Rancang Bangun Layanan Web (Web Service) Untuk Aplikasi Rekam Medis Praktik Pribadi Dokter

Article · January 2017

CITATION
1

READS
2

3 authors, including:

Roslidar Roslidar
Syiah Kuala University
9 PUBLICATIONS **5** CITATIONS
SEE PROFILE

Rahmad Dawood
Syiah Kuala University
32 PUBLICATIONS **11** CITATIONS
SEE PROFILE

e-ISSN: 2252-7036
Vol.2 No.1 2017: 1-8

Rancang Bangun Layanan Web (Web Service) Untuk Aplikasi Rekam Medis Praktik Pribadi Dokter

Yusrizal [#1], Rahmad Dawood[#2] dan Roslidar [#3]

Jurusan Teknik Elektro, Fakultas Teknik, Universitas Syiah Kuala

Jl. Tgk. Syech Abdurrauf No. 7, Darussalam, Banda Aceh,23111,Aceh, Indonesia

[1]yusrizal2793@gmail.com
[2]rahmad.dawood@unsyiah.ac.id
[3]roslidar@unsyiah.ac.id

***Abstrak*—Rekam medis merupakan hal yang penting di setiap rumah sakit, puskesmas, dan klinik-klinik kesehatan. Rekam medis biasanya berisi tentang data penting pasien, serta seluruh riwayat penyakit yang dialami oleh pasien. Rekam medis bagi seorang dokter merupaka sebuah hal yang sangat berguna, karena pada rekam medis ini dokter dapat melihat semua tindakan pengobatan/terapi yang pernah diberikan terhadap pasien. Tempat praktik pribadi dokter yang ada pada saat ini rekam medisnya masih dikelola secara konvensional, yaitu rekam medis yang masih ditulis diatas kertas dan disimpan pada sebuah lemari penyimpanan rekam medis. Hal tersebut tentunya akan membuat pengelolaan dan pengaksesan rekam medis menjadi sulit. Untuk mengatasi hal tersebut peneliti mencoba membuat layanan web yang bersifat RESTful bagi rekam medis supaya dapat dipakai oleh berbagai unit layanan kesehatan. Hal ini diharapkan dapat membuat pengelolaan dan pengaksesan rekam medis yang lebih mudah.**

Kata Kunci*—Electronic Health Record, Web Service, google cloud endpoint, JSON, RESTful.

I. PENDAHULUAN

Rekam medis merupakan keterangan tertulis maupun terekam tentang identitas, anamnesa, penentuan fisik, laboratorium, diagnosa segala pelayanan dan tindakan medis yang diberikan kepada pasien dan pengobatan baik yang dirawat inap, rawat jalan maupun yang mendapatkan pelayanan gawat darurat[1]. Rekam medis yang ada pada tempat praktik pribadi dokter, masih banyak yang menggunakan sistem konvensional, dimana rekam medis masih ditulis pada sebuah kartu dan disimpan pada lemari penyimpanan. Pada saat peneliti melakukan wawancara dengan pihak praktik pribadi dokter yang ada di Kota Banda Aceh, peneliti melihat ketika pasien yang terdaftar pada sebuah tempat praktik pribadi dokter semakin banyak, maka ruang penyimpanan untuk semua rekam medis pasien juga akan semakin luas.

Pihak praktik pribadi dokter membuat sebuah lemari khusus untuk menyimpan rekam medis semua pasien yang ada. Ketika lemari penyimpanannya penuh, pihak praktik pribadi dokter akan meletakkan beberapa rekam medis pasien di atas lemari. Sistem penyimpanan konvensional ini juga memiliki kekurangan yaitu resiko kerusakan fisik. Kekurangan lainnya yang peneliti lihat adalah ketika pihak praktik pribadi dokter ingin mencari rekam medis pasien, maka pihak praktik pribadi dokter harus mencari rekam medis pasien pada lemari penyimpanan dan membutuhkan waktu yang agak lama.

Adapun salah satu solusi untuk mengatasi kesemrawutan pengelolaan rekam medis di atas adalah dengan memanfaatkan teknologi yang ada sekarang ini. Salah satu teknologi yang dapat di terapkan adalah penggunaan berupa teknologi penyimpanan data berbasis digital. Data yang disimpan secara digital akan mudah dikelola karena penyimpanan data ini tidak memerlukan ruang yang besar dan dapat mengindari kerusakan data-data penting. Salah satu teknologi yang dapat digunakan adalah penyimpanan data di awan. Untuk mengakses data yang tersimpan di awan tersebut digunakan layanan web berbasis REST, atau lebih dikenal dengan sebuatan RESTful. RESTful digunakan karena lebih mudah dalam pengelolaannya, karena RESTful merupakan web service berorientasi pada *resource*. Maksud berorientasi pada *resource* adalah orientasi yang menyediakan resource-*resource* sebagai layanan dan bukan kumpulan-kumpulan dari aktifitas yang mengelola *resource* tersebut. Melalui aplikasi yang diakses menggunakan RESTful ini, semua data pengguna dari setiap kelompok pengguna akan disimpan di awan, diharapkan dengan penyimpanan seperti ini dapat mempermudah pengelolaan rekam medis yang lebih baik dari yang sebelumnya.

