Sistem Pembacaan Kode Pos yang Terintegrasi dengan Pencarian Alamat pada OpenStreetMap

Adrian Evan, Leo Willyanto Santoso, Rudy Adipranata
Program Studi Informatika, Fakultas Teknologi Industri, Universitas Kristen Petra
Jln. Siwalankerto 121 – 131 Surabaya 60236
Telp. (031)-2983455, Fax. (031)-8417658
yohanesadrianevan@gmail.com, leow@petra.ac.id, rudya@petra.ac.id

ABSTRAK

Kode pos merupakan sekumpulan data yang merujuk pada suatu alamat dengan pembagian wilayahnya secara rinci (tingkat provinsi, kota/kabupaten, kecamatan, desa/kelurahan) bahkan hingga tingkat blok RT/RW yang terdapat pada setiap *digit* nya kecuali tingkatan RT/RW. Namun data ini cukup sulit untuk diolah pada *map database* karena pembagian tingkatannya yang cukup dalam dan terpilah-pilah sementara pada umumnya *digital maps* hanya mampu mengolah data *longitude*, *latitude* serta informasi alamat hingga tingkat kota dalam bentuk *plain text*.

Untuk mengatasi masalah kurangnya korelasi dan potensinya untuk saling melengkapi antara *database system* dengan *postal data system*, maka digabungkanlah keduanya menjadi satu sistem yang utuh. Dengan perkembangan *web* yang semakin mumpuni dengan adanya HTTPS dan *Progressive Web Apps*, proses inti dari pencarian tempat yaitu *geocoding*, *geolocation*, dan *reverse geocoding*, dapat berfungsi lebih optimal tanpa harus mengorbankan keamanan dan kenyamanan pengunaan secara keseluruhan.

Hasil pengujian menyimpulkan bahwa proses penyatuan sistem ini sudah cukup meningkatkan ketepatan dalam proses pencarian alamat, selama diatur oleh pihak yang sesuai dan kompeten. Tidak dapat dipungkiri bahwa pengetahuan dasar akan sistem pemetaan serta sistem informasi geografis diperlukan disini untuk mengkaji seberapa jauh tingkatan pencatatan alamat pada peta yang sudah ada. Kegunaan sistem ini akan lebih sesuai apabila kodepos diterapkan secara total dalam proses *query* nya, kemudian dengan tambahan fitur cerdas lainnya seperti pelacakan kondisi lokasi dan penentuan rute terbaik dari daftar yang ada serta kondisi saat ini secara *real-time*.

Kata Kunci: kode pos, *Progressive Web Apps, longitude, latitude, geocoding, geolocation, reverse geocoding*

ABSTRACT

Postal code consist of sequential data which refers to a certain address in detail to their smallest form such as rural areas that are seperated by its own respective digits except the tiniest rurals. On the contrary, these pieces of data cannot be used conveniently on map database due to its thorough categorization and filtering process. Some of them may uncorrelated and may cause misunderstanding as of commonly existing digital maps merely processing longitude, latitude, also address data to city level simply using plain text.

To solve the database system and postal data system disjoint problem, these two separate yet potentially powerful systems has to be integrated. With the continuous web development and securer HTTPS protocol into practice, also putting Progressive Web Apps into consideration, these core address finding functions namely geocoding, geolocation, also reverse geocoding majorly functions optimally without any need to concern users' convenience on using this merged system at any sort.

Test results conclude that this merged system has been significantly improving location finding accuracy, as long as the system works within the right hands. Basic knowledges of mapping system and GIS are inevitably mandatory to unleash existing maps capability of location finding into deeper levels. This will operates even optimally when the postal data system are wholefully implemented into the querying process, in addition to another smart features, such as shortest/fastest route tracking also traffic/riot detection into play.

Keywords: *postal code, Progressive Web Apps, longitude, latitude, geocoding, geolocation, reverse geocoding*

1. PENDAHULUAN

Meskipun banyak sekali inovasi dari penemuan teknologi yang sudah ataupun sedang diteliti, tidak dapat dipungkiri bahwa masih terdapat celah-celah yang menimbulkan masalah baru ataupun berakibat permasalahan yang masih ditemukan disitu. Contoh konkrit mengenai hal ini antara lain masih ditemukannya kasus dimana GPS tidak merujuk pada alamat yang semestinya, tidak bekerja secara lebih baik dari metode konvensional seperti penafsiran, penjelajahan oleh pelaku sendiri, mana saja yang harus dilewati serta tempat-tempat yang mengelilingi lokasi yang dimaksud [8]. Bentuk implementasi lain dari teknik penelusuran alamat via kode pos dengan metode yang terstandarisasi, dimana setiap digit memiliki makna dan pembacaan data secara spesifik, terorganisir, dan terkelompokkan sesuai kegunaan dan fungsinya [2], dan digunakan secara baik oleh beberapa negara.

Metode yang terdapat pada GPS pada umumnya memiliki cara kerja yang cukup teknis meskipun pertanggungjawabannya masih sangat minim, dimana hanya menggunakan penarikan dua titik (x,y) dari perhitungan luas bidang datar yang sudah teridentifikasi [7] akan berdampak pada pekerjaan yang berulang karena sudah pasti ditemukan lebih dari satu objek pada satu tempat juga harus memperhitungkan jarak antara kedua objek yang berdekatan, kemungkinan *collision* dan *interference* antar partikel/bahan penyusun objek yang berpengaruh terhadap pelacakan suatu titik tempat.

2. LANDASAN TEORI

2.1. *Location Precision* serta *Longitude* & *Latitude*

Longitude dan *latitude* merupakan posisi pertemuan kedua titik dari dua posisi *point of view* (x,y) suatu objek. Sementara angka

desimal yang mengikutinya menunjukkan jarak antar titik yang mengelilinginya, dimana tiga *digit* setara dengan satu daerah kota, delapan *digit* dapat menemukan kawanan semut yang sedang menggerombol, serta sebanyak dua belas *digit* diibaratkan seperti menghitung jarak antar helai rambut. Sementara GPS pada umumnya dapat mendeteksi posisi pergerakan *smartphone* yang sedang berpindah (nomaden) sejumlah empat *digit* desimal. [7]

GPS selain sebagai penyedia informasi tambahan, GPS juga memiliki *radius* pencarian koordinat hingga 100m dari posisi pencarian, dengan tingkat ketepatan 2m per pembagian *range* dalam *radius* yang sama. Hal ini disebabkan karena desain antena yang tidak semestinya pada *smartphone* (mengingat ukuran dan *form factor*-nya) sehingga mengakibatkan pengurangan ketepatan pengukuran secara signifikan dan ketidakmampuan pelacakan pada wilayah dengan beberapa objek yang berdekatan. (*interference* yang tidak tertolerir mengingat kemampuan antena pada *smartphone*) [7]

2.2. Perbandingan OpenStreetMap dengan Google Maps

Perbandingan mendasar antara keduanya yang juga merupakan salah satu faktor terpenting mengacu pada kinerja kedua sarana ini adalah akses akan data peta dan data secara geografis. Berdasarkan acuan Open Database License (ODbL) menyebutkan bahwa data yang tertera pada OpenStreetMap harus terbuka dan dapat digunakan oleh semua pengguna tanpa kecuali (termasuk untuk pengguna biasa dan donatur) sehingga upaya untuk monopoli data oleh pihak yang tidak bertanggung jawab dan penyalahgunaannya untuk tujuan apapun dapat dihindari sebaik mungkin. [9]

Memang secara *interface* kurang memadai karena bukan merupakan tujuan utamanya, namun data yang tersedia selalu diperbaharui sehingga akan menjadi jarang terjadi kesalahpahaman dan pengguna akan selalu mendapatkan informasi yang tepat mengenai lokasi yang dimaksud. Layaknya GitHub, setiap perubahan data peta yang terjadi dipantau dalam tingkatan *version control* yang mengakibatkan semua pengguna dapat ambil bagian dalam pengembangan daerah secara spesifik per masing-masing yang dikerjakan. [3]

2.3. Perbandingan Wi-Fi *Round-Trip Time* dengan GNSS *Measurement*

Wi-Fi *round-trip time* (Wi-Fi RTT) dengan GPS API (GNSS *measurement*) perlu digunakan secara bersamaan untuk mencapai tingkat akurasi setinggi mungkin. Cara kerjanya masing-masing akan dijelaskan secara bertahap. Wi-Fi RTT menghitung waktu yang diperlukan dalam proses pertukaran data (kirim terima) antara *access point* (*wifi router*) dengan *device* mempertimbangkan jarak antara keduanya. [13] Secara alamiah, sinyal radio memiliki kecepatan perpindahan yang setara dengan kecepatan cahaya, sehingga jika dipatok pada satu titik tidak akan menghasilkan tingkat akurasi yang berarti. Maka dalam percobaan ini perlu dilakukan beberapa titik/posisi yang berbeda, dengan menghitung waktu antar prosesnya (*send*, *fine timing measurement*; FTM; jarak antara *device* dengan *router*, *acknowledge*, mencari rentang sinyal dengan waktu kali kecepatan cahaya bagi dua) termasuk memperhitungkan apakah ada yang menghalangi jalur perjalanan sinyal. (*signal interference from buildings*)

2.4. Perbandingan API dengan *Web Service*

Web service merupakan segelintir bagian dari *software* yang tersedia secara awan melalui jaringan internet (*cloud*, *online*) dan terstandarisasi melalui XML *parsing*/*encoding*. [14] Pesan dikirim berupa *request* dan dikembalikan berupa *response*. *Web service* terikat dengan aturan/protokol SOAP dan XML-RPC.

Sementara API, sesuai istilahnya, lebih ditekankan pada *client side* dibandingkan secara *server* (*procedure call* dan lainnya) dan bekerja dalam satu sistem HTML yang fleksibel dan sistem/protokol lainnya yang mendukung fleksibilitas dari API (RESTful, protokol GET dan POST) sehingga dapat memproses beberapa bahasa pemrograman sekaligus seperti Java, Phython, Ruby. *Output* yang dihasilkan juga lebih beragam, selain XML juga dapat berupa JSON. (*database parsing*)

Kesamaan antara keduanya, keduanya mampu memproses *template* data XML (dibaca oleh mesin pembaca), namun JSON lebih mudah dan lebih *familiar* oleh kalangan pengguna. Selain itu, API juga memerlukan proses pekerjaan yang lebih sedikit sehingga kinerja layanannya otomatis meningkat. Karena karakteristik yang demikian, API lebih cocok digunakan untuk tujuan *mobility* (*smartphone*, *tablet*, memerlukan interaksi pengguna) berbeda dengan *desktop* yang lebih terbatas dan tertentu dalam mekanisme penerimaan data.

2.5. Google Firebase

Firebase merupakan *real-time cloud database* yang difungsikan untuk bekerja dengan komponen secara *mobile*. Informasi yang tersimpan didalamnya pasti disetarakan dengan semua *user* yang terhubung menggunakannya. (*auto-synchronization*) Salah satu penyebabnya ialah satu jenis output yang ditetapkan menjadi standar/kaidah dalam ranah Firebase, yaitu berupa JSON parsing. [5]

Dibalik kenyamanan tersebut, terkuak didalamnya sebuah problematika yang tak terelakkan. Semua data yang tersimpan pada Firebase harus sudah dalam bentuk tingkatan tunggal yang baku, mengingat *server* sudah mengalami *over workload* dengan melakukan proses sinkronasi kepada semua pengguna yang terhubung secara tepat waktu. Apabila terdapat *nested*/*relational datasets*, masih dapat diakali dengan mengolahnya diluar Firebase, dengan tetap menampilkannya pada Firebase menyesuaikan *format* yang sudah ditentukannya.

2.6. *Progressive Web App*

Progressive Web App atau biasa disebut dengan singkatan PWA merupakan *application-based website* dengan pembentukan strukturnya pada *device* yang lebih sedikit memakan *resource* dibandingkan *native app*. [1]

PWA pada umumnya mendapatkan perlakuan yang sama dengan *native app* dengan beberapa penyesuaian. Hal ini dimungkinkan terjadi karena komponen pada *native app*; *button*, *intent*, *textbox*, dll terjatahkan masing-masing bagiannya, memiliki *fixed size*, dan tersimpan secara *local*, berbeda dengan PWA yang menggunakan komponen HTML *native* sehingga secara *size* sudah *minified* dan untuk beberapa komponen UI lainnya sudah terangkum dalam sepaket *library* yang dapat di-*host*/*load* secara local maupun *hosting server* (CDN).

2.7. *Tiled Maps*

Merupakan kenyataan yang cukup ironis bahwa setiap *digital*/*web maps* yang sudah ada dibuat bukan oleh ahlinya (kartografer) melainkan seorang *server administrator* yang hanya mampu mengakali *template* XML dan menggambarkannya sesuai dengan keterangan yang ada dan menambahkan *icon* yang dirasa perlu untuk melengkapi peta terkait. Hal ini membuat permasalahan serius GIS terkesan hanya masalah remeh yang dapat diselesaikan dengan mudahnya dan cepat. Pemahaman ini cukup berbahaya karena *misinformation* pada peta berakibat penggunanya tersesat atau bahkan mengalami kejadian yang tidak diinginkan. Menyinggung permasalahan ini, penambahan dan penggunaan *tiled maps* dirasa perlu, bahkan bersifat wajib, demi kenyamanan dan keselamatan bersama para *user* nya. [11]

Tiled maps sendiri adalah tingkatan pemetaan, yang setiap tingkatannya terpapar informasi yang unik dan berbeda, yang digambarkan pada petak-petak yang setiap petaknya memiliki ukuran baku. Karena setiap informasinya terkandung dalam setiap petak petanya, maka lebih memungkinkan untuk lebih cepat melakukan perubahan informasi dalam peta dan menyebarkannya kepada pengguna secara *real-time*. Maka dari itu kinerja/performa dari *web maps* sudah pasti pada kurun terbaiknya. Implementasi AJAX juga dapat secara lebih baik digunakan karena pembagian data per *tile* nya, termasuk kemudahan untuk menambahkan informasi berupa gambar per sisi jalan/bangunan/lokasinya.

Satu-satunya kelemahan dari tiled maps ialah keterlibatan pengguna (*user interface*) yang sangat minim. Pengguna harus dengan lapang dada menggunakan segala sarana yang disediakan. Bahkan untuk hal sederhana seperti penataan halaman (*view layout change*) tidak dapat dilakukan karena sudah diatur sedemikian rupa untuk tujuan kinerja dan kemudahan.

2.8. Tinjauan Studi

Sejauh ini belum ada penelitian yang mengkaji secara presisi mengenai peningkatan akurasi pada OpenStreetMap API maupun pemantauan perubahan kondisi tempat/tanah di suatu posisi tertentu.

Sementara dari penelitian yang mendekati, fungsi GIS yang diterapkan antara lain penerapan sistem kategorial yang lebih spesifik, dimana menggunakan Google Maps API untuk mencari tempat wisata untuk rekreasi, ziarah, serta tujuan lainnya. [6]

Pada penelitian lainnya disebutkan bahwa Google Maps API digunakan untuk mencari lokasi SPBU terdekat dengan wilayah cakupan pada kota Jepara dan Kudus dari *waypoint* yang ditetapkan oleh *user* dengan integrasi dengan sistem Node.JS. [16]

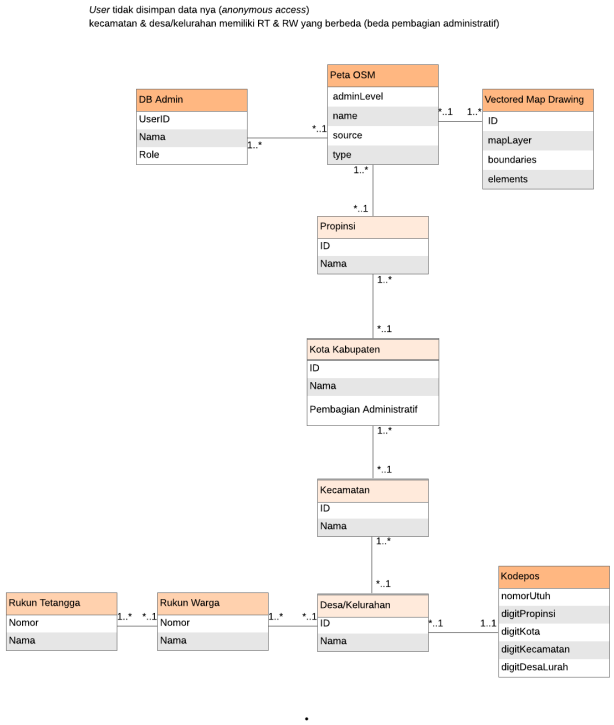
Informasi berharga pada peta dan sistem GIS seringkali disalahgunakan oleh pihak yang tidak bertanggung jawab untuk kepentingannya sendiri, misalnya pada sistem *fake* GPS pada aplikasi ojek *online* seolah-olah menjemput penumpang nyata (anonim, 2018). Maka dari itu, mereka selaku tim peneliti mencetuskan metode dan/atau steganografi dengan menggunakan posisi dan pemetaan koordinat dimana proyeksi nilainya pada *secret bits* pasti bernilai 2c per c *bit* yang dibawanya per pertemuan beberapa titik (*vertices*) [15]

GeoNames *data repositories* dan *library* LibPostal dimana menggunakan *database* pada OpenStreetMap diimplementasikan untuk mempermudah proses pemetaan dan klasifikasi detail suatu daerah hasil *query* ataupun *waypoint*, metode ontologi untuk pemecahan dan penggambaran wilayah beserta detil pemetaannya, dalam penelitiannya yang memiliki tujuan yang serupa dengan penulis. Namun kami memiliki perbedaan yang mendasar dalam pendekatan penelitian ini, dimana penulis mengorelasikan antara sistem kode pos, *real-time* data dari tempat yang dimaksud (termasuk foto fisik tempat dan letaknya pada lokasi dan pemetaan dari peta), serta API pembacaan, proyeksi, serta pemetaan lokasi pada peta *digital* menjadi satu sistem yang terintegrasi [12]

Metode triangulasi Delaunay yang kemudian digambarkan pada segi-lima Thiessen sebagai bentuk proyeksi dan estimasi dari titik posisi koordinat yang sudah ditentukan dalam dua bentuk peta yang berbeda sebagai perbandingan untuk meningkatkan ketepatan dari tempat yang dimaksud yang kemudian nantinya akan diproyeksikan kemungkinan terdekatnya dari beberapa jenis peta serta lokasi aslinya diterapkan pada penelitian lain. [4]

Aplikasi GoBis dapat digunakan untuk memantau posisi bis per trayek, termasuk bis tumpuk (*double decker*), mirkolet, dan bemo. Namun penulis tidak menuturkan bahwa *app* ini dibuat dalam dasar OpenStreetMap dan *by default* tidak dapat memantau *user waypoint*/*pinpoint* dimana fitur ini juga tidak kalah pentingnya. [10]

3. DESAIN SISTEM



.

Gambar 1. *Entity Relationship Diagram*

Pada Gambar 1., dimana berisi tabel *database admin*, terdapat *primary key* pada atribut userid, dimana memiliki jenis penyimpanan data integer dan pencatatan datanya dimulai dari angka 1 pada digit utamanya. *Role entity* berjenis data string, dan *entity* ini dapat diartikan sebagai jabatan dan tanggung jawab yang harus dipenuhi oleh admin terkait. Hubungan *database admin* dengan peta OSM sebagai bentuk pertanggungjawaban atas jatah *map tiles* (dapat menjangkau satu kawasan yang sama, bahkan lintas kawasan) yang diampunya dan termasuk aktivitas

pemeliharaannya apabila terjadi penambahan dan/atau perubahan informasi (termasuk didalamnya ukuran lokasi dan batasannya dengan tempat disekitarnya) dan/atau posisi koordinat lokasi terkait karena GPS *misaccuracy* dan faktor penentu lainnya.

Pada *adminLevel entity* memiliki jenis data *single-digit* integer yang menunjukkan kemampuan dan/atau kewenangannya untuk merevisi setiap detil informasi yang terkandung dalam lokasi terkait dalam tingkatan yang sudah spesifik. (sampai tingkat kecamatan bahkan desa/kelurahan, berdasarkan *level* yang diperoleh seorang admin) *Name entity* menunjukkan bagian yang sedang dikerjakan oleh admin tersebut. (misal kelurahan Siwalankerto) *Source entity* merupakan sumber data untuk bagian peta yang diperolehnya. (berdasarkan hasil survey bahkan dari arsip *offline maps*, atlas, dan sejenisnya) *Type entity* merupakan perlakuan pembatasan yang berlaku/diberlakukan secara *system-wide* oleh OpenStreetMap. (contohnya *boundary type* artinya pembagian wilayahnya dibatasi secara administratif/melalui kewenangan pemerintah) Relasi peta OSM dengan *vectored map drawing* sebagai fungsi estetikanya supaya peta dapat ditampilkan dan digunakan secara lebih tertata. (mengikuti sifat alamiah dari *tiled maps* yang dimana setiap lapisan petanya/*map layers* mengandung informasi yang berbeda dan unik, pembatasan per propinsi, kota, kelurahan, desa/kelurahan, bahkan daerah sekitar *location pinpoint* ditampilkan masing-masing *layer* secara tersendiri) Untuk membuat keterkaitannya menjadi lebih efektif digunakan, tentu harus direlasikan secara definitif. (peta dengan propinsi direlasikan secara langsung, sementara kota sampai desa/kelurahan merupakan *inheritance* dari propinsi, informasinya saling terkait dan setiap elemennya saling bertukar informasi)

Informasi yang terkandung dalam propinsi, kecamatan, dan desa/kelurahan merupakan informasi secara umum/general, artinya informasi yang tersedia sesuai dengan deskripsi/definisi yang tertuang pada *entity* yang ada. Sementara untuk *entity* kota/kabupaten ada unsur yang menjadi pembeda disini. Karena kota dan kabupaten merupakan