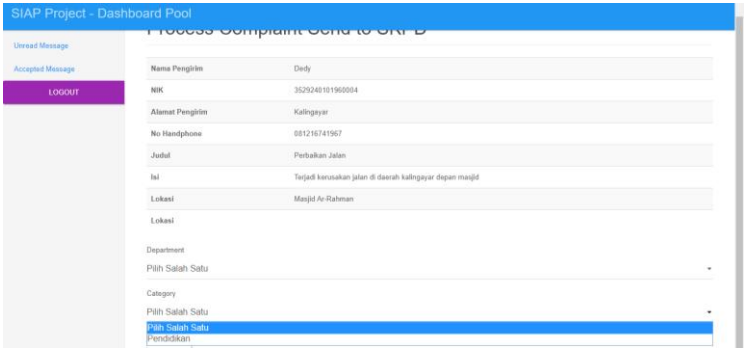
Fungsi menerima aduan membutuhkan microservice yaitu memperbarui status aduan menjadi diterima Pool. Potongan kode program microservice tersebut adalah seperti pada gambar 5.28 dibawah ini

```
//digunakan untuk mengambil semua aduan oleh pool
//digunakan untuk mengambil semua aduan oleh supervisor
@RequestMapping(value="rest/complaint/kokabandstatus", method=RequestMethod.POST)
public List <Complaint> findByidkokabAndStatus (@RequestBody Complaint complaint){
    String idkokab = complaint.getIdkokab();
    String status = complaint.getStatus();

    return complaintservice.findByidkokabAndStatus(idkokab, status);
}
```

Gambar 5.28. Potongan kode microservice fungsional menerima aduan

5.2.10 Implementasi Mengelompokkan Aduan

Fungsi mengelompokkan aduan dilakukan pool yaitu mengelompokkan aduan berdasarkan kategori tertentu yang telah dibuat sebelumnya. Tampilan fungsi tersebut seperti pada gambar 5.29 dibawah ini



Gambar 5.29. Tampilan mengelompokkan aduan

Untuk membuat fungsi tersebut maka dibutuhkan kode pemrograman front-end seperti pada gambar potongan kode program 5.30 dibawah ini

```
updatecategory(){
  this.service.getAllcategorybykokab(this.idkokab).subscribe(
    data => this.category = data,
    error => error
  )
}
```

Gambar 5.30. Potongan kode front-end mengelompokkan aduan

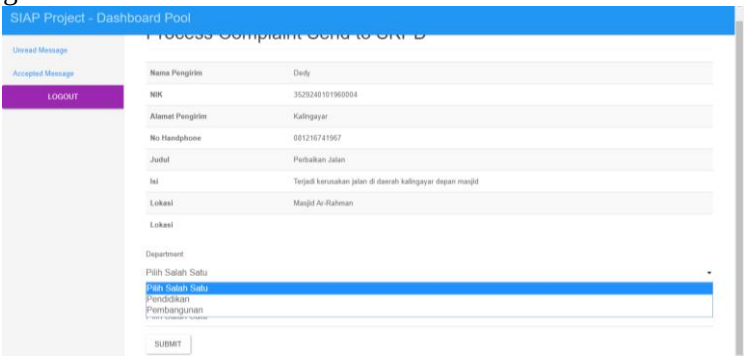
Fungsi mengelompokkan aduan membutuhkan microservice untuk memperbarui nama kategori dalam aduan. Kode program microservice tersebut adalah seperti pada gambar 5.31 dibawah ini

```
//digunakan untuk update category oleh Pool
@RequestMapping(value="rest/complaint/update/category/{id}", method=RequestMethod.PUT)
public void updateCategory (@PathVariable Long id, @RequestBody Complaint complaint){

    Complaint update = complaintservice.findByid(id);
    update.setCategory(complaint.getCategory());
    complaintservice.updateComplaint(update);

}
```

Gambar 5.31. Potongan kode microservice fungsi mengelompokkan aduan

5.2.11 Implementasi Mengirim Aduan ke SKPD

Fungsi mengirim aduan ke SKPD dilakukan oleh pool dengan memilih department yang cocok berdasarkan aduan yang diproses. Tampilan mengirim aduan ke SKPD seperti pada gambar 5.32 dibawah ini



Gambar 5.32. Tampilan mengirim aduan

Dalam membuat fungsi mengirim aduan ke SKPD maka dibutuhkan kode pemrogaman front-end seperti pada gambar potongan kode program 5.33 dibawah ini

```
onSubmit(){
  let confirmmsg;
  let r = confirm("Yakin Aduan dikirim?");
  if (r == true) {
    this.operatorservice.sentComplaint(this.idcomplaint, this.f.value).subscribe(
      ()=>{
        alert("Aduan berhasil dikirim");
        this.router.navigate(["../../"], {relativeTo: this.route});
      }
```

Gambar 5.33. Potongan kode front-end mengirim aduan ke SKPD

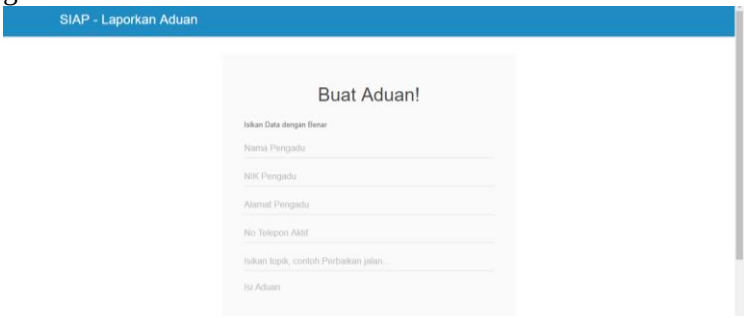
Fungsi mengirim aduan ke SKPD membutuhkan microservice untuk mengubah nama departemen dalam laporan aduan sesuai dengan nama departemen yang dipilih. Potongan kode program microservice tersebut adalah seperti pada gambar 5.34 dibawah ini

```
    //buat ngirim dari pool ke SKPD
    @RequestMapping(value="rest/complaint/update/sentskpd/{id}", method=RequestMethod.PUT)
    public void sentToSkpd (@PathVariable Long id, @RequestBody Complaint complaint){

    Complaint update = complaintservice.findByid(id);
    update.setDepartment(complaint.getDepartment());
    update.setCategory(complaint.getCategory());
    update.setStatus(complaint.getStatus());
    complaintservice.updateComplaint(update);

}
```

Gambar 5.34. Potongan kode microservice fungsi mengirim aduan ke SKPD

5.2.12 Implementasi Melaporkan Aduan

Fungsi melaporkan aduan melalui web dilakukan oleh warga dalam mengirim laporan melalui media aplikasi web SIAP. Fungsi tersebut mempunyai tampilan seperti pada gambar 5.35 dibawah ini



Gambar 5.35. Tampilan melaporkan aduan melalui web

Dalam membuat fungsi tersebut maka dibutuhkan kode program pada front-end seperti pada gambar potongan kode program 5.36 dibawah ini

```
onSubmit(){
  this.service.createcomplaint(this.f.value).subscribe(
    data =>{
        alert("Laporan telah terkirim");
    }, err => this.errorMessage="Connection failed",
    () => {
      this.router.navigate(['complaint_ui/success/']);
    }
  )
}
```

Gambar 5.36. Potongan kode front-end melaporkan aduan web

Fungsi melaporkan aduan membutuhkan microservice membuat laporan aduan baru seperti pada gambar potongan kode program 5.37 dibawah ini

```
//digunakan untuk membuat aduan baru
@RequestMapping(value="complaint/new", method=RequestMethod.POST)
public Complaint makeComplaint (@RequestBody Complaint complaint){
    return complaintservice.newComplaint(complaint);
}
```

Gambar 5.37. Potongan kode microservice fungsi melaporkan aduan

5.2.13 Implementasi Verifikasi NIK

Verifikasi yang dilakukan adalah verifikasi NIK oleh warga sebelum melaporkan aduan melalui web. Tampilan dari fungsi tersebut adalah seperti pada gambar 5.38 dibawah ini

Gambar 5.38. Tampilan verifikasi

Untuk membuat tampilan front-end maka diperlukan kode program seperti pada gambar potongan kode program 5.39 dibawah ini

```
verifyNik(nik){
  let idkokab = nik.value.toString().substring(0,4);
  this.complaintservice.checkactivecustomer(idkokab)
  .subscribe(result => {
      if (result === true) {
          this.verifiedkokab=true;
          this.successMessagekokab = "Pemerintahan telah terdaftar"
        }
      }, error => {
        this.errorMessagekokab = 'Pemerintahan belum terdaftarr';
  });
```

Gambar 5.39. Potongan kode front-end verifikasi

Verifikasi NIK membutuhkan microservice mengambil data NIK dari data kependudukan dan kemudian membandingkannya dengan masukan NIK dari warga. Potongan kode program microservice tersebut seperti pada gambar 5.40 dibawah ini

```
//digunakan untuk verifikasi nik pada saat membuat dan mengecek aduan
@RequestMapping(value="citizen/checknik", method = RequestMethod.POST)
public Boolean findCitizenBynik(@RequestBody Citizen nik) throws ServletException{


    String idnik = nik.getNik();
    Citizen citizen = citizenservice.findBynik(idnik);

    if (citizen == null){
        throw new ServletException("Customer not Registered.");
    }

    return true;
}
```

Gambar 5.40. Potongan kode microservice fungsi verifikasi NIK

5.2.14 Implementasi Menerima Tiket Aduan

Fungsi ini merupakan fungsi yang menampilkan tiket aduan setelah aduan dikirim melalui web seperti pada gambar 5.41 dibawah ini

Gambar 5.41. Tampilan menerima tiket aduan

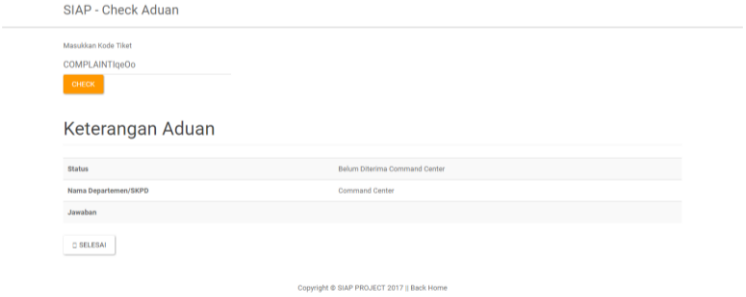
Untuk membuat fungsi tersebut maka diperlukan kode program front-end seperti pada gambar potongan kode program 5.42 dibawah ini

```
<div class="container">
<div class="col-md-6 col-md-offset-3">
  <div class="form-area">
      <p style="text-align:center"> Selamat Aduan Anda Telah terkirim </p>
      <h1 style="text-align:center"> {{secretcode}} </h1>
      <p style="text-align:center"> Silahkan catat kode diatas untuk pengecekan aduan </p>
  </div>
</div>
</div>
```

Gambar 5.42. Potongan kode front-end tiket aduan

5.2.15 Implementasi Melihat Status Aduan

Melihat status aduan dilakukan oleh warga yang telah melaporkan aduan. Fungsi ini memiliki tampilan seperti pada gambar 5.43 dibawah ini



Gambar 5.43. Tampilan melihat status aduan

Untuk membuat fungsi tersebut maka dibutuhkan kode program front-end seperti pada gambar potongan kode program 5.44 dibawah ini

```
checkTicket(ticket){
  this.service.checkusingticket(ticket.value).subscribe(
    data=>{
      this.complaintinf=data;
      switch(this.complaintinf.status) {
          case "ANSWERED":
              this.statustext="Telah Dijawab Departemen"
              break;
          case "ACCEPT":
              this.statustext="Telah Diterima Departemen"
              break;
          case "SENT":
              this.statustext="Telah Dikirim ke Departemen"
              break;
```

Gambar 5.44. Potongan kode front-end melihat status aduan

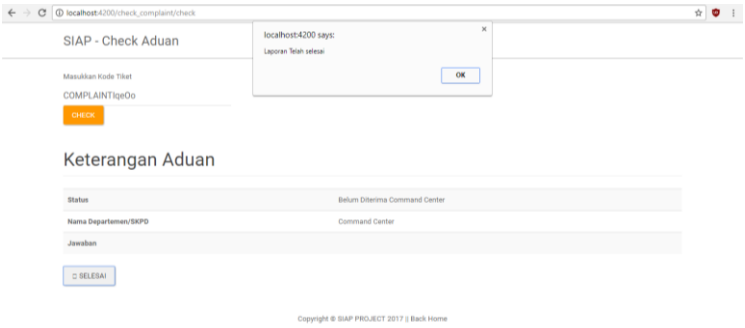
Melihat status aduan menggunakan fungsi microservice mengambil data aduan berdasarkan nomor tiket aduan. Potongan kode program microservice tersebut adalah seperti pada gambar 5.45 dibawah ini

```
 //digunakan untuk mencari aduan berdasarkan tiket kode
@RequestMapping(value="complaint/find/ticket/{ticketcode}", method=RequestMethod.GET)
public Complaint findByticketcode (@PathVariable String ticketcode){
    return complaintservice.findByticketcode(ticketcode);
}
```

Gambar 5.45. Potongan kode microservice fungsi melihat status aduan

5.2.16 Implementasi Mengubah Status Aduan

Mengubah status aduan dilakukan oleh warga karena aduan tersebut telah ditangani oleh pemerintah. Warga klik tombol selesai seperti pada gambar 5.46 dibawah ini



Gambar 5.46. Tampilan mengubah status aduan

Dalam membuat fungsi tersebut maka diperlukan kode program front-end seperti pada gambar potongan kode program 5.47 dibawah ini

```
complaintDone(ticket){
  let confirmmsg;
  let r = confirm("Laporan yang telah selesai tidak bisa diubah lagi. Yakin selesai?");
  if (r == true) {
    this.service.complaintComplee(ticket.value,"COMPLETED").subscribe(
      ()=>{
        alert("Laporan Telah selesai")
        this.router.navigate(['home'])
      }
    )
```

Gambar 5.47. Potongan kode front-end mengubah status aduan

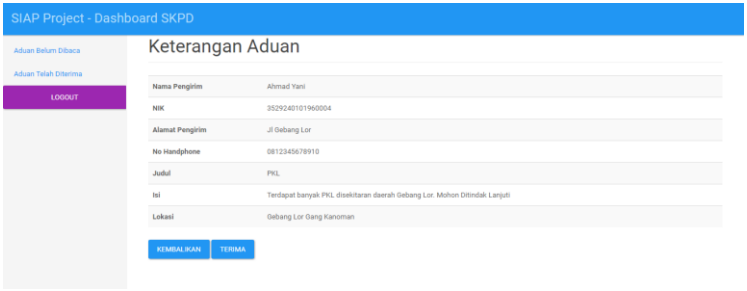
Mnengubah status aduan dibuat menggunakan microservice memperbarui status aduan. Potongan kode program microservice tersebut adalah seperti pada gambar 5.48 dibawah ini

```
//digunakan untuk mengubah status aduan menjadi selesai oleh warga
@RequestMapping (value="complaint/complete/ticket/{ticket}", method = RequestMethod.PUT)
public void complaintDone (@PathVariable String ticket, @RequestBody Complaint complaint){

    Complaint update = complaintservice.findByticketcode(ticket);
    update.setStatus(complaint.getStatus());
    complaintservice.updateComplaint(update);
}
```

Gambar 5.48. Potongan kode microservice fungsi mengubah status aduan

5.2.17 Implementasi Menerima Aduan dari Pool

Fungsi ini digunakan SKPD untuk menerima aduan yang telah diberi label kategori dan departemen sesuai dengan SKPD. Tampilan fungsi ini seperti pada gambar 5.49 dibawah ini



Gambar 5.49. Tampilan menerima aduan dari Pool

Untuk membuat fungsi tersebut maka dibutuhkan kode program front-end seperti pada gambar potongan kode program 5.50 dibawah ini

```
acceptcomplaint(){
  this.service.acceptComplaint(this.idcomplaint,"ACCEPT").subscribe(
    ()=>alert("Aduan telah diterima, cek menu Aduan diterima")
  )
  this.router.navigate(["../../"], {relativeTo: this.route});
}
```

Gambar 5.50. Potongan kode front-end menerima aduan dari pool

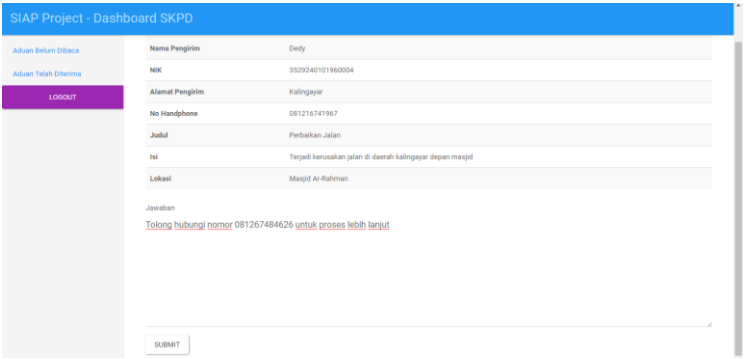
Fungsi menerima aduan dari Pool membutuhkan microservice dengan fungsi yaitu mengubah status aduan. Potongan kode program microservice tersebut adalah seperti pada gambar potongan kode program 5.51 dibawah ini

```
//digunakan untuk update status oleh Pool, SKPD, dan warga
@RequestMapping(value="complaint/update/status/{id}", method=RequestMethod.PUT)
    public void updateStatus (@PathVariable Long id, @RequestBody Complaint complaint){
    Complaint update = complaintservice.findByid(id);
    update.setStatus(complaint.getStatus());
    complaintservice.updateComplaint(update);

}
```

Gambar 5.51. Potongan kode microservice fungsi menerima aduan dari Pool

5.2.18 Implementasi Menjawab Aduan

Fungsi menjawab aduan digunakan oleh SKPD untuk menjawab aduan yang telah masuk ke dalam system. Tampilan menjawab aduan seperti pada gambar 5.52 dibawah ini



Gambar 5.52. Tampilan menjawab aduan

Fungsi tersebut dibuat dengan kode program front-end seperti pada gambar potongan kode program 5.53 dibawah ini

```
onSubmit(){
  this.operatorservice.skpdAnswerComplaint(this.idcomplaint,this.f.value).subscribe(
    ()=> {
      alert("Aduan Telah Dijawab");
      this.router.navigate(["../../"], {relativeTo: this.route});
    }
  )
}
```

Gambar 5.53. Potongan kode front-end menjawab aduan

Fungsi menjawab aduan menggunakan microservice untuk memperbarui jawaban pada data aduan. Potongan kode program microservice tersebut adalah seperti pada gambar 5.54 dibawah ini

```
    //buat menjawab aduan oleh skpd
    @RequestMapping(value="rest/complaint/update/answer/{id}", method=RequestMethod.PUT)
    public void answerComplaint (@PathVariable Long id, @RequestBody Complaint complaint){

    Complaint update = complaintservice.findByid(id);
    update.setAnswer(complaint.getAnswer());
    update.setStatus(complaint.getStatus());
    complaintservice.updateComplaint(update);

}
```

Gambar 5.54. Potongan kode microservice fungsi menjawab aduan

5.2.19 Implementasi Mengembalikan Aduan

Fungsi mengembalikan aduan digunakan oleh SKPD untuk mengembalikan aduan ke Pool karena bukan bidangnya. Tampilan mengembalikan aduan seperti pada gambar 5.55 dibawah ini

Gambar 5.55. Tampilan Mengembalikan Aduan

Untuk membuat fungsi tersebut maka dibutuhkan kode program front-end seperti pada gambar potongan kode program 5.56 dibawah ini

```
sendBack(note){
  this.service.skpdSendBackComplaint(this.idcomplaint,this.backDepartment, note, this.status).subscribe(
    ()=>alert("Aduan berhasil dikembalikan, REFRESH DATA ADUAN!")
  )
  this.router.navigate(["../../"], {relativeTo: this.route});
}
```

Gambar 5.56. Potongan kode front-end mengembalikan aduan

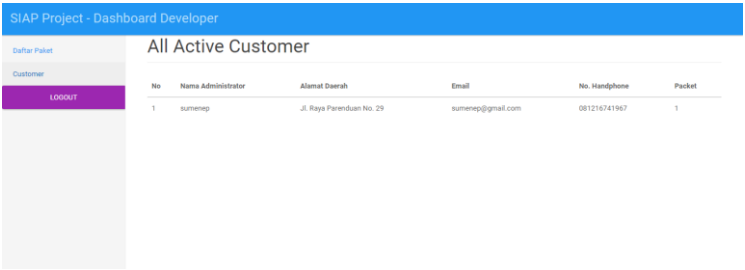
Sedangkan pada backend dibutuhkan kode program seperti pada gambar potongan kode progeam 5.57 dibawah ini

```
//buat mengirim laporan kembali dari skpd ke pool
@RequestMapping(value="rest/complaint/update/sentback/{id}", method=RequestMethod.PUT)
public void sendBackComplaint (@PathVariable Long id, @RequestBody Complaint complaint){
Complaint update = complaintservice.findByid(id);
update.setDepartment(complaint.getDepartment());
update.setNote(complaint.getNote());
update.setStatus(complaint.getStatus());
complaintservice.updateComplaint(update);
```

Gambar 5.57. Potongan kode microservice fungsional mengembalikan aduan

5.2.20 Implementasi Melihat Penyewa Layanan

Fungsi melihat penyewa layanan digunakan oleh vendor untuk melihat semua daerah yang menyewa layanan aplikasi seperti pada gambar 5.58 dibawah ini



Gambar 5.58. Tampilan Melihat Penyewa Layanan

Untuk membuat fungsi tersebut maka dibutuhkan kode program front-end seperti pada potongan kode program 5.59 dibawah ini

```
constructor(
  private service: DevService,
) { }

ngOnInit() {
  this.service.listAllCustomer().subscribe(
    data => this.allcustomer = data,
  )
}
```

Gambar 5.59. Potongan kode front-end melihat penyewa layanan

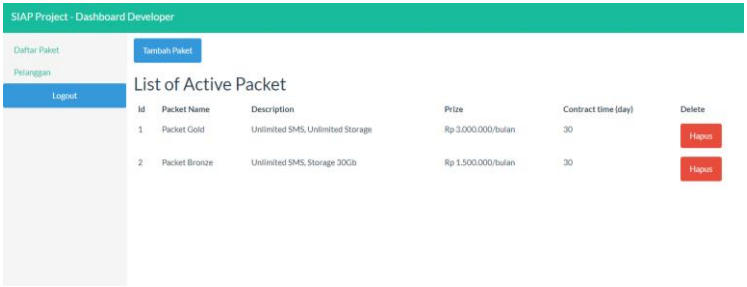
Fungsi melihat penyewa layanan menggunakan microservice mengambil semua data customer yang telah teregitrasi dalam sistem. Potongan kode microservice seperti pada gambar 5.60 dibawah ini

```
//ambil semua data customer - dev
@RequestMapping(value="/rest/user-dev/customer/all", method=RequestMethod.GET)
public List<Customer> allCustomer (){
    return customerservice.findAll() ;
}
```

Gambar 5.60. Potongan kode microservice fungsional melihat penyewa layanan

5.2.21 Implementasi Manajemen Daftar Paket

Fungsi ini dilakukan oleh pengelolah untuk membuat paket sewa, menghapus, dan melihat semua paket sewa yang ada. Tampilan fungsi ini seperti pada gambar 5.61 dibawah ini



Gambar 5.61. Tampilan Manajemen Daftar Paket

Untuk membuat fungsi tersebut dibutuhkan kode program front-end seperti pada gambar potongan kode program 5.62 dibawah ini

```
updateList(){
  this.service.listpacket().subscribe(
    data => this.listpacket = data,
    error => this.errorMessage = <any>error
  )
}
```

Gambar 5.62. Potongan kode front-end manajemen daftar paket

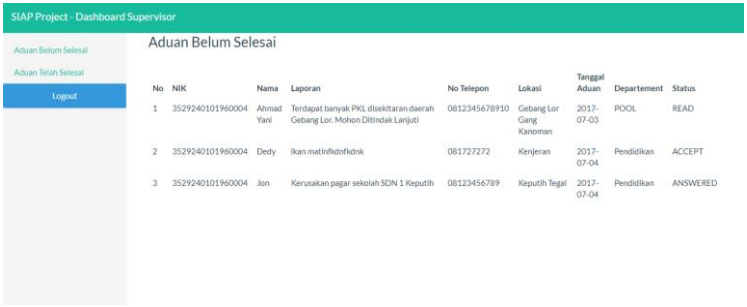
Manajemen daftar paket membutuhkan microservice dengan fungsi mengambil semua paket yang telah dibuat,membuat paket baru, dan menghapus data paket. Potongan kode program microservice seperti pada gambar 5.63 dibawah ini

```
@Autowired
private PacketService packetnameservice;

@RequestMapping(value="/rest/user-dev/packet/new", method=RequestMethod.POST)
public void addpacketname(@RequestBody Packet packetname){
        packetnameservice.addpacketname(packetname);
}

@RequestMapping(value="/rest/user-dev/packet/delete/{packetid}", method=RequestMethod.DELETE)
public void deletepacketbyid (@PathVariable Long packetid){
        packetnameservice.deletepacketbyid(packetid);
}

@RequestMapping(value="user-dev/packet/all", method=RequestMethod.GET)
public List<Packet> findAllPacketName(){
        return packetnameservice.findAllPacketName();
```

Gambar 5.63. Potongan kode microservice fungsional manajemen daftar paket

5.2.22 Implementasi Melihat Aduan belum Selesai

Melihat aduan belum selesai dilakukan oleh supervisor atau kepala pemerintah untuk mengawasi semua aduan yang masuk di command center. Tampilan dari fungsi ini seperti pada gambar 5.64 dibawah ini



Gambar 5.64. Tampilan Melihat Aduan Belum Selesai

Untuk membuat fungsi tersebut maka dibutuhkan kode program front-end seperti pada gambar potongan kode program 5.65 dibawah ini

```
ngOnInit() {
  this.service.inCompleteComplaint(this.idkokab,this.statusincomplete).subscribe(
    data=>{
      this.allcomplaint = data
    },
    err => console.log(err)
  )
}
```

Gambar 5.65. Potongan kode front-end aduan belum selesai

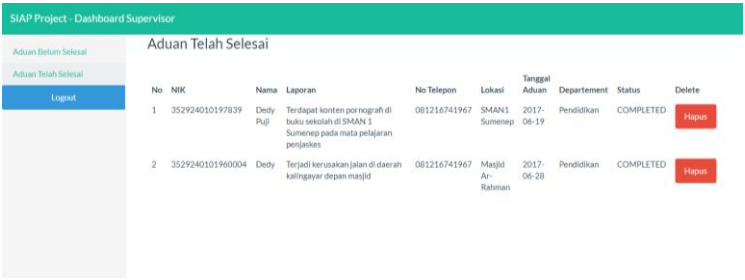
Fungsi melihat aduan belum selesai dibuat dengan microservice yang mengambil semua aduan dengan status yang belum selesai. Potongan kode program microservice tersebut adalah seperti pada gambar 5.66 dibawah ini

```
//digunakan untuk mengambil semua aduan oleh pool
//digunakan untuk mengambil semua aduan oleh supervisor
@RequestMapping(value="rest/complaint/kokabandstatus", method=RequestMethod.POST)
public List <Complaint> findByidkokabAndStatus (@RequestBody Complaint complaint){
    String idkokab = complaint.getIdkokab();
    String status = complaint.getStatus();

    return complaintservice.findByidkokabAndStatus(idkokab, status);
}
```

Gambar 5.66. Potongan kode microservice fungsional melihat aduan belum selesai

5.2.23 Implementasi Melihat Aduan Selesai

Fungsi tersebut dikerjakan oleh supervisor atau kepala pemerintahan untuk melihat aduan yang selesai. Tampilan fungsi tersebut seperti pada gambar 5.67 dibawah ini



Gambar 5.67. Tampilan tabel aduan selesai

Untuk membuat fungsi tersebut maka dibutuhkan kode program front-end seperti pada gambar potongan kode program 5.68 dibawah ini

```
ngOnInit() {
  this.service.complaintKokabStatus(this.idkokab, this.status).subscribe(
    data=>{
      this.allcomplaint = data
    },
    err => console.log(err)
  )
}
```

Gambar 5.68. Potongan Kode front-end melihat aduan selesai

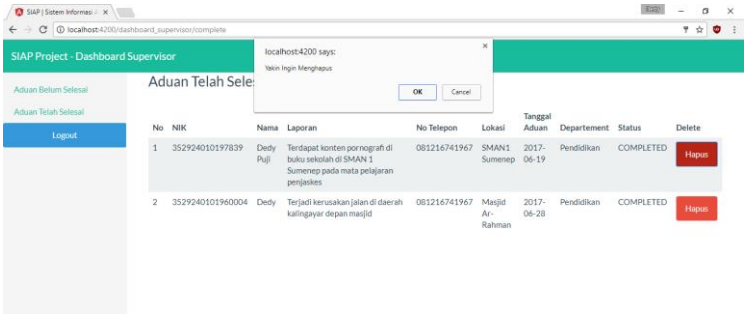
Fungsi melihat aduan selesai dibangun dengan membuat microservice yang mengambil semua aduan dengan status telah selesai. Potongan kode program microservice tersebut adalah seperti gambar 5.69 dibawah ini

```
//digunakan untuk mengambil semua aduan oleh pool
//digunakan untuk mengambil semua aduan oleh supervisor
@RequestMapping(value="rest/complaint/kokabandstatus", method=RequestMethod.POST)
public List <Complaint> findByidkokabAndStatus (@RequestBody Complaint complaint){
    String idkokab = complaint.getIdkokab();
    String status = complaint.getStatus();

    return complaintservice.findByidkokabAndStatus(idkokab, status);
}
```

Gambar 5.69. Potongan kode microservice fungsional melihat aduan selesai

5.2.24 Implementasi Menghapus Aduan

Fungsi ini dilakukan oleh supervisor untuk menghapus aduan yang telah selesai. Tampilan menghapus aduan seperti pada gambar 5.70 dibawah ini



Gambar 5.70. Tampilan Menghapus Aduan

Untuk membuat fungsi tersebut maka dibutuhkan kode program front-end seperti pada gambar potongan kode program 5.71 dibawah ini

```
deletecomplaint(idcomplaint){
  let confirmmsg;
  let r = confirm("Yakin Ingin Menghapus");
  if (r == true) {
    this.service.deletecomplaint(idcomplaint).subscribe(
      ()=>{
    alert("Aduan Telah Dihapus");
    this.ngOnInit();
    }
    )
  } else {
    alert("Aduan Gagal Dihapus");
  }
```

Gambar 5.71. Potongan kode front-end menghapus aduan

Fungsi melihat aduan dibangun dengan microservice yang berfungsi menghapus salah satu data aduan yang dipilih. Potongan kode microservice tersebut adalah seperti pada gambar 5.72 dibawah ini

```
@RequestMapping(value="complaint/delete/{id}", method=RequestMethod.DELETE)
public void deleteByid (@PathVariable Long id){
    complaintservice.deleteByid(id);
}
```

Gambar 5.72. Potongan kode microservice fungsional menghapus aduan

5.2.25 Implementasi Login

Login digunakan oleh semua aktor kecuali warga untuk masuk ke dashboard masing-masing. Berikut adalah salah satu tampilan halaman login pada gambar 5.73 dibawah ini

Gambar 5.73. Tampilan halaman login

Untuk membuat login maka diperlukan kode program front-end seperti pada gambar potongan kode program 5.74 dibawah ini

```
<form novalidate class="form-signin" [formGroup]="f" (ngSubmit)="onSubmit(f)">
  <h3 class="form-signin-heading" style="text-align:center">Login Petugas Daerah</h3>
  <input class="form-control" type="text" placeholder="Username" name="username" formControlName="username">
  <div class="alert alert-danger" *ngIf="f.get('username').hasError('required') && f.get('username').touched">
   Username is required
 </div>
  <input type="password"  class="form-control" placeholder="Password" name="password" formControlName="password">
  <div class="alert alert-danger" *ngIf="f.get('password').hasError('required') && f.get('password').touched">
   Password is required
 </div>
  <button class="btn btn-lg btn-primary btn-block" type="submit" [disabled]="f.invalid">Sign in</button>
</form>
```

Gambar 5.74. Potongan kode front-end login

Fungsi login dibangun dengan microservice dengan fungsi yang mengambil data username dan data password dari data customer atau user petugas dan kemudian mencocokkannya dengan inputan yang dikirim. Jika cocok maka akan dikirim sebuah token dengan menggunakan microservice mengirim token. Potongan kode program microservice seperti pada gambar 5.75 dibawah ini

```
//untuk login customer
@RequestMapping(value = "customer/login", method = RequestMethod.POST)
public String login(@RequestBody Customer login) throws ServletException {

    String jwtToken = "";

    if (login.getUsername()== null || login.getPassword()== null) {
        throw new ServletException("Please fill in username and password");
    }

    String username = login.getUsername();
    String password = login.getPassword();
```

Gambar 5.75. Potongan kode microservice fungsional login

5.2.26 Implementasi Logout

Fungsi logout digunakan oleh semua aktor kecuali warga untuk menghapus data login. Salah satu tampilan logout adalah seperti pada gambar 5.76 dibawah ini

Gambar 5.76. Tampilan logout

Untuk membuat fungsi logout dibutuhkan hanya kode program front-end seperti pada gambar potongan kode program 5.77 dibawah ini

```
logout(){
  localStorage.removeItem("token");
  localStorage.removeItem("currentuser");
  alert("You just logged out.");
}
```

Gambar 5.77. Potongan kode program front-end logout

5.3 Implementasi Test Case

Pengujian pada aplikasi SIAP menggunakan metode *blackbox testing* yang artinya akan menguji aplikasi berdasarkan fungsionalitasnya. Penggunaan *blackbox testing*

juga dikarenakan pengembangan aplikasi SIAP tidak menggunakan algoritma perhitungan yang rumit. *Blackbox testing* dilakukan menggunakan *testcase* sesuai dengan kebutuhan yang sudah dijabarkan. *Testing* dilakukan pada tahapan aplikasi SIAP sudah siap untuk *dideploy* ke *cloud*. Maka sebelum itu dilakukan pengujian untuk memastikan telah sesuai dengan kebutuhan fungsional. Dengan memastikan bahwa fungsional aplikasi berjalan dengan baik maka bisa dipastikan fungsional *microservice* yang ada pada tiap fungsional aplikasi juga berjalan dengan baik. Berikut adalah tabel *testcase* untuk *blackbox testing*

Tabel 5.3 Tabel Pengujian

No	Nama Use Case	Pengujian	Identifikasi pengujian
1	Login	Skenario Normal	UCT001
		Skenario Alternatif	UCT002
2	Logout	Skenario Normal	UCT003
3	Melakukan Registrasi	Skenario Normal	UCT004
		Skenario Alternatif	UCT005
4	Verifikasi	Skenario Normal	UCT006
		Skenario Alternatif	UCT007
5	Cek Status Aduan	Skenario Normal	UCT008
		Skenario Alternatif	UCT009

5.4 Implementasi Deployment Cloud

Implementasi tahap ini merupakan tahapan akhir dari aplikasi yaitu *deployment* cloud. Aplikasi SIAP yang terdiri dari *front-end* dan *back-end* akan di letakkan pada server cloud ITS yang mempunyai alamat url http://siap.its.ac.id/. Aplikasi SIAP bagian front-end di deploy dalam bentuk *production application*. Sedangkan pada bagian back-end yang berupa program java, maka harus di deploy dalam bentuk *file* jar. Setelah itu harus dilakukan beberapa konfigurasi pada server sehingga aplikasi bisa berjalan secara otomatis. Konfigurasi yang diperlukan adalah seperti melakukan instalasi web server apache, MYSQL, dan Java untuk membangun lingkungan implementasi aplikasi seperti yang dilakukan pada tahap pengembangan.

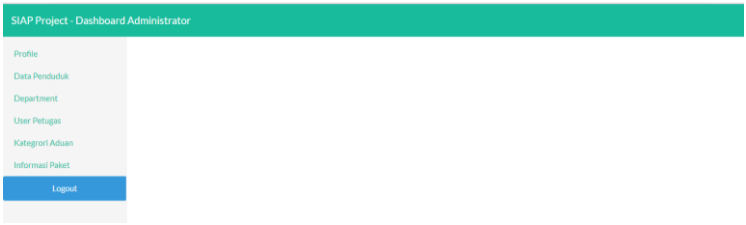
BAB 6
HASIL DAN PEMBAHASAN

Pada bagian ini akan membahas mengenai hasil dan pembahasan dari proses pengembangan aplikasi SIAP

6.1 Operasional Aplikasi SIAP

Proses operasional aplikasi SIAP sesuai dengan proses bisnis aplikasi yang telah dibahas sebelumnya bisa dimulai dengan melakuakn registrasi daerah. Untuk melakukan hal tersebut pemerintah bisa menuju halaman registrasi dari halaman awal aplikasi kemudian pada menu daftarkan daerah anda. Data yang diperlukan untuk melakukan registrasi adalah seperti nama *administrator*, alamat pemerintahan, email, nomor telepon, nama provinsi dan nama kabupaten kota, kemudian memilih paket serta *username* dan *password* yang nantinya akan digunakan untuk *login*.

Setelah melakukan registrasi, *administrator* daerah *login* dengan mengakses halaman *login* yang sesuai. Admin akan diarahkan ke halaman *dashboard* ketika *login* berhasil. Tampilan *dashboard* admin daerah seperti pada gambar 6.1 dibawah ini



Gambar 6.1. Dashboard Admin

Dalam *dashboard*, admin dapat melihat profile yaitu berupa informasi data registrasi sebelumnya. Admin dapat mengunggah data kependudukan. Data ini penting karena akan digunakan sebagai acuan untuk verifikasi NIK warga yang akan mengirim aduan agar laporan bisa diproses sesuai dengan

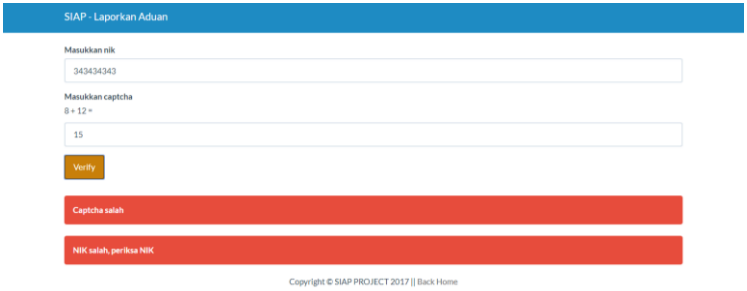
domisili daerahnya sesuai NIK KTP. Mengunggah data kependudukan yang dimaksud adalah mengunggah *file* excel yang berisi data NIK, nama, dan beberapa informasi penting lainnya. Pemerintah harus memastikan data NIK pada *file* excel yang akan diunggah tidak ada yang sama karena NIK adalah nomor unik yang berbeda tiap orang. Admin juga bisa melihat semua data penduduk sesuai dengan daerahnya dan dapat menghapus semua data tersebut.

Admin menyiapkan daftar departemen, daftar kategori aduan yang nantinya akan digunakan dalam proses pengelolaan aduan. Selain itu admin juga harus membuat data user petugas sesuai dengan departemen yang telah dibuat. *User* petugas ini nantinya yang akan mengelola aduan lebih lanjut. Menu terakhir adalah menu tagihan yang berisi informasi paket yang telah dipilih pada waktu registrasi.

Warga dapat mengirim aduan dengan memilih menu Buat Aduan yang dapat diakses pada halaman utama aplikasi SIAP seperti pada gambar 6.2 dibawah ini

Gambar 6.2. Halaman Utama Aplikasi SIAP

Warga melakukan verifikasi NIK dan menyelesaikan *capthca* yang diberikan sistem. Jika verifikasi NIK dan capthca berhasil maka akan muncul tombol lanjut yang jika diklik akan menuju menu membuat aduan. Jika verifikasi NIK dan capthca tidak berhasil maka akan muncul pesan error seperti pada gambar 6.3 dibawah ini



Gambar 6.3. Pesan Error salah NIK

Pada menu membuat aduan, warga memasukkan informasi-informasi yang dibutuhkan pada *form* buat aduan. Beberapa informasi penting tersebut seperti nama pengadu, nomor pengadu yang bisa dihubungi, alamat pengadu, NIK pengadu, topik, isi aduan, dan lokasi terjadinya aduan. Jika data yang dimasukkan telah benar, warga bisa mengirimkan aduan dengan klik tombol kirim aduan. Setelah itu warga akan menuju halaman untuk menampilkan nomor aduan yang bisa dicatat warga yang nantinya bisa digunakan untuk mengecek status aduan. Laporan aduan kemudian telah masuk dalam sistem aplikasi SIAP yang nantinya akan diproses oleh user Pool dan SKPD lebih lanjut.

Laporan aduan pertama kali diproses oleh user petugas Pool. Pool *login* pada halaman *login* yang sesuai yaitu halaman *login user* petugas daerah seperti pada gambar 6.4 dibawah ini

Gambar 6.4. Halaman Login Pool

Setelah login sukses, maka pool akan menuju dashboard pool. Dalam dashboard pool terdapat dua menu yaitu aduan masuk dan aduan telah diterima. Semua aduan yang masuk akan berada pada menu aduan masuk. Menu tersebut berisi tabel yang menampilkan informasi daftar aduan sesuai dengan daerah user petugas. Untuk melakukan proses pada aduan maka klik detail pada salah satu aduan. Maka akan muncul halaman detail yang berisi keterangan yang lebih detail dari tabel sebelumnya.

Pool melakukan seleksi pada laporan aduan. Aduan yang memang benar valid bisa diterima dan diproses lebih lanjut dengan klik tombol terima. Sedangkan aduan yang tidak valid bisa dihapus sehingga tidak bisa di proses. Halaman detail aduan pada pool seperti pada gambar 6.5 dibawah ini

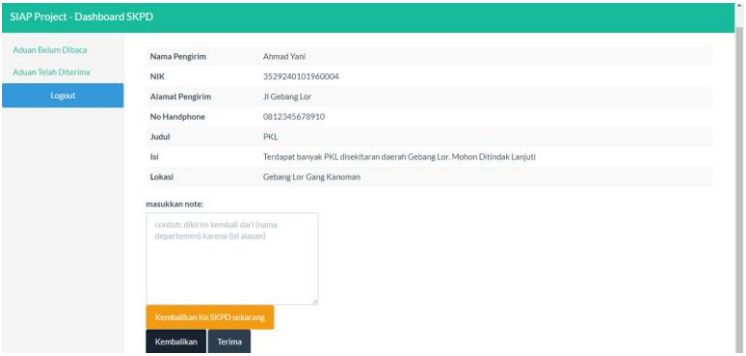
Gambar 6.5. Detail Aduan Pool

Aduan yang telah diterima akan masuk ke menu aduan telah diterima. Untuk melakukan proses lebih lanjut klik pada tombol detail. Selanjutnya akan masuk halaman detail laporan aduan yang berisi informasi lebih lengkap. Aduan kemudian diberi label kategori. Pool mengirim aduan ke SKPD dengan memilih departemen yang telah dibuat sebelumnya dan departemen yang sesuai dengan aduan pada menu detail aduan ini. Berikut adalah halaman detail proses mengirim aduan ke SKPD seperti pada gambar 6.6 dibawah ini

Gambar 6.6. Mengirim Aduan ke SKPD

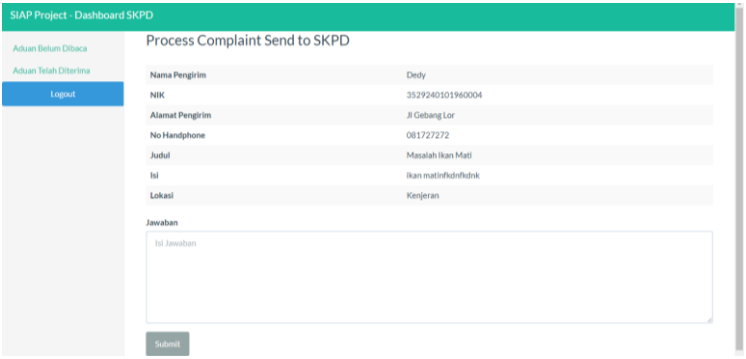
Laporan aduan telah berada di user petugas SKPD. SKPD bisa login di halaman yang sesuai untuk memproses aduan tersebut lebih lanjut. SKPD yang telah sukses login akan diarahkan ke dashboard SKPD yang memiliki dua menu yaitu Aduan belum dibaca dan aduan telah diterima. Aduan yang dikirim dari pool akan masuk pada menu aduan belum dibaca.

Proses aduan lebih lanjut oleh SKPD dengan mengklik detail pada aduan yang dipilih. SKPD akan diarahkan menuju halaman detail aduan. Untuk memproses lebih lanjut maka SKPD klik terima. Sedangkan aduan yang tidak sesuai dengan SKPD, maka SKPD bisa mengembalikannya dengan klik kembalikan. Selanjutnya akan muncul masukan untuk memasukkan pesan atau note atas dikembalikannya laporan tersebut seperti pada gambar 6.7 dibawah ini.



Gambar 6.7. Mengembalikan Aduan

Laporan aduan selanjutnya diproses pada menu Aduan telah diterima. SKPD klik pada detail salah satu laporan maka akan muncul halaman detail untuk menjawab aduan tersebut seperti pada gambar 6.8 dibawah ini



Gambar 6.8. Menjawab Aduan

Aduan telah dijawab. Warga bisa melakukan check aduan dengan menggunakan nomor tiket aduan yang telah didapat sebelumnya. Untuk mengecek aduan, warga perlu verifikasi NIK terlebih dahulu. Jika NIK benar maka akan muncul tombol lanjut untuk menuju halaman check status aduan. Warga memasukkan nomor tiket aduan lalu klik check.

Maka akan muncul status aduan, dan jawaban aduan beserta departemen yang bertanggung jawab dalam menjawab aduan tersebut.

Jika aduan memang benar-benar telah ditangani dilapangan, maka warga dapat mengklik tombol selesai yang artinya bahwa aduan telah selesai ditangani. Jika memang aduan belum selesai, warga tidak harus mengklik tombol tersebut agar laporan bisa diproses.

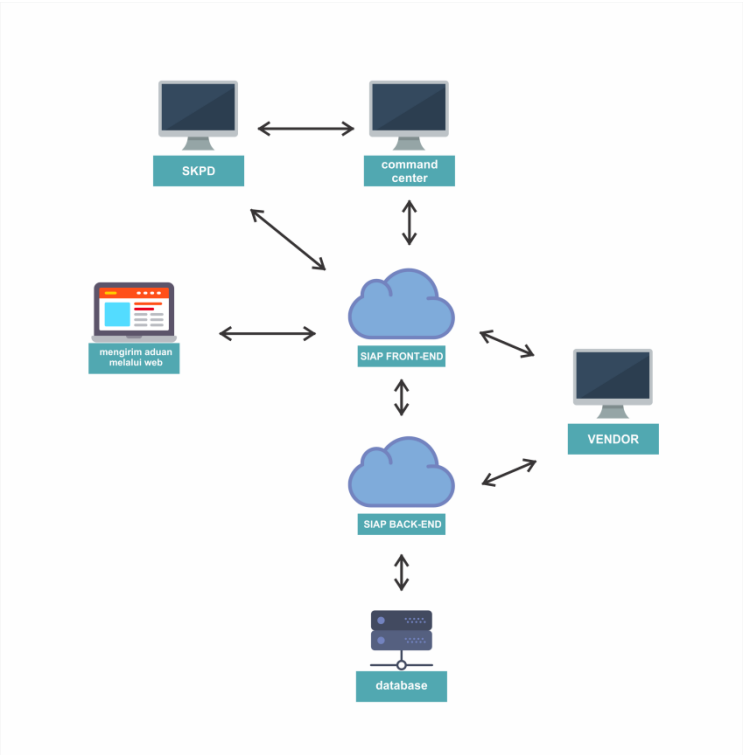
Kepala pemerintahan yang bertindak sebagai supervisor yang memantau proses pengelolaan dan penindaakan laporan bisa melakukan *login* untuk masuk ke *dashboard* supervisor. Pada *dashboard* tersebut terdapat dua menu yaitu menu aduan yang belum selesai dan menu aduan yang telah selesai.

Aduan yang belum selesai meliputi aduan yang masih belum diproses pool, aduan yang masih belum diproses SKPD, dan aduan yang telah dijawab SKPD tetapi bukan aduan yang telah selesai berdasarkan warga. Supervisor bisa memberikan peringatan kepada SKPD atau Pool yang lambat dalam penanganan laporan aduan.

Aduan yang telah selesai yaitu semua aduan yang telah diubah statusnya oleh warga akan ada pada menu ini. Supervisor bisa menghapus aduan tersebut jika memang diperlukan.

6.2 Arsitektur Aplikasi SIAP

Aplikasi SIAP yang menggunakan teknologi *cloud* mempunyai arsitektur cloud seperti gambar 6.9 dibawah ini



Gambar 6.9. Arsitektur Aplikasi SIAP

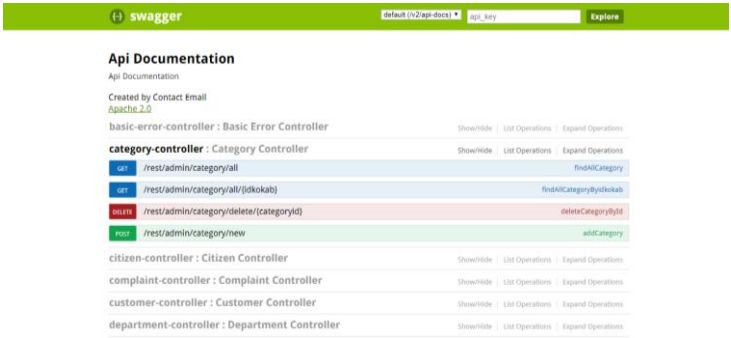
Front-end aplikasi SIAP akan di simpan di cloud begitu juga dengan *back-end* aplikasi beserta *database* aplikasi dalam layanan *cloud* yang sama untuk menghemat penggunaan *resource*.

Aplikasi SIAP diakses oleh masyarakat melalui web. Aduan yang masuk melalui web aplikasi akan dimasukkan ke dalam *database* untuk selanjutnya dikelola lebih lanjut oleh pemerintahan yang dalam hal ini adalah *command center* dan SKPD. Vendor mengakses aplikasi SIAP baik *back-end* mauoun *front-end* untuk melakukan perawatan terhadap aplikasi.

6.2 Arsitektur Microservice Aplikasi SIAP

Aplikasi SIAP sebagai aplikasi *microservice*, memecah aplikasi menjadi banyak *service-service* yang kecil yang saling mendukung satu sama lain sehingga terbentuk satu kesatuan aplikasi.

Suatu *tool* yang dapat mempermudah pengembang untuk melihat semua *microservice* yang ada pad aplikasi. *Tool* tersebut bernama swagger. Swagger adalah sebuah *framework* standart yang memungkinkan pengembang untuk menemukan dan memahami semua *service* pada aplikasi dengan mudah tanpa mengakses kode program, dokumentasi pengembangan program aplikasi, dan tanpa memerlukan inspeksi pada jaringan layanan aplikasi [**38**]. Tampilan swagger seperti pada gambar 6.10 dibawah ini



Gambar 6.10. Tampilan Swagger

Swagger memetakan microservice berdasarkan *class controller* yang telah dibuat. *Class controller* atau bisa disebut *Controller* adalah sebuah *file class* pada java yang terdapat *method-method* untuk melakukan proses input output sesuai kebutuhan bisnis yang dibuat. Pada *class controller* terdapat *method-method* yang dapat dibuatkan REST API. Swagger menampilkan semua *controller* yang telah dibuat di aplikasi. Pada setiap *controller* swagger menampilkan REST API yang bisa diakses dilengkapi dengan keterangan *method* yang

digunakan (metode REST yang digunakan biasanya adalah GET, POST, DELETE, PUT), kemudian terdapat URL REST API, dan juga nama *method* yang berada pada *class controller* tersebut. REST API inilah yang nantinya akan digunakan sebagai akses terhadap suatu *microservice* dengan metode komunikasi yang telah ditentukan sebelumnya pada *method* yang telah dibuat.

Berikut dibawah ini adalah tabel *class controller* 6.1 yang ada pada aplikasi SIAP yang kemudian akan dijabarkan *microservice* yang ada pada setiap *class controller*

Tabel 6.1. Class Controller

No	Nama Class Controller
1	Category Controller
2	Citizen Controller
3	Complaint Controller
4	Customer Controller
5	Department Controller
6	Operator Controller
7	User Dev Controller
8	Packet Controller

6.2.1 Microservice pada Category Controller

Class category controller mempunyai fungsi dalam kelola data yang berhubungan dengan kategori. Berikut daftar *microservice* yang terdapat pada *class category* sesuai pada tabel 6.2 dibawah ini

Tabel 6.2. Microservice pada Category Controller

Kebutuhan Microservice	Method REST	URL REST
Membuat data ketegori	POST	/rest/admin/category/new
Mengapus data	DELETE	/rest/admin/category/delete/

kategori		{categoryid}
Mengambil data daftar kategori	GET	/rest/admin/category/all/{id kokab}

6.2.2 Microservice pada Citizen Controller

Class citizen controller adalah sebuah *class* yang berisi *method-method* dalam kelola data yang berhubungan dengan data kependudukan. Berikut adalah daftar *microservice* pada *class citizen controller* sesuai pada tabel 6.3 dibawah ini

Tabel 6.3. Microservice pada Citizen Controller

Kebutuhan Microservice	Method REST	URL REST
Mengambil data penduduk	GET	/rest/citizen/allby/{idkokab}
Upload data penduduk	POST	/citizen/upload
Menghapus data penduduk	DELETE	/rest/citizen/delete/by/{idko kab}
Mengambil data NIK penduduk	POST	/citizen/checknik

6.2.3 Microservice pada Complaint Controller

Class complaint controller merupakan *class* yang berisi metode dalam mengelola data yang berhubungan dengan aduan. Berikut adalah daftar *microservice complaint controller* pada tabel 6.4 dibawah ini

Tabel 6.4. Microservices pada Complaint Controller

Kebutuhan Microservice	REST Method	URL REST
Memperbarui	GET	/rest/citizen/allby/{idkokab}

status aduan		
Memperbarui nama kategori data aduan	PUT	/rest/complaint/update/category/{id}
Memperbarui nama departemen data aduan	PUT	/rest/complaint/update/department/{id}
Mengirim aduan ke SKPD	PUT	/rest/complaint/update/sentskpd/{id}
Memperbarui status aduan	PUT	/complaint/update/status/{id}
Memperbarui note aduan	PUT	/rest/complaint/update/sentskpd/{id}
Menampilkan data daftar aduan status selesai	POST	rest/complaint/kokabandstatus
Mengambil data daftar aduan dengan semua status kecuali status selesai	POST	/rest/complaint/not_complete
Membuat data aduan	POST	/complaint/new
Mengambil data aduan berdasarkan nomor tiket	GET	/complaint/find/ticket/{ticket code}
Konfirmasi Aduan Selesai	PUT	/complaint/complete/ticket/{ticket}

6.2.4 Microservice pada Customer Controller

Class customer controller adalah sebuah *class* yang berisi fungsi-fungsi yang berhubungan dengan segala jenis data yang berkaitan dengan penyewa layanan, berikut adalah tabel 6.5 *microservice* yang terdapat pada *customer controller*

Tabel 6.5. Microservice pada Customer Controller

Kebutuhan Microservice	Method REST	URL REST
Membuat customer	POST	/customer/new
Login customer	POST	/customer/login
Mengambil data username customer	POST	/customer/reset/username
Memperbarui password customer	PUT	/customer/reset/password/{id}
Mengambil data nomor kota kabupaten customer	POST	/customer/kokab
Menampilkan data daftar customer	GET	/rest/user-dev/customer/all

6.2.5 Microservice pada Department Controller

Class department controller adalah sebuah *class* yang berisi fungsi-fungsi yang berhubungan dengan segala jenis data yang berkaitan dengan departemen yang telah dibuat oleh admin. Berikut tabel 6.6 yang menjelaskan *microservice* yang terdapat pada *department ccontroller*

Tabel 6.6. Microservices pada Department Controller

Kebutuhan Microservice	Method REST	URL REST

Mengambil data daftar departemen	GET	/rest/admin/department/all/{idkokab}
Membuat data departemen	POST	/rest/admin/department/delete/{departmentid}
Menghapus data departemen	POST	/rest/admin/department/new

6.2.6 Microservice pada Operator Controller

Class operator controller adalah sebuah *class* yang berisi fungsi-fungsi yang berhubungan dengan kelola segala jenis data yang berhubungan dengan data user petugas yang telah dibuat oleh admin. Berikut adalah tabel 6.7 yang menjelaskan *microservice* yang terdapat pada *operator controller*

Tabel 6.7. Microservices pada Operator Controller

Kebutuhan Microservice	Method REST	URL REST
Login User Petugas	GET	/operator/login
Membuat user petugas	POST	/rest/admin/operator/new
Mengambil data daftar user petugas	GET	/rest/admin/operator/all/{idkokab}
Menghapus user petugas	DELETE	/rest/admin/operator/delete/{id}

6.2.7 Microservice pada User Dev Controller

Class User Dev Controller adalah sebuah *class* yang berisi fungsi-fungsi yang berhubungan dalam kelola data yang berkaitan dengan vendor. Berikut adalah tabel 6.8 yang menjelaskan *microservice* yang ada pada *class* ini

Tabel 6.8. Microservice pada User Dev Controller

Kebutuhan Microservice	Method REST	URL REST
Login Vendor	POST	/user-dev/login

6.2.8 Microservice pada Packet Controller

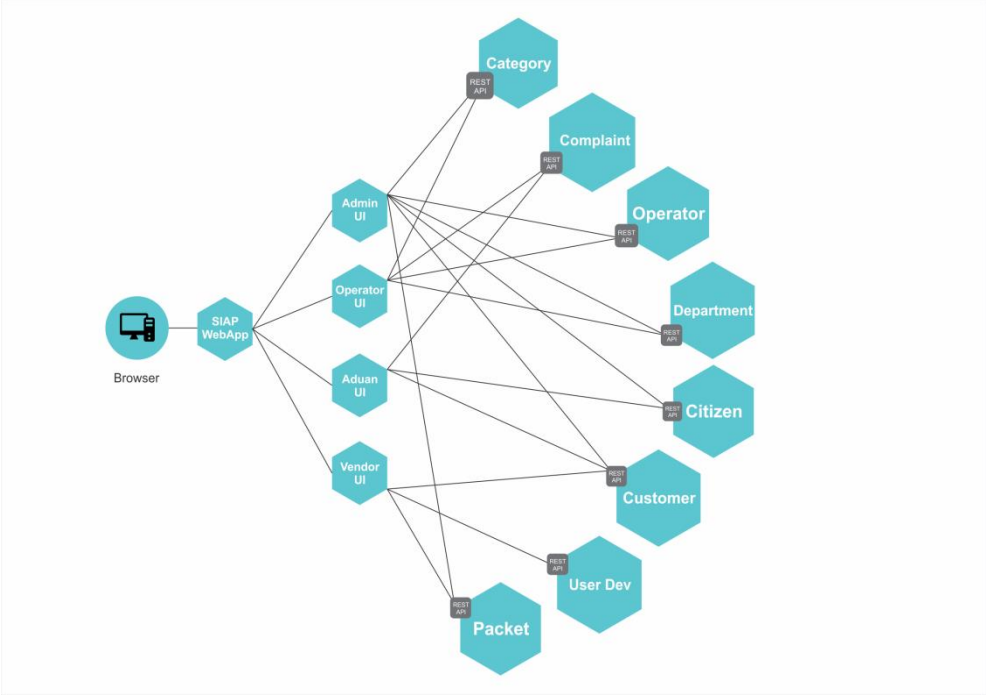
Class Packet Controller adalah sebuah *class* yang berisi fungsi-fungsi yang berhubungan dalam kelola data yang berkaitan dengan paket sewa. Berikut adalah tabel 6.9 yang menjelaskan *microservice* yang ada pada *class* ini

Tabel 6.9. Microservice pada Packet Controller

Kebutuhan Microservice	Method REST	URL REST
Membuat data paket	POST	/rest/user-dev/packet/new
Menghapus data paket	DELETE	/rest/user-dev/packet/delete/{packetid}
Mengambil data daftar paket	GET	/user-dev/packet/all

6.2.9 Diagram Microservice Aplikasi SIAP

Aplikasi SIAP pada bagian *front-end* bisa secara keseluruhan bisa dibagi menjadi empat *user interface* (UI) yaitu *dashboard* admin (Admin UI), skpd *dashboard*, pool *dashboard*, dan supervisor *dashboard* yang dikelompokkan dalam Operator UI, tampilan untuk membuat aduan (aduan UI),dan bagian *dashboard* developer (Vendor UI). Setiap UI akan berhubungan dengan beberapa *microservice* seperti pada gambar diagram microservice 6.11 dibawah ini



Gambar 6.11. Diagram Microservice Aplikasi SIAP

BAB 7
KESIMPULAN DAN SARAN

Bab ini berisikan kesimpulan dari hasil penelitian dan juga saran perbaikan untuk penelitian kedepannya

7.1 KESIMPULAN

Berdasarkan proses-proses yang telah dilakukan dalam pengerjaan tugas akhir ini maka ada beberapa kesimpulan yang dapat diambil, diantaranya sebagai berikut.

1. Aplikasi SIAP adalah jenis aplikasi *software as a service* yang dikembangkan dengan metode *microservice* Springboot. Oleh karena itu aplikasi ini di *deploy* dalam *cloud*. Dalam melakukan *deploy*, aplikasi siap di *deploy* dalam bentuk *file* jar dari aplikasi SIAP bagian *back-end.* Pada bagian *front-end* juga di *deploy* ke dalam sistem *cloud* yang sama begitu juga *database* yang digunakan aplikasi. Hal ini dikarenakan untuk menghemat *resourse*. Untuk mengakses aplikasi maka dibutuhkan *browser* yang nantinya akan melakukan komunikasi ke aplikasi yang berada di *cloud*.
2. Dalam membangun aplikasi SIAP yang berbentuk *microservice* maka kebutuhan fungsional harus dipecah menjadi beberapa *microservice* yang saling berhubungan sehingga membentuk satu kesatuan aplikasi sesuai dengan proses bisnis yang telah ditentukan.

7.2 SARAN

Adapun saran yang diberikan untuk pengembangan selanjutnya yaitu sebagai berikut

1. Aplikasi SIAP yang berupa *software as a service* yang harusnya terdapat fungsi manajemen penggunaan layanan yang baik. Tetapi pada pengembangan tahap pertama ini masih belum bisa memenuhi fungsi tersebut secara

sempurna. Perlu diadakannya pengembangan pada sistem pengelolaan penyewa pada aplikasi SIAP secara lebih baik. Pengembangan kali ini berfokus kepada pengelolaan aduan dan aspirasi saja.

2. Pengembangan pertama aplikasi SIAP masih berfokus pada pengembangan di sisi *back-end* aplikasi sehingga mengesampingkan *front-end* yang meliputi UI dan UX aplikasi. Oleh karena itu diharapkan hal tersebut lebih diperhatikan lagi pada pengembangan selanjutnya.
3. Dalam menangani laporan aplikasi SIAP belum bisa menjawab masalah mengenai warga yang berpindah domisili daerah. Oleh karena itu lagi diharapkan pada pengembangan selanjutnya hal ini bisa ditemukan solusinya.
4. Ada baiknya pada pengembangan aplikasi ini pada tahap selanjutnya dilakukan *testing* dengan menggunakan metode *white box* dan *blackbox testing* karena memungkinkan kedepannya pengembangan yang akan dilakukan akan lebih mendalam.
5. Selain itu terdapat beberapa isu seperti aduan yang bisa melibatkan dua departemen atau lebih sekaligus, dan isu-isu lain mengenai pengelolaan aduan perlu dicari solusinya dan diperbaiki pada pengembangan selanjutnya.

BAB 8
DAFTAR PUSTAKA

[1] Dwi AD Putra and Nur Aminuddin, "LANGKAH – LANGKAH TAKTIS PENGEMBANGAN E-GOVERNMENT UNTUK PEMERINTAHAN DAERAH (PEMDA) KABUPATEN PRINGSEWU," *Jurnal TAM (Technology Acceptance Model)*, p. 67, 2014.

[2] Tri Kuntoro Priyambodo and Mugenzi Thierry, "SMS and Web-Based e-Government Model Case Study: Citizens Complaints Management System at District of Gihosha – Burundi," *Indonesian Journal of Computing and Cybernetics Systems*, p. 67, 2017.

[3] Darmawan Napitupulu, "ANALISA KUALITAS LAYANAN E-GOVERNMENT DENGAN PENDEKATAN E-GOVQUAL & IPA," *Jurnal Penelitian Pos dan Informatika*, p. 153, 2016.

[4] Budi Sutedjo Dharma Oetomo, "KESIAPAN PEMERINTAH DALAM MENGEMBANGKAN SISTEM E-GOVERNMENT," *Jurnal Eksplorasi Karya Sistem Informasi dan Sains*, 2016.

[5] We Are Social Ltd. (2017, Februari) We Are Social. [Online]. http://wearesocial.com/

[6] Joe Stubbs, Walter Moreira, and Rion Dooley, "Distributed Systems of Microservices Using Docker and Serfnode," , Budapest, 2015.

[7] Sourabh Sharma, "Mastering Microservices with Java," in *Mastering Microservices with Java*. Birmingham: PACKT publishing, 2016.

[8] Rajesh RV, "Spring Microservices," in *Spring Microservices*. Birmingham: PACKT publishing, 2016.

[9] K.J. Hole, "Anti-fragile ICT Systems," *Simula SpringerBriefs on Computing 1*, 2016.

[10] Marijn Janssen and Anton Joha, "CHALLENGES FOR ADOPTING CLOUD-BASED SOFTWARE AS A SERVICE (SAAS) IN THE PUBLIC SECTOR," in *European Conference on Information Systems*, 2011.

[11] Sam Newman, "How to Model Services," in *Building Microservices: Designing Fine-Grained Systems*. United States of America: O'Reilly Media, 2014, ch. 3.

[12] Amin Jula, Elankovan Sundararajan, and Zalinda Othman, "Cloud computing service composition: A systematic literature review," *Expert Systems with Applications 41*, pp. 3809–3824, 2014.

[13] Chunlin Li, Yun Chang Liu, and Xin Yan, "Optimization-based resource allocation for software as a service application in cloud computing," *Springer Science+ Business Media New York*, 2016.

[14] Y Ishida, *Self-Repair Networks: A Mechanism Design (Vol. 101)*. Springer, 2015.

[15] Boyke Dian Triwahyudhi. (2016, April) indosystem. [Online]. https://indosystem.com/blog/microservices-konsep-dan-implementasi/

[16] Mario Villamizar et al., "Evaluating the Monolithic and the Microservice Architecture Pattern to Deploy Web Applications in the Cloud," *IEEE*, pp. 583-590, 2015.

[17] Mike Youngstrom and Phil Webb. (2012, Oktober) Spring. [Online]. https://jira.spring.io/browse/SPR-9888

[18] Mr.Anil Kumar, Dr. Arvind Kumar, and Mr. M. Iyyappana, "Applying Separation of Concern for Developing Softwares Using Aspect Programming Concepts," *International Conference on Computation*, pp. 906-9014, 2016.

[19] Ignatius Aldi Pradana. (2017, Februari) Codepolitan. [Online]. https://www.codepolitan.com/spring-boot-pengenalan-588da0c4bedd1

[20] Cecep Juliansyah Abbas and Panji Novantara, "RANCANG BANGUN SISTEM INFORMASI KECAMATAN BERBASIS E-GOVERNMENT," *JEJARING: Jurnal Teknologi dan Manajemen Informatika*, pp. 52-61, 2016.

[21] Budi Yanto, "PERANCANGAN APLIKASI ONLINE “JOGJA PEDULI” BERBASIS MOBILE UNTUK PENJARINGAN ASPIRASI PUBLIK TERHADAP INFRASTRUKTUR SARANA DAN PRASARANA JALAN DALAM PERKOTAAN DAERAH ISTIMEWA

YOGYAKARTA," *JURNAL DASI*, pp. 25-31, 2013.

[22] Anita Imawati, "PEMBANGUNAN SISTEM INFORMASI PENGADUAN PUBLIK KOTA BANDUNG," 2011.

[23] Ni Luh Yuni Lestari, Bandiyah, and Kadek Wiwin Dwi Wismayanti, "PENGELOLAAN PENGADUAN PELAYANAN PUBLIK BERBASIS E-GOVERNMENT," 2014.

[24] LAPOR. (2017) Layanan Aspirasi dan Pengaduan Online Rakyat. [Online]. https://lapor.go.id/

[25] Kediri. (2014) Sistem Layanan Aduan Pemerintah Kota Kediri. [Online]. http://surga.kedirikota.go.id/aduan/buat

[26] Hayder Saad et al., "E-LEARNING DEVELOPMENT AND ASSESSMENT REPORT: CASE STUDY I-FOLIO," vol. 85, 2016.

[27] Bukhary Ikhwan Ismail et al., "Evaluation of Docker as Edge Computing Platform," , Malaysia, 2015.

[28] Susi Susilowati, "Pengembangan Sistem Informasi Manajemen Zakat, Infaq, Shadaqoh, Waqaf dan Hibah Menggunakan Metode Waterfall," *Paradigma*, 2017.

[29] Manish Kumar, Santos Kumar Singh, and Dwivedi, "A Comparative Study of Black Box Testing and White Box Testing Techniques," *International Journal of Advance Research in Computer Science and Management Studies*,

2015.

[30] Syed Roohullah Jan, Syed Tauhid Ullah Shah, Zia Ullah Johar, Yasin Shah, and Fazlullah Khan, "An Innovative Approach to Investigate Various Software Testing Techniques and Strategies," *IJSRSET*, 2016.

[31] Alessandro Orso and Gregg Rothermel, "Software testing: a research travelogue (2000–2014)," in *Proceedings of the on Future of Software Engineering*, New York, 2014.

[32] Avitia Nurmatari. (2016, Desember) detiknews. [Online]. https://news.detik.com/berita-jawa-barat/d-3365344/aplikasi-seluruh-skpd-akan-terintegrasi-di-bandung-command-center

[33] Niagahoster. Niaga Hoster. [Online]. https://panel.niagahoster.co.id/invoice

[34] Yudho Yudhanto, Ovide Decroly Wisnu Ardhi, and Agus Purbayu, "Perancangan Dan Pembuatan Aplikasi Sistem Informasi Desa (SIMDA) Desa Ngemplak Sukoharjo," *OPEN JOURNAL SYSTEM "SEMNASTEKNOMEDIA ONLINE "*, 2017.

[35] Dedi Manholik and Agus Junaidi, "Sistem Informasi Pengaduan Masyarakat Pada Dewan Pers Indonesia," in *Konferensi Nasional Ilmu Sosial & Teknologi (KNiST)*, Jakarta, 2017, pp. 187-192.

[36] Maimunah, Hariyansyah, and Galu Jihadi, "Rancang Bangun Sistem Aplikasi Penyewaan Lapangan Futsal

Berbasis Web," *OPEN JOURNAL SYSTEM "SEMNASTEKNOMEDIA ONLINE "*, 2017.

[37] Rizki Aditya Saputra, Sulistiowati, and Julianto Lemantara, "Rancang Bangun Sistem Informasi Akademik Mahasiswa Berbasis Web Pada “Akbid Griya Husada” Surabaya," *JSIKA*, 2016.

[38] Swagger. Swagger. [Online]. https://swagger.io/getting-started/

[39] Krishna Srinivasan. (2013, Oktober) Java Beat. [Online]. http://javabeat.net/spring-boot/

[40] Chris Richardson. (2017) Microservice Architecture. [Online]. http://microservices.io/patterns/monolithic.html

[41] Heri Purnama and Indra Yatini, "APLIKASI PENGELOLAAN SKRIPSI DI STMIK AKAKOM YOGYAKARTA MENGGUNAKAN ARSITEKTUR MICROSERVICE DENGAN Node.js," in *Seminar Riset Teknologi Informasi (SRITI)*, 2016, pp. 83-88.

[42] Aditya Hadi Pratama. (2017, Januari) Techinasia. [Online]. https://id.techinasia.com/pertumbuhan-pengguna-internet-di-indonesia-tahun-2016

[43] P. China Venkanna Varma, Venkata Kalyan Chakravarthy K, V. Valli Kumari, and S. Viswanadha Raju, "Analysis of a Network IO Bottleneck in Big Data Environments Based on Docker Containers," *Big Data Research*, pp. 24-28, 2016.

[44] Adrian Mouat, *Using Docker*. United State of America: O'Reilly Media, 2015.

[45] Keyuan Jiang and Qunhao Song, "A Preliminary Investigation of Container-Based Virtualization in Information Technology Education," in *SIGITE '15 Proceedings of the 16th Annual Conference on Information Technology Education* , Chicago, 2015, pp. 149-152.

[46] Zikrul Alim and Yuni Cancer, "LAYANAN, PLATFORM AS A SERVICE (PaaS) SEBAGAI SISTEM OPERASI CLOUD COMPUTING," *Jurnal TIMES , Vol. V No 1*, pp. 32-35, 2016.

[47] Fais Risaludin Islami, Kodrat Iman Satoto, and Rinta Kridalukmana, "Pengembangan Aplikasi Manajemen Pelatihan Laboratorium Software Engineering di Fakultas Teknik Sistem Komputer," *Jurnal Teknologi dan Sistem Komputer*, 2016.

[48] Bagian Pengelola Data Elektronik Pemerintah Kota Kediri, "Dokumen Perancangan Aplikasi E-Pengaduan Suara Warga (SURGA 2.0) Kota Kediri," Kediri,.

Halaman ini sengaja dikosongkan

BIODATA PENULIS

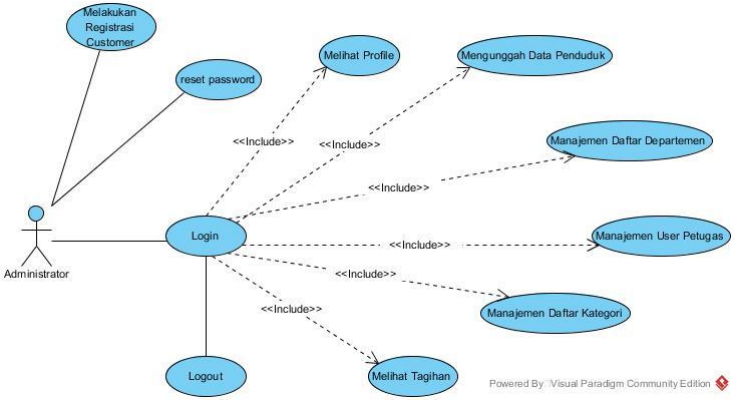
Dedy Puji Jayanto dilahirkan di kota Sumenep Madura pada tanggal 1 Januari 1996. Penulis adalah anak kedua dari tiga bersaudara yang dibesarkan di dua pulau yaitu desa Arjasa kepulauan Kangean dan di kota Sumenep kepulauan Madura. Penulis menempuh pendidikan. Penulis menempuh pendidikan dasar di SDN 1 Arjasa serta melanjutkan pendidikan di SMPN 1 Arjasa dan kemudian pindah ke Sumenep untuk melanjutkan pendidikan di SMAN 1 Sumenep dan akhirnya bisa kuliah di Departemen Sistem Informasi, Fakultas Teknologi Informasi, Institut Teknologi Sepuluh Nopember dengan NRP 5213100041. Selain kuliah, penulis juga aktif dalam beberapa kegiatan organisasi di kampus. Hal ini dibuktikan dengan menjadi staff di Kementrian Kominfo BEM ITS periode 2013/2014, staff kaderisasi di KISI ITS periode 2013/2014, manajer kreatif Badan Semi Otonom Vivat Press BEM ITS periode 2014/2015, dan wakil direktur Badan Semi Otonom Vivat Press BEM ITS periode 2015/2016. Selain itu penulis juga aktif dalam kepanitian yang diadakan di ITS dan luar ITS seperti Gerakan ITS Menulias (GIM) dan Popular Seminar Writing for Indonesia (POSEIDON ITS), LPDP EduFair, serta Gerakan Integralistik ITS (GERIGI ITS). Topik tugas akhir yang dipilih penulis merupakan topik Desain Arsitektur Software : Resilient Information Systems untuk kebencanaan dan e-government dan merupakan bidang minat dari Laboratorium Infrastruktur dan Kemanan Teknologi Informasi Departemen Sistem Informasi. Penulis dapat dihubungi melalui alamat email dedypuji@outlook.com atau nomor HP 081216741967.

Halaman ini sengaja dikosongkan

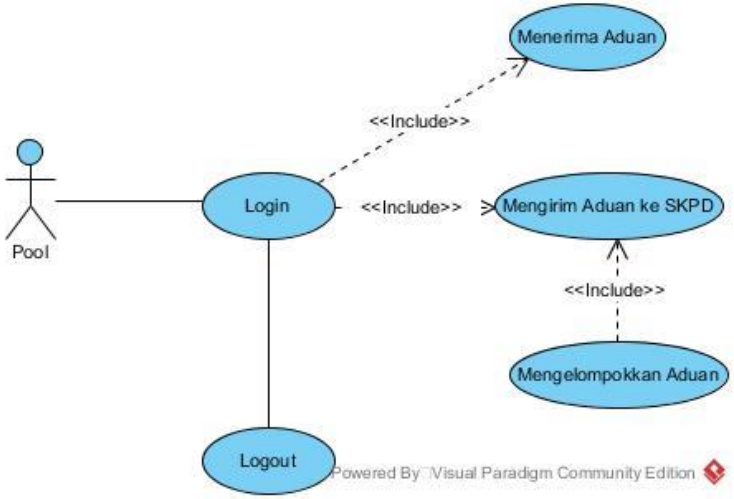
LAMPIRAN A

A.1 Desain Use case

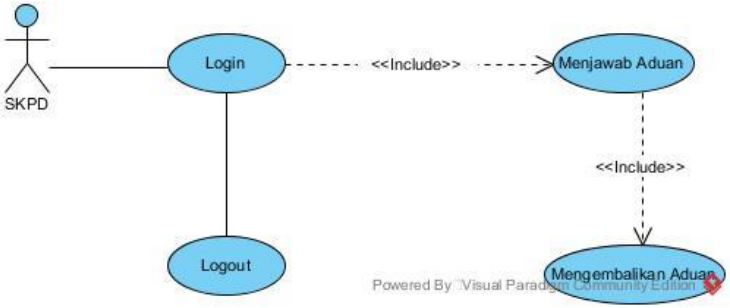
1. Use case Administrator



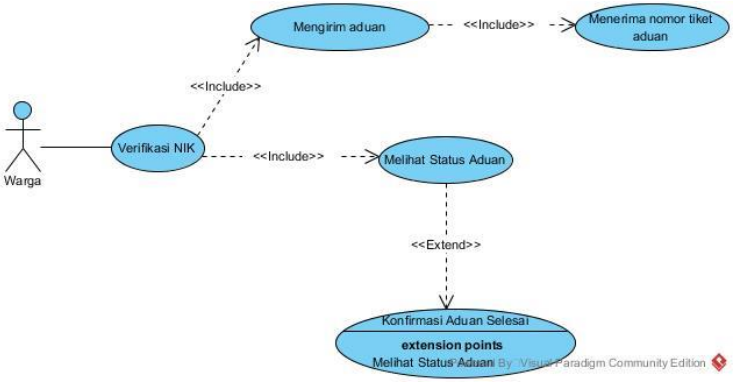
2. Use case Pool



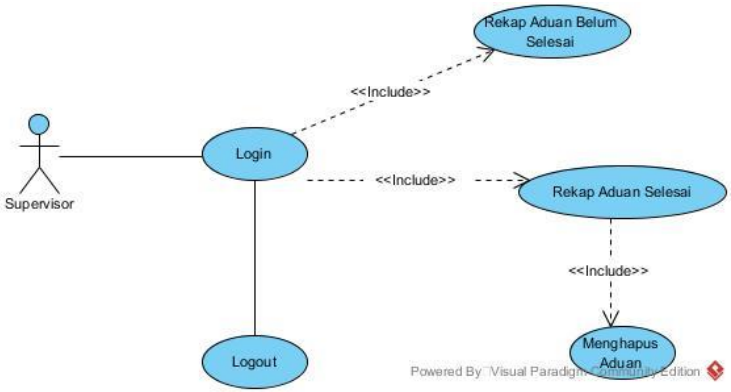
3. Use case SKPD



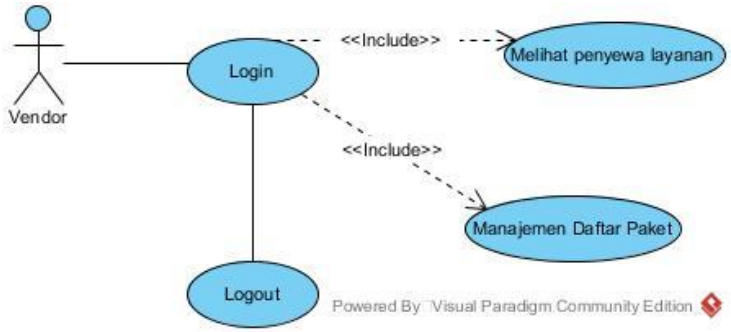
4. Use case Warga



5. Use case Supervisor



6. Use case vendor



A.2 Use case Story

Use Case name : Melakukan Registrasi Customer	UC1
Primary actor : Administrator (Admin)	
Brief Description : use case ini digunakan agar administrator sebagai wakil dari pemerintahan tertentu bisa menyewa aplikasi dengan melakukan registrasi terlebih dahulu pada website.	
Pre-condition : Untuk dapat melakukan registrasi harus mengakses alamat yang telah ditentukan	
Basic Course: 1. Setelah membuka alamat web di atas maka admin dapat memulai pendaftaran dengan klik Register 2. Admin mengisi data yang dibutuhkan pada field-field yang telah disediakan yaitu nama pemerintahan, nama lengkap, nomer telepon pemerintahan, alamat pemerintahan, email, username, password dan memlihi paket 3. Kemudian klik register	
Alternate course : 1. Jika terdapat salah satu field yang kosong pada form pendaftaran maka akan menampilkan pesan bahwa harus dilakukan pengisian dengan benar dan tidak boleh kosong.	

Use Case name : Manajemen data Kependudukan	UC2
Primary actor : Administrator (Admin)	
Brief Description : use case ini digunakan administrator untuk mengunggah data kependudukan	
Pre-condition : 1. Administrator telah melakukan login	
Basic Course: 1. Administrator memilih menu data penduduk 2. Sistem akan menampilkan halaman data penduduk 3. Untuk menambah data penduduk maka klik pada	

tombol untuk menunggah data 4. Administrator memilih file xls yang berisi data penduduk yang ingin diunggah 5. Sistem akan menampilkan table yang berisi data kependudukan yang baru 6. Administrator dapat menghapus semua data penduduk dengan menekan tombol hapus
Alternate course : -

Use Case name : Manajemen Daftar Departement	UC3
Primary actor : Administrator (Admin)	
Brief Description : use case ini digunakan administrator untuk menambah, mengedit, dan menghapus departemen yang akan terhubung dengan sistem SIAP	
Pre-condition : 1. Administrator telah melakukan login	
Basic Course: 1. Administrator memilih menu Departement 2. Sistem akan menampilkan halaman yang berupa tabel dengan isi nama department dan form departement 3. Jika admin ingin menambah departement maka masukkan nama department dalam form 4. Admin klik pada tombol simpan 5. Sistem akan memunculkan notifikasi “departemen telah ditambah” 6. Jika admin ingin menghapus daftar departemen maka klik hapus pada departemen yang ingin dihapus	
Alternate course : -	

Use Case name : Manajemen user petugas	UC4
Primary actor : Administrator (Admin)	

Brief Description : use case ini digunakan administrator untuk menambah, mengedit, dan menghapus role petugas yang akan mengoperasikan aplikasi SIAP
Pre-condition : 1. Administrator telah melakukan login
Basic Course: 1. Administrator memilih menu Operator 2. Sistem akan menampilkan halaman manajemen role petugas yang berisi tabel yang berisi data-data petugas yaitu username, password, nama, email, departemen, dan nomer telepon 3. Jika admin ingin menambah role petugas maka klik tombol tambah role 4. Sistem akan memunculkan form isian nama, nomor handphone, email, departemen, serta username dan password 5. Admin mengisi semua data yang diperlukan 6. Admin menekan tombol “submit” 7. Jika ingin menghapus role maka klik “hapus” pada salah satu data role 8. Sistem akan memunculkan pesan peringatan 9. Admin menekan tombol “delete”
Alternate course : -

Use Case name : Manajemen kategori aduan	UC5
Primary actor : Administrator (Admin)	
Brief Description : use case ini digunakan administrator untuk melihat, menambah, mengedit, dan menghapus kategori aduan	
Pre-condition : 1. Administrator telah melakukan login	
Basic Course: 1. Administrator memilih menu Kategori 2. Sistem akan menampilkan halaman kategori aduan 3. Jika ingin menambah kategori maka masukkan	

nama kategori pada masukan untuk nama kategori 4. Admin klik pada tombol save 5. Untuk melakukan hapus maka admin klik tombol hapus pada salah satu kategori
Alternate course : -

Use Case name : Melihat Tagihan UC6
Primary actor : Administrator (Admin)
Brief Description : use case ini digunakan administrator untuk melihat data tagihan sewa aplikasi
Pre-condition : 1. Administrator telah melakukan login
Basic Course: 1. Administrator memilih menu Tagihan 2. Sistem akan menampilkan halaman tagihan yang berupa tabel yang berisi data nama, nama pemerintahan, jenis paket, tanggal daftar, harga, dan hari kontrak
Alternate course : -

Use Case name : Melihat Profile UC7
Primary actor : Administrator (Admin)
Brief Description : use case ini digunakan administrator untuk melihat data informasi profile pada saat registrasi
Pre-condition : 1. Administrator telah melakukan login
Basic Course: 3. Administrator memilih menu Profile 1. Sistem akan menampilkan halaman tagihan yang berupa tabel yang berisi data nama administrator, nama pemerintahan, alamat, email, dan nomor telepon
Alternate course : -

Use Case name : Reset Password UC8
Primary actor : Administrator (Admin)
Brief Description : use case ini digunakan administrator untuk mengubah password
Pre-condition : 1. Administrator telah melakukan registrasi
Basic Course: 1. Administrator memilih menu Lupa password pada saat di halaman login 2. Sistem akan menampilkan halaman untuk verifikasi username apakah cocok dengan yang ada di database 3. Jika cocok maka akan menampilkan halaman reset password 4. Admin memasukkan password baru lalu klik reset 5. Kemudian aka nada notifikasi apakah yakin akan mengganti password 6. Jika admin memilih ok maka password berhasil diubah 7. Jika admin memilih cancel maka password tidak berubah
Alternate course : -

Use Case name: Menerima Aduan UC9
Primary actor : Pool
Brief Description : use case ini digunakan petugas Pool untuk melihat semua aduan
Pre-condition : 1. Pool telah login
Basic Course: 1. Pool memilih menu inbox 2. Pool memilih tab aduan “belum dibaca” 3. Sistem akan memunculkan halaman semua aduan yang masuk ke dalam sistem yaitu berupa tabel yang berisi nama pengadu, isi aduan, dan status 4. Untuk menerima aduan yang ingin dijawab maka

klik detail 5. Kemudian sistem akan memunculkan detail dari aduan 6. Untuk menerima aduan maka tekan tombol "diterima" 7. Sistem akan mengganti status aduan menjadi diterima system
Alternate course : -

Use Case name: Mengelompokkan Aduan	UC10
Primary actor : Pool	
Brief Description : use case ini digunakan petugas Pool untuk mengelompokkan aduan pada kategori-kategori yang sudah dibuat	
Pre-condition : 1. Pool telah login	
Basic Course: 1. Pool memilih menu inbox 2. Pool memilih tab "aduan diterima" 3. Pool memilih laporan yang akan diberikan kategori 4. Pool klik "teruskan" pada laporan 5. Sistem akan memunculkan filed untuk memberi kategori 6. Pool memilih kategori yang sesuai yang sesuai	
Alternate course : -	

Use Case name: Mengirim aduan ke SKPD	UC11
Primary actor : Pool	
Brief Description : use case ini digunakan petugas Pool untuk mengelompokkan aduan pada kategori-kategori yang sudah dibuat	
Pre-condition : 1. Pool telah login	

2. Pool telah memilih kategori aduan
Basic Course: 1. Setelah pool memilih kategori maka akan muncul pop up yang menampilkan daftar SKPD 2. Pool memilih SKPD yang sesuai 3. Sistem akan memunculkan pesan "laporan telah diteruskan" 4. Sistem akan mengganti status laporan menjadi "telah diteruskan"
Alternate course : -

Use Case name: Melaporkan aduan melalui web	UC12
Primary actor : Warga	
Brief Description : use case ini digunakan warga untuk melaporkan aduan melalui website	
Pre-condition : 1. Warga telah melakukan verifikasi	
Basic Course: 1. Sistem memunculkan form yang berisi field Nama, Nomer handphone yang bisa dihubungi, dan isi aduan 2. Warga juga bisa upload data pendukung seperti lokasi map (melalui google maps) dan gambar pendukung 3. Klik submit	
Alternate course : -	

Use Case name: Verifikasi NIK	UC14
Primary actor : Warga	
Brief Description : use case ini digunakan warga verifikasi apakah warga tersebut sesuai dengan domisili daerah tertentu menggunakan no nik dari ktp dan tanggal lahir warga untuk melakukan pelaporan	
Pre-condition :-	

Basic Course: **1.** Warga klik pada tombol Laporkan **2.** Sistem menampilkan field memasukkan nik dan tanggal lahir **3.** Warga memasukkan nik dan tanggal lahir lalu klik validate **4.** Sistem memunculkan centang hijau yang berarti telah tervaldiasi
Alternate course : 1. Warga tidak mempunyai nik yang sesuai dengan daerah tertentu sehingga sistem akan mengeluarkan notifikasi “bukan warga wilayah x”

Use Case name: Menerima tiket aduan UC15
Primary actor : Warga
Brief Description : use case ini digunakan warga sehingga mendpatkan nomer aduan
Pre-condition : 1. Warga telah mengirim aduan melalui web atau sms
Basic Course: 1. Setelah warga mengirim aduan maka sistem akan memunculkan notifikas aduan telah dikirim 2. Sistem juga akan menampilkan nomor aduan yang acak 3. Warga menyimpan nomor tiket aduan untuk mengecek status aduan 4. Jika warga melaporkan melalui sms, maka nomer aduan akan dikirim setelah sms dikirim ke nomer yang telah ditentukan dengan format laporan yang benar
Alternate course : -

Use Case name: Melihat status aduan UC16
Primary actor : Warga
Brief Description : use case ini digunakan warga untuk

melihat status aduan yang telah dikirim
Pre-condition : 1. Warga telah mengirim aduan melalui web atau sms 2. Warga telah melakukan verifikasi
Basic Course: 1. Warga memilih menu Lihat Aduan 2. Sistem memunculkan halaman verifikasi 3. Setelah sukses verifikasi, sistem akan memunculkan field isi nomer aduan 4. Warga mengisi nomer aduan 5. Lalu klik “cek” 6. Sistem akan memunculkan detail aduan yang terdiri dari status aduan dan jawaban aduan
Alternate course : -

Use Case name: Konfirmasi Aduan Selesai	UC17
Primary actor : Warga	
Brief Description : use case ini digunakan warga untuk mengubah status aduan menjadi selesai jika aduan telah ditangani oleh SKPD	
Pre-condition : 1. Warga sedang melihat status aduan	
Basic Course: 1. warga melihat detail aduan beserta status 2. Warga klik selesai. Sistem akan mengganti status laporan menjadi telah selesai	
Alternate course : -	

Use Case name: Menerima Aduan dari Pool	UC18
Primary actor : SKPD	
Brief Description : use case ini digunakan petugas SKPD untuk menerima dan melihat semua aduan	
Pre-condition :	

1. SKPD telah login
Basic Course: 1. SKPD memilih menu inbox 2. SKPD memilih tab “belum dibaca” 3. Sistem akan memunculkan halaman semua aduan yang masuk ke dalam sistem yaitu berupa tabel yang berisi nama pengadu, isi aduan, kategori dan status 4. Untuk menerima aduan yang ingin dijawab maka klik detail 5. Untuk menerima aduan maka tekan tombol “diterima” 6. Sistem akan mengganti status aduan menjadi “diterima” 7. Namun jika laporan yang dimasukkan salah SKPD maka skpd bisa klik “kembalikan” untuk mengembalikan ke pool
Alternate course : -

Use Case name: Menjawab Aduan	UC19
Primary actor : SKPD	
Brief Description : use case ini digunakan petugas SKPD untuk menjawab aduan	
Pre-condition : 1. SKPD telah login	
Basic Course: 1. SKPD memilih menu Inbox 2. SKPD memilih tab “aduan diterima” 3. Sistem akan memunculkan halaman semua aduan yang masuk ke dalam sistem yaitu berupa tabel yang berisi nama pengadu, isi aduan, kategori dan status 4. Untuk menjawab aduan maka klik detail 5. Untuk menjawab maka klik jawab 6. Sistem akan menampilkan field untuk menjawab laporan 7. SKPD menjawab laporan lalu klik “Jawab” 8. Sistem akan mengubah status laporan menjadi	

"telah dijawab"
Alternate course : -

Use Case name: Mengembalikan Aduan UC20
Primary actor : SKPD
Brief Description : use case ini digunakan petugas SKPD untuk mengembalikan aduan yang salah SKPD
Pre-condition : 1. SKPD telah login 2. SKPD telah masuk pada menu inbox pada menu "belum dibaca"
Basic Course: 1. Sistem akan memunculkan halaman semua aduan yang masuk ke dalam sistem yaitu berupa tabel yang berisi nama pengadu, isi aduan, kategori dan status 2. Untuk mengembalikan aduan maka klik detail pada aduan 3. Kemudian klik kembalikan
Alternate course : -

Use Case name: Melihat Penyewa Layanan UC21
Primary actor : Vendor
Brief Description : use case ini digunakan vendor untuk melihat siapa saja yang menyewa layanan aplikasi SIAP
Pre-condition : 1. Vendor telah login
Basic Course: 1. Vendor memilih menu Lihat Penyewa 2. Sistem akan memunculkan halaman semua penyewa dalam bentuk tabel yang berisi informasi penyewa layanan
Alternate course : -

Use Case name: Manajemen Daftar Paket UC22
Primary actor : Vendor
Brief Description : use case ini digunakan vendor untuk melihat semua paket dan membuat paket
Pre-condition : 1. Vendor telah login
Basic Course: 1. Vendor memilih menu Daftar Paket 2. Sistem akan memunculkan halaman yang berisi tabel dengan informasi semua paket yang telah dibuat 3. Jika ingin menambah paket maka klik Tambah Paket 4. Maka akan muncul form untuk menambah paket 5. Jika ingin mengapus paket maka klik Hapus
Alternate course : -

Use Case name: Rekap Aduan Belum Selesai UC23
Primary actor : Kepala Pemerintahan
Brief Description : use case ini digunakan Kepala Pemerintahan untuk melihat semua aduan
Pre-condition : 1. Kepala Pemerintahan telah login
Basic Course: 1. Kepala Pemerintahan memilih menu Aduan Belum Selesai 2. Sistem akan menampilkan field untuk melihat semua aduan dengan semua status kecuali status selesai 3. Jika ingin melihat Semua aduan yang telah selesai maka pilih menu Aduan Telah selesai
Alternate course : -

Use Case name: Rekap Aduan Selesai UC24
Primary actor : Kepala Pemerintahan
Brief Description : use case ini digunakan Kepala Pemerintahan untuk melihat semua aduan
Pre-condition : 1. Kepala Pemerintahan telah login
Basic Course: 1. Kepala Pemerintahan memilih menu Aduan Telah Selesai 2. Sistem akan menampilkan field untuk melihat semua aduan dengan semua status selesai
Alternate course : -

Use Case name: Menghapus Aduan Selesai UC25
Primary actor : Kepala Pemerintahan
Brief Description : use case ini digunakan Kepala Pemerintahan untuk menghapus aduan yang telah selesai
Pre-condition : 1. Kepala Pemerintahan telah login
Basic Course: 1. Kepala Pemerintahan memilih menu Aduan Telag Selesai 2. Sistem akan menampilkan field untuk melihat semua aduan yang telah selesai 3. Kepala pemerintahan menghapus dengan klik hapus
Alternate course : -

Use Case name : Login UC26
Primary actor : Administrator, Pool, SKPD, Kepala Pemerintahan, Vendor
Brief Description : use case ini digunakan Admin untuk masuk (login) dalam sistem
Pre-condition : 1. Aktor telah teregistrasi

2. Aktor mengakses alamat url untuk login
Basic Course: 1. Sistem akan memunculkan halaman untuk login 2. Aktor menginputkan username dan password yang telah teregistrasi pada field yang disediakan 3. Sistem melakukan validasi atas data username dan password yang telah diinputkan. 4. Sistem melakukan proses verifikasi username dan password yang sudah terdaftar dalam database 5. Sistem menampilkan halaman sesi sesuai pengguna.
Alternate course : 1. Jika terdapat salah satu field yang kosong pada form login maka akan menampilkan pesan bahwa harus dilakukan pengisian dengan benar dan tidak boleh kosong. 2. Jika username dan password yang diinputkan tidak sesuai dengan database maka akan menampilkan pesan "username/password Anda tidak sesuai".

Use Case name : Logout UC27
Primary actor : Administrator, Pool, SKPD, Kepala Pemerintahan, Vendor
Brief Description : use case ini digunakan Admin untuk keluar (logout) dalam sistem
Pre-condition : 1. Aktor telah login
Basic Course: 1. Sistem memunculkan tombol untuk logout **2.** Aktor klik pada tombol logout
Alternate course : -

Halaman ini sengaja dikosongkan

LAMPIRAN B

B.1 Perancangan Test Case

Pengujian yang digunakan pada penilitian ini adalah pengujian dengan menggunakan metode blackbox. Berikut adalah beberapa prosedur pengujian yang akan digunakan

Identifikasi pengujian	Skenario	Prosedur Pengujian	Masukan	Output Espektasi
UCT001	Skenario Normal	1. Membuka halaman login aplikasi 2. Kemudian memasukan username dan password yang sudah terdaftar dalam database untuk login.	Input karakter password dan username	Berhasil ke halaman dashboard sesuai role

UCT002	Skenario Alternatif	1. Membuka halaman login aplikasi 2. Memasukan username dan password yang tidak ada dalam database.	Input karakter username dan password	Keluar pesan " username atau password salah "
UCT003	Skenario Normal	Melakukan klik tombol logout	-	Masuk ke halaman home
UCT004	Skenario Normal	1. Memamasukan data yang diperlukan untuk melakukan registrasi 2. Menekan tombol simpan	Input karakter	Data berhasil disimpan dalam sistem dan langsung menuju halaman login
UCT005	Skenario Alternatif	1. Memasukkan data yang diperlukan 2. Terjadi kesamaan nama daerah dan username	Input Karakter	Menampilkan pesan daerah telah teregister dan username telah ada

UCT006	Skenario Normal	1. Memasukkan nomor Nik 2. Memasukkan captcha 3. Klik verifikasi	Input Karakter	Menuju halaman selanjutnya yaitu halaman membuat aduan atau halaman untuk cek status aduan
UCT007	Skenario Alternatif	1. Memasukkan nomor nik 2. Memasukkan captcha 3. Klik verifikasi	Input Karakter	1. Nik salah maka akan muncul pesan nik salah 2. Jika pemerintahan berdasarkan kode kabupaten kota dari input nik tidak terdaftar maka muncul pesan pemerintahan tidak terdaftar 3. Jika captcha salah maka akan muncul pesan captcha salah
UCT008	Skenario Normal	1. Warga memasukkan nomor tiket aduan 2. Klik check	Input Karakter	Sistem akan memunculkan keterangan mengenai

				status aduan seperti status, nama departemen dan juga jawaban
UCT009	Skenario Alternatif	1. Warga memasukkan nomor tiket aduan 2. Klik check	Input Karakter	Jika tidak ditemukan aduan dengan nomor tiket aduan yang telah dimasukkan maka sistem akan memunculkan pesan Tidak ada Complaint dengan No Kode. Hubungi service