II. DASAR TEORI

A. Rekam Medis

Rekam medis adalah berkas yang berisikan catatan dan dokumen tentang identitas pasien, pemeriksaan, pengobatan, tindakan dan pelayanan lain yang telah diberikan kepada pasien [2]. Rekam medis harus disimpan dan dirawat dengan baik karena rekam medis merupakan harta benda rumah sakit yang sangat berharga. Penyimpanan rekam medis merupakan sebuah pencegahan untuk menghindari kerusakan pada rekam medis. Ada dua

e-ISSN: 2252-7036
Vol.2 No.1 2017: 1-8

cara pengurusan penyimpanan dalam pengelolaan rekam medis, yaitu Sentralisasi dan Desentralisasi. Sentralisasi adalah penyimpanan rekam medis pasien yang disatukan baik itu catatan kunjungan poliklinik atau catatan selama seorang pasien dirawat, yang kemudian disimpan pada satu tempat. Desentralisasi adalah penyimpanan rekam medis pada masing-masing unit pelayanan. Rekam medis antara pasien poliklinik dengan pasien yang dirawat akan disimpan di tempat yang berbeda. Rekam medis poliklinik akan disimpan pada poliklinik yang besangkutan, sedangkan rekam medis pasien dirawat disimpan ditempatnya sendiri[3].

B. Web Service (Layanan Web)

Web services merupakan sebuah sistem terdistribusi memiliki komponen yang dapat di-deploy dan diakses menggunakan protokol HTTP (Hyper Text Transport Protocol) maupun HTTPS (HTTP Secure). Layanan web dapat di program dalam berbagai bahasa pemograman yang ada. Pada web services sekurang-kurangnya terdapat sebuah web server (jaringan penyedia layanan) dan sebuah klien. Klien meminta layanan yang ditawarkan oleh web server bisa melaui desktop/PC maupun mobile.

Layanan web memiliki dua teknik popular, yaitu SOAP (Simple Object Access Protocol) dan RESTfull. Dalam layanan berbasis SOAP, klien mengirimkan pesan SOAP kelayanan dan layanan merespon dalam jenis pesan SOAP juga. RESTfull adalah salah satu yang menggunakan HTTP tidak hanya sebagai infrastruktur pertukaran informasi, tetapi juga sebagai sebuah pedoman untuk merancang permintaan layanan dan tanggapan dari layanan. Dalam layanan RESTfull, HTTP itu sendiri dapat diketahui sebagai suatu API. SOAP memiliki standar, toolkit, dan perpustakaan *software* yang melimpah. REST tidak memiliki standard resmi, relatif sedikit toolkit, dan pustaka *software* yang tidak merata antara bahasa pemrograman, namun ada dukungan yang terus ditingkatkan untuk layanan REST di seluruh bahasa pemograman[4].

C. Javascript Object Notation (Json)

Javascript Object Notation (JSON) adalah format pertukaran data yang ringan, mudah dibaca dan ditulis, serta mudah diterjemahkan dan dibuat (*generate*). Format ini dibuat berdasarkan bagian dari Bahasa Pemrograman JavaScript. JSON merupakan format teks yang tidak tergantung pada bahasa pemrograman apapun karena menggunakan bahasa yang umum digunakan oleh programmer keluarga C antara lain C, C++, C#, Java, JavaScript, Perl, Python, dll. Oleh karena itu sifat-sifat tersebut, JSON ideal sebagai bahasa pertukaran data. JSON memilik beberapa kelebihan dibandingkan dengan XML, kelebihan-kelebihan antara lain:

1) Format Penulisan: Untuk mereprensentasikan sebuah struktur data yang rumit dan berbentuk hirarkis penulisan, JSON relatif lebih terstruktur dan mudah.

2) Ukuran: Ukuran karakter yang dibutuhkan JSON lebih kecil dibandingkan XML untuk data yang sama. Hal ini tentu berpengaruh pula pada kecepatan pertukaran data walaupun tidak signifikan untuk data yang kecil, namun cukup berarti jika koneksi yang digunakan relatif lambat untuk mengakses aplikasi web kaya fitur yang memanfaatkan pertukan data. Di sini JSON lebih unggul dibandingkan dengan XML, kecuali jika data dikompresi terlebih dahulu sebelum dikirimkan, perbedaan JSON dan XML yang telah dikompresi tidaklah signifikan.

3) Browser Parsing: Proses parsing merupakan proses pengenalan token atau bagian-bagian kecil dalam rangkaian dokumen XML/JSON. Contohnya, terdapat data text dalam format JSON. Data tersebut harus di parsing terlebih dahulu sebelum dapat diakses dan dimanipulasi. Browser parsing berarti proses yang terjadi pada sisi client/browser. Melakukan browser parsing pada JSON lebih sederhana dibandingkan pada XML, JSON menggunakan function JavaScript eval() untuk melakukan parsing. Sementara dokumen XML diparsing oleh XML HttpRequest [5]

D. Google App Engine

Google App Engine adalah salah satu PaaS (*Platform as a Service*) yang disediakaan oleh Google bagi pengembang aplikasi. Google App Engine merupakan platform pengembangan dan hosting aplikasi milik Google. Google App Engine memungkinkan Anda membuat aplikasi web dengan lalu lintas tinggi tanpa harus mengelola infrastruktur dengan lalu lintas tinggi. Aplikasi yang dibuat di Google App Engine menggunakan teknologi yang sama dengan teknologi yang memberdayakan situs web Google dengan kecepatan dan keandalannya. Pemilik/ pengembang aplikasi cukup menangani kode sumber program.

Google app engine dirancang untuk *host* sebuah aplikasi dengan banyak pengguna. Sebuah aplikasi yang dapat melayani banyak pengguna disebut sebagai skala. Karena orang lebih banyak menggunakan *app engine*, maka *app engine* mengalokasikan banyak sumber daya untuk aplikasi serta pengelolaan penggunaan sumber daya tersebut. *Google app engine* menyediakan tiga buah runtime envirotment untuk aplikasi, yaitu *java envirotment, python exnvirotment,* dan *envirotment* yang berdasarkan bahasa sistem baru yang dikembangkan oleh google[9].

E. Google Endpoint

Salah satu layanan yang tersedia pada Platform as a Service (PaaS) Google App Engine adalah Google Endpoint. Google Endpoint memungkinkan pengguna untuk membuat RESTFul web services, yang selanjutnya dapat digunakan oleh aplikasi klien, baik itu iOS, Android, dan JavaScript. Google Endpoint secara otomatis juga dapat membuat pustaka bagi klien. Beberapa fitur built-in yang dimiliki Google Endpoint adalah proteksi terhadap denial-of-service (DOS), dukungan otentikasi Oauth 2.0, dan manajemen kunci klien [6].

F. RESTFUL

Salah satu kriteria desain web services yang paling sering digunakan adalah restful web services, restful sendiri bekerja denga cara resource-oriented. Pada restful

e-ISSN: 2252-7036
Vol.2 No.1 2017: 1-8

web services *client* (requester) mengakses services yang ditawarkan oleh web server, yaitu dengan cara mengakses url dari resource menggunakan method pada http.
Dalam dunia web API, protokolnya adalah http. API *client* dapat berinteraksi dengan API dengan mengirimkan berbagai jenis pesan http. Standar http mendefinisikan delapan jenis pesan, yaitu:

- GET
  Method Get mengambil data dari web server dengan menentukan parameter di bagian url dari permintaan.
- DELETE
  Method Delete menghapus sumber daya
- POST
  Method Post memanfaatkan badan pesan untuk mengirim data ke server web.
- PUT
  Method Put mirip dengan post memanfaatkan badan pesan untuk mentransfer data.
- HEAD
  Method Head digunakan untuk mengambil informasi tentang url dari web server.
- OPTION
  Method Option berguna untuk mencari tahu mana metode http dapat diakses oleh klien.
- LINK
  Method Link dapat digunakan untuk membuat sambungan jaringan ke server web melalui http.
- UNLINK
  Method Unlink dapat digunakan untuk memutus sambungan jaringan ke server web [7].

G. *Unified Modeling Language (UML)*

Unified Modeling Language(UML) didefinisikan sebagai tujuan umum bahasa pemodelan standar di bidang rekayasa perangkat lunak berorientasi objek. Standar ini dikelola dan telah dibuat oleh Object Management Group (OMG). Ini pertama kali ditambahkan ke daftar OMG pada tahun 1997, dan sejak itu menjadi standar industri untuk software-intensif pemodelan sistem. Ini mencakup seperangkat teknik notasi grafis untuk menciptakan model visual sistem perangkat lunak berorientasi objek [8].

1) Deployment Diagram: Deployment diagram menunjukkan bagaimana sebuah sistem/aplikasi berjalan dalam perangkat komputasi deployment. Tujuan dari *deployment diagram* adalah untuk menunjukkan dimana semua komponen akan berjalan, dan bagaimana komponen-komponen tersebut saling berkomunikasi satu sama lain. Notasi dalam deployment diagram mencakup unsur-unsur notasi yang digunakan dalam komponen diagram, dengan beberapa penambahan, termasuk konsep simbol. Sebuah simbol mewakili simbol yang ada pada sebuah simbol yang ada pada vitual mesin.

2) Class Diagram: Struktur statis sebuah kumpulan *database* aplikasi dapat ditampilkan pada class diagram. Hal ini menunjukkan bagaimana entitas (orang, benda, dan data) yang berbeda dapat saling berhubungan antara satu dengan yang lainnya. *Calss diagram* adalah struktur statis UML yang menggambarkan suatu sistem dengan menunjukkan kelas, atribut, operasi atau method, serta hubungan antara kelas-kelas yang ada pada suatu sistem.

3) Component Diagram: Komponen diagram memberikan penglihatan fisik dari system, ini menggambarkan bagaimana komponen dihubungkan bersama untuk membentuk komponen yang lebih besar atau sistem dari sebuah perangkat lunak. Komponen diagram diguakan untuk menggambarkan struktur sebuah sistem yang lebih komplek. Tujuannya adalah untuk menunjukkan bahwa perangkat lunak komponen dari perangkat lunak lainnya (seperti, perpustakaan perangkat lunak) pada sebuah system[8].

*4) Artifact Model:*Artifact merupakan informasi fisik yang berkaitan dengan proses pengembangan sebuah aplikasi. Artifact digunakan untuk menggambarkan kebutuhan sebuah sistem yang dibuat, seperti panduan pengguna atau hasil dari aplikasi ketika aplikasi tersebut dijalankan. Biasanya untuk menentukan apakah sebuah artifact dapat digunakan sebagai dasar sebuah aplikasi, UML harus bisa merepresentasikan sebuat artifact memiliki konteks yang jelas. Maksudnya, sebagian besar model menggunakan artifact sebagai contoh awal dari pengembangan aplikasi[10].

H. *RAML (RESTful API Modeling Language)*

RAML adalah cara untuk mendeskripsikan skema praktikan API RESTful agar lebih mudah untuk dipahami oleh manusia dan komputer. RAML dikatakan sebagai praktikal karena RAML untuk saat ini mengikuti aturan standar pengembangan RESTful API dalam mendeskripsikan *resources*, *methods parameters*, *response*, *media types* dan komponen HTTP yang membentuk dasar untuk API pada saat ini. Keuntungan menggunakan RAML adalah pengembang, *partner* dan konsumer API dapat lebih mudah memahami struktur dan spesifikasi dari RESTful API yang tersedia sehingga pengembangan API kedepannya juga akan lebih mudah[11].

I. *Unit Testing*

Unit testing terbagi menjadi dua, yaitu JUnit dan JWebUnit, berikut penjelasan dari masing-masing unit testing tersebut.

1) JUnit: JUnit memiliki API yang sederhana yang menggunakan anotasi untuk memungkinkan untuk memasang sebua framework kedalam kode pengujian. JUnit didesain untuk menguji setiap unit terkecil dari *code* pada aplikasi yang sedang dikembangkan. Semua kelas dan tampilan tersedia pada paket "junit.jar". JUnit disediakan sebagai perpustakaan pada eclipse, tetapi tidak otomati ditambahkan ke *classpath* sebuah proyek[12].

2) JWebUnit: JWebUnit menyediakan API Java untuk pengendalian sebuah aplikasi web yang dikombinasikan dengan satu set pernyataan untuk memverifikasi kebenaran aplikasi. Itu semua termasuk nafigasi melalu link, formulir pendaftaran dan pengajuan, validasi isi tabel, dan fitur lainnya yang terdapat pada aplikasi web.

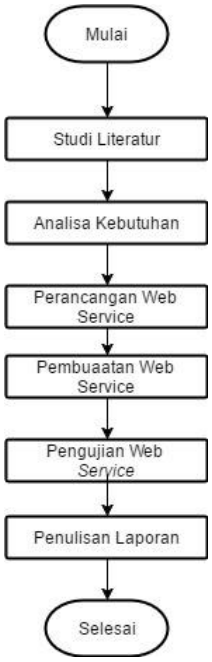
e-ISSN: 2252-7036
Vol.2 No.1 2017: 1-8

Method navigasi sederhana dan asersi yang siap digunakan, memungkinkan untuk melakukan uji lebih cepat daripada hanya menggunakan Junit[13].

III. METODE PENELITIAN

A. *Diagram Alir Penelitian*

Dalam melakukan sebuah penelitian diperlukan adanya metode yang sistematis sehingga penelitian dapat berjalan sesuai dengan yang diharapkan. Penulis menggunakan metode seperti gambar 1 berikut.



Gambar 1. Diagram Alir Tahapan Penelitian

Metode penelitian ini dimulai dengan proses studi literatur yang terkait. Pada tahap ini yang dilakukan adalah pengumpulan literature-literatur serta kaijan yang berkaitan dengan penelitian yang sedang dibuat.Kajian-kajian tersebut dapat diambil dari buku, internet, jurnal dan sumber-sumber lain yang terkait dengan penelitian ini. Setelah tahap ini dilanjutkan dengan analisa kebutuhan, Pada tahap analisa kebutuhan ini yang dilakukan adalah melakukan pengumpulan data rekam medis yang ada pada beberapa tempat praktik pribadi dokter di Banda Aceh. Data rekam medis yang telah dikumpulkan dijadikan acuan apa saja data yang akan disimpan pada layanan web yang akan dibuat. Layanan web ini dibangun berdasarkan rancang bangun aplikasi rekam medis untuk praktik pribadi dokter berbasis android dan memanfaatkan layanan web[14].

Pada tahap selanjutnya yang dilakukan adalah Pada tahap ini dilakukan adalah layanan web berdasarkan dari analisa kebutuhan. Untuk pembuatan layanan awan terdiri dari persiapan laptop dan penginstalan aplikasi pendukung. Selanjutnya dilakukan pembuatan rancangan layanan web menggunakan RAML diaman format datanya berupa JSON, perancangan datastore memakai *class* diagram, dan component diagram digunakan untuk struktur aplikasi. Setelah selesai pembuatan layanan web dilanjutkan dengan pengujian layanan web, pada tahap ini hal yang akan diuji adalah semua *method* dan *property* pada setiap *class* yang dibuat. Pengujian menggunakan Junit untuk semua *class*, dan untuk menguji web digunakan JWebUnit. Pengujian dikatakan berhasil apabila tidak ada lagi *error* atau *failure* yang muncul pada *ouput* uji ketika proses pengujian selesai dijalankan. Setelah semuanya selesai barulah dapat dilakukan penulisan laporan.

B. *Kebutuhan Sistem Pada Penelitian*

1) Bahan Penelitian: Adapun bahan penelitian yang digunakan adalah data yang ada pada rekam medis serta info pasien yang diambil dari beberapa tempat praktik pribadi dokter kesehatan yang ada di Banda Aceh. Data yang sudah terkumpulkan akan digunakan sebagai acuan untuk pembuatan layanan web.

2) Alat Penelitian: Penelitian ini membutuhkan alat – alat penelitian sebagai berikut:

- Komputer dengan spesifikasi yang cukup untuk menjalankan perangkat lunak Google App Engine SDK pada sistem operasi Linux atau Windows.
- Google App Engine SDK sebagai *runtime* untuk menjalankan *backend local server* untuk aplikasi yang akan dibangun
- Eclipse sebagai software pengembangan *Backend Server*

IV. HASIL DAN PEMBAHASAN

A. *Analisa Kebutuhan*

1) Artifact Model: Analisa kebutuhan pada pengembangan penelitian ini berdasarkan pada penelitian [14]. Data yang akan disimpan pada aplikasi rekam medis ini didapatkan dari artifact-artifact yang ada, antara lain kartu dokter, kartu pasien, rekam medis dan resep. Untuk lebih jelasnya dapat dilihat pada artifact model dibawah ini.

- Kartu Dokter
  - Aktifitas : Masuk ke klinik sebagai dokter
  - Artifact : kartu dokter sebagai anggota klinik
  - Fungsinya : sebagai pengenal bahwa dia dokter yang bekerja pada klinik tersebut
  - Pemakaian : ditempel/disematkan pada baju
  - Masalah: Kartu lupa dibawa dan tulisan pada kartu sudah luntur

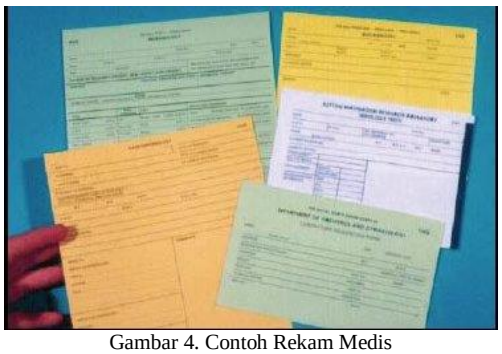
e-ISSN: 2252-7036
Vol.2 No.1 2017: 1-8

Gambar 2. Kartu Dokter

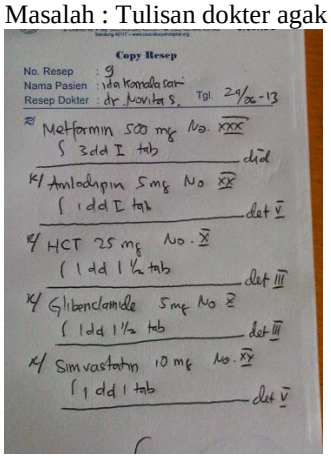
- Kartu Pasien
  - Aktifitas : Sebagai anggota klinik
  - Astifact : kartu identitas pasien pada sebuah klinik
  - Fungsinya: Sebagai kartu anggota sebuah klinik dan syarat untuk mendapatkan layanan pada klinik
  - Pemakaian: Ditunjukkan ketika mau konsultasi
  - Masalah: kartu hilang

Gambar 3. Contoh kartu pasien

- Rekam medis
  - Aktifitas : Penyimpanan data sesi pemeriksaan
  - Artifact : Kartu rekam medis
  - Fungsi : Untuk menyimpan rekam medis pasien
  - Pemakaian : Diisi oleh dokter
  - Masalah: kartu robek dan kartu mengalami kerusakan karena terkena air

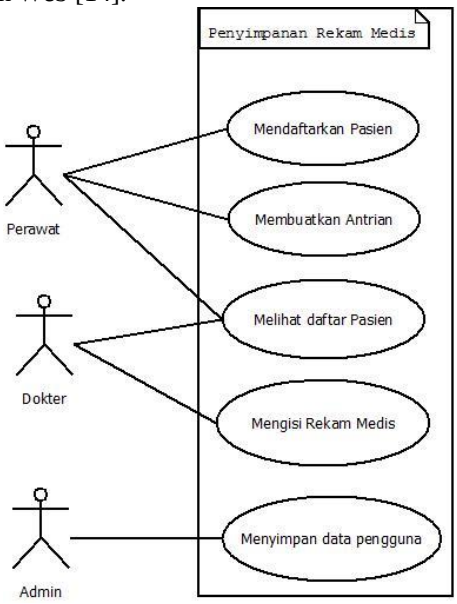

Gambar 4. Contoh Rekam Medis

- Resep
  - Aktifitas: Menulis obat untuk pasien
  - Artifact : Resep obat pasien
  - Fungsi : Mencatat resep obat pasien dan sebagai dasar pembayaran konsultasi dengan dokter
  - Pemakaian : Diisi oleh dokter
  - Masalah : Tulisan dokter agak susah dibaca



Gambar 5. Contoh resep obat

2) Use Case Diagram: Berikut ini adalah use case diagram yang diperoleh dari informasi penelitian yang dilakukan oleh Iwansyah Putra dengan judul penelitian Rancang Bangun Aplikasi Rekam Medis Untuk Praktik Pribadi Dokter Berbasis Android Dan Memanfaatkan Layanan Web [14].



Gambar 6. Use case diagram penyimpanan rekam medis[14]

B. Perancangan

1) RESTful API: Google Endpoint menggunakan skema RESTful API untuk digunakan oleh aplikasi mobile android. Pada Google Endpoint data di kirim dalam format JSON. Sebelum sebuah layanan langsung dibuat, sebaiknya dilakukan sebuah perancangan, hal ini dimaksudkan supaya layanan yang dibuat tidak ada yang salah dan bisa menghindari pembuatan layanan yang tidak diperlukan pada aplikasi.

2) Rancangan Format Data Web Service : Untuk perancangan data web service yang di gunakan adalah

e-ISSN: 2252-7036
Vol.2 No.1 2017: 1-8

JSON. Dengan menggunakan JSON kesalahan dan kekurangan pada aplikasi yang dibuat dapat diminimalisir.

C. Implementasi

Google app engine pada aplikasi ini terdiri dari *model* dan *controller*.

1) Model: *Model* merupakan database dari aplikasi ini yang berbentuk tabel. Pada kode 1, *password*() memiliki property seperti email() serta method setter () dan getter(). Pada *model* terdapat beberapa anotasi seperti @PersistenceCapable yang menandakan bahwa *class* yang memiliki anotasi ini harus disimpan ke datastore, SuppressWarnings("unused") yang berarti untuk menghilangkan tanda warning bahwa *constrructor* yang menggunakan anotasi ini tidak pernah digunakan, @Persistent yang menandakan bahwa variabel yang menggunakan anotasi ini disimpan sebagai property pada *class* tersebut dan @PrimaryKey menandakan property yang memiliki anotasi ini merupakan *Key* dari semua entitas yang ada pada sebual *class*.

```
@PersistenceCapable
publicclassDokter{
@PrimaryKey
@Persisten
(valueStrategy=IdGeneratorStrategy.IDENTITY)
      privateKeykey;
      @Persistent
      Private String get email;
      ….
Private byte[]password;
      @Persistent
Private byte[]salt;

@SuppressWarnings("unused")
      privateDokter(){
      }
```

Kode 1 Contoh kode model

2) Controller: Controller hanya diperlukan untuk menambah fungsi CRUD, yaitu tambah, ubah, hapus dan cari. Pada contoh kode 2 adalah fungsi dokter baru. Pertama yang dilakukan adalah pembuatan *property* yang akan digunakan oleh *class* dokter. Selanjutnya dilakukan pembuatan key untuk *class* dokter. Setelah semua selesai dilakukan pengaksesan ke *datastore* untuk penyimpanan semua *property* yang di buat. Setelah melakukan penyimpanan ke *datastore*, akses ke *datastore* ditutup.

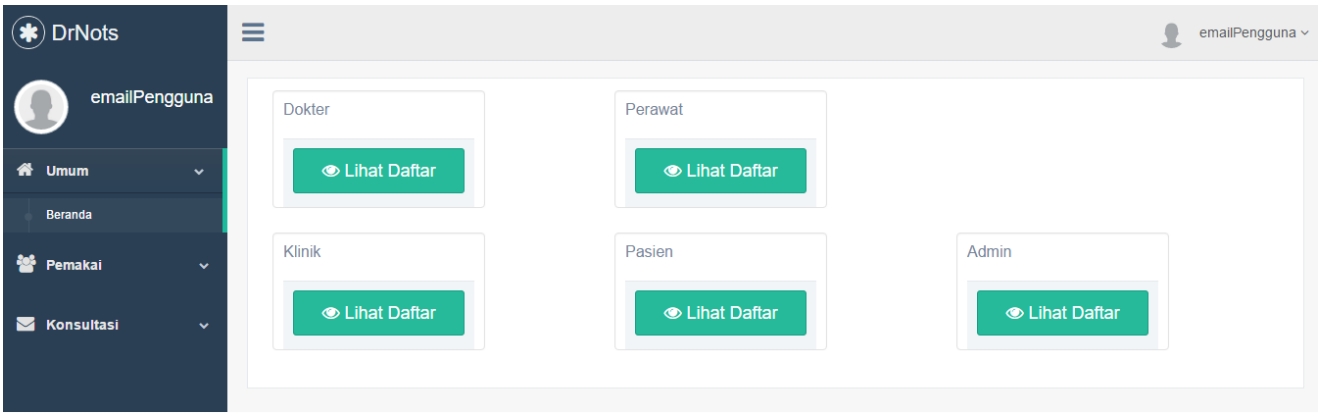
```
      public Dokter baru(String
      namaLengkap, String email,
      String noHp,String alamat,
      String password,long peran,
      Stringspesialis,StringgelarDepan
      , String gelarBelakang, String
      Konsultan,Boolean statusAktif)
      throws
      DuplikasiEmailException,
      NoSuchAlgorithmException,
      InvalidKeySpecException {
         // Buatkan Key dari entitas
      dari id Dokter yang diberikan
         Pemakai pemakai = new
      AturPemakai().baru(namaLengkap,
      email, noHp,alamat, password,
      Peran.DOKTER);

         Dokter dokterBaru = new
      Dokter(pemakai, spesialis,
      gelarDepan, gelarBelakang,
      konsultan, statusAktif);

         Key key = pemakai.getKey();
         // Minta factory untuk buat
      PersistenceManager untuk
      mengakses Datastore
         PersistenceManager pm =
      PMF.getInstance().getPersistence
      Manager();
         try {
         // Simpan di Datastore
      pm.makePersistent(dokterBaru);
      pemakai =
      pm.getObjectById(Pemakai.class,
      key);

        pemakai.setKeyPeran(dokterBaru.
      getKey());
                } finally {
         // Tutup koneksi ke Datastore
                     pm.close();
               }
         // perubahan pada object yang
      dikembalikan tidak akan
         // mengubah nilai di
      Datastore karena koneksi telah
      ditutup.
         return dokterBaru;
   }
```

Kode 2 Contoh *controller* untuk method baru()

3) Google end point: Pengimplementasian Google Endpoint pada aplikasi adalah dengan membuat *class* layanan. Sebelum membuat layanan terlebih dahulu harus dibuat *Model* dan *controller* terlebih dahulu. Pada *Class* layanan menggunakan anotasi @Api, dimana anotasi ini menyatakan *Class Web Service*, pada anotasi terdapat beberapa konfigurasi diantaranya name yaitu nama dari layanan, title yaitu judul yang akan muncul pada explorer, version yaitu versi dari layanan yang aka dibuat dan description yaitu penjelasan ringkas tentang API.

Pada *Class* ini juga terdapat anotasi @ApiMethod yang dipanggil menggunakan httpMethod. @ApiMethod memiliki sebuah konfigurasi yaitu *name*, dimana fungsi dari *name* ini adalah nama method dari layanan. Semua parameter pada *Class* layanan ini diberikan anotasi @Named. Anotasi @Named berfungsi untuk mengirimkan nama parameter yang akan dikirimkan ke API.

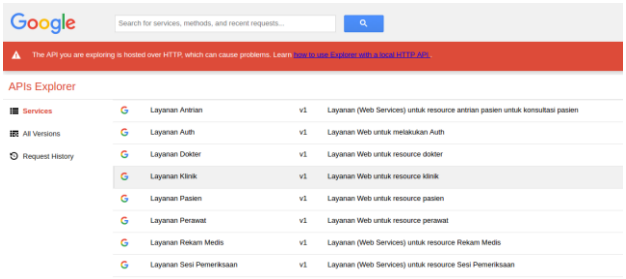
e-ISSN: 2252-7036
Vol.2 No.1 2017: 1-8



Gambar 8. Tampilan beranda ke aplikasi rekam medis untuk administrator

Pada class ini juga terdapat beberapa method, diantaranya baru(), cari(), hapus(), dan ubah(). Method baru() berfungsi untuk membuat klinik baru, method cari() berfungsi untuk mencari klinik berdasarkan id-nya, method hapus() berfungsi untuk menghapus klinik dan method ubah() berfungsi untuk mengubah data klinik. Berikut ini contoh kode dari Google Endpoint, yaitu layananKlinik.

Setelah selesai pembuatan *Class* layanan selanjutnya *Class* tersebut akan diuji. Untuk mengujinya harus menggunakan chrome klusus dengan flag, yaitu dengan cara masuk ke *Directory* chrome, dan jalan kan chrome dengan cara "chrome.exe--user-data-dir=test--unsafely-treat-insecure-origin-as-secure=http://localhost:8888". Hasil yang akan muncul halaman API Explorer dapat dilihat pada Gambar 7.



Gambar 7. Halaman API Explorer Setelah semua layanan

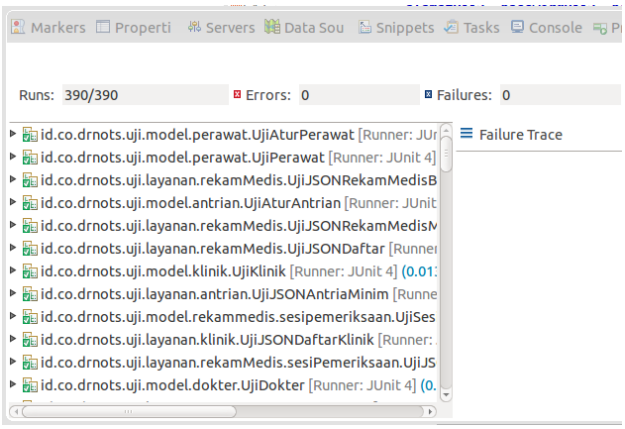
4) JSP dan Servlet: Pengembangan aplikasi web dilakukan dengan menggunakan Servlet dan *Java Server Page* (JSP) yang berjalan pada aplikasi Google App Engine, Servlet digunakan untuk mengatur logika program dan menampilkan halaman pada aplikasi web bagi pengguna *administrator*. Servlet mengatur *routing*, session dan pengolahan parameter HTML untuk digunakan oleh JSP. JSP bekerja layaknya HTML biasa, hanya saja berjalan dan dikompilasi pada server. Servlet dan JSP yang merupakan program dan skrip berbasis Java yang telah dikompilasi akan dikirimkan ke *front-end* aplikasi web menjadi file dokumen web biasa, sehingga pengguna tidak dapat mengganggu logika program yang berjalan pada server.

Untuk tampilan aplikasi web, tampilan ditambahkan fitur responsif menggunakan Bootstrap dan menggunakan *dashboard* admin sumber terbuka sehingga pengguna dapat menggunakan aplikasi dengan baik menggunakan perangkat dengan ukuran layar peraangkat yang berbeda-beda. Tampilan aplikasi *web* yang disediakan bagi pengguna *administrator* dapat dilihat pada Gambar 8. Setelah berhasil diuji pada localhost, selanjutnya semua project dapat di *deploy* ke google app engine(GAE). Hal tersebut dilakukan supaya aplikasi dapat digunakan secara daring.

D. Pengujian

Pada gambar 9 dapat dilihat bahwa banyaknya uji yang dilakukan adalah 390 kondisi, tidak ada errors ataupun failure. Kondisi error terjadi karena masih ada kesalahan pada pengujian atau pada kode yang sedang diuji, sedangkan kondisi failure terjadi karena kesalahan semantik (kesalahan yang dilakukan oleh pembuat kode).

Uji yang pertama dilakukan adalah untuk class model, setelah tidak ada kesalahan pada class model, barulah dapat dilanjutkan ke controller. Apabila tidak ada terjadi failures ataupun error pada *class controller* ataupun *class model*, barulah dapat dilanjutkan pembuatan API, karena apabila masih ada *error* atau *failure* pada *class model* dan *class controller* untuk pembuatan API dipastikan akan ada yang salah. Setalah API selesai dibuat, maka API harus diuji juga dengan JUnit.



Gambar 9. Hasil Run As JUnit Test

e-ISSN: 2252-7036
Vol.2 No.1 2017: 1-8

V. KESIMPULAN

Adapun beberapa kesimpulan dari karya ilmiah ini adalah sebagai berikut:

- Layanan web sudah dapat diakses oleh aplikasi rekam medis praktik pribadi dokter yang telah dikembangkan, sehingga sudah dapat digunakan oleh dokter dan perawat.
- Layanan web sudah dapat memenuhi kebutuhan pada aplikasi rekam medis praktik pribadi dokter, yaitu sudah dapat mengakses datastore yang tersimpan Google App Engine (GAE).
- Admin menggunakan aplikasi web untuk mengelola data klinik serta data pengguna dokter dan perawat yang akan diguakan oleh aplikasi rekam medis praktik pribadi dokter.
- Junit digunakan untuk menguji semua kode yang telah dibuat dan Jweb unit untuk menguji web, kedua uji ini digunakan untuk memastikan apakah aplikasi web dan *mobile* sudah berjalan dengan baik.

UCAPAN TERIMA KASIH

Dengan selesainya karya ilmiah ini penulis mengucapkan banyak terimakasih kepada Orang tua dan keluarga yang telah banyak memberikan bantuan, doa, semangat dan dukungan selama ini. Bapak Rahmad Dawood S.Kom., M.Sc. dan Ibu Roslidar S.T., M.S.Tc.E selaku Dosen Pembimbing I dan Dosen Pembimbing II yang telah menyediakan banyak waktu luang, tenaga, dan pikiran untuk mengarahkan saya dalam penyusunan skripsi ini. Bapak Ardiansyah, BSEE., M.Sc dan Dr. Taufiq A Gani, S.Kom., M.Eng.Sc selaku Dosen Penguji I dan Dosen Penguji II, serta Bapak Ahmadiar S.T., M.Sc. selaku Ketua Sidang.

Bapak Zulhelmi S.T.,M.Sc selaku pembimbing akademik. Bapak Dr. Nasaruddin, ST., M.Eng selaku Ketua Jurusan Teknik Elektro Fakultas Teknik Universitas Syiah Kuala. Bapak Dr. Ir. Mirza Irwansyah, MBA., MLA., selaku Dekan Fakultas Teknik Universitas Syiah Kuala. Teman-teman mahasiswa teknik elektro, khususnya angkatan 2011 dan seluruh pihak yang telah ikut campur tangan dalam membantu penyusunan karya ilmiah ini.

Dalam penyusunan karya ilmiah ini penulis sangat berterima kasih kepada semua pihak yang telah banyak membantu semoga ALLAH SWT dapat membalas kebaikan, semoga karya ilmiah ini memiliki manfaat untuk perkembangan ilmu pengetahuan.

REFERENSI

[1] S. Gondodiputro, "*Rekam Medis Dan Sistem Informasi Kesehatan Di Pelayanan Kesehatan Primer* (PUSKESMAS)." 2007.

[2] Mentri Kesehatan Republik Indonesia, "Peraturan Mentri Kesehatan Republik Indonesia Nomor 269/MENKES/PER/III/2008 Tentang Rekam Medis," 2008.

[3] W. O. Zalukhu, *Pengelolaan Rekam Medis Dalam Upaya Peningkatan Pelayanan Pada Rumah Sakit Umum Gunungsitoli*. Medan, 2010.

[4] M. Kalin, *Java Web Services: Up and Running*, Second edition. Sebastopol, California: O"Reilly, 2013.

[5] E. Belina P, *Perancangan Dan Implementasi Aplikasi E-Learning Versi Mobile Berbasis Android*. Medan, 2013.

[6] Krishnan, S.P.T., Gonzalez, J.L.U., 2015. *Getting Started with Google Cloud Platform, in: Building Your Next Big Thing with Google Cloud Platform*. Springer, pp. 13–25.

[7] L. Richardson and Mike Amundsen, *RESTful Web APIs*. Farnham: O"Reilly, 2013.

[8] Lee, S., 2012. *Unified Modeling Language (UML) for Database Systems and Computer Applications. Int. J. Database Theory Appl*. 05, 157–164.

[9] Sanderson, *Programming Google App Engine*, 2nd ed. Sebastopol, California: O"Reilly, 2013.

[10] Pilone D and Pitman Neil, *UML 2.0 In A Nutshell*. O'Reilly, 2005.

[11] "About RAML | RAML." [Online]. Available: http://raml.org/about/about-raml. [Accessed: 07-Mar-2017].

[12] D. Parsons, *Foundational Java*. London: Springer London, 2012.

[13] "jwebunit."[Online]. Available : https://jwebunit.github.io/jwebunit/. [Accessed: 07-Mar-2017]

[14] Putra Iwansyah,2017. "*Rancang Bangun Aplikasi Rekam Medis Untuk Praktik Pribadi Dokter Berbasis Android Dan Memanfaatkan Layanan Web(Skripsi)".* Universitas Syiah Kuala, Banda Aceh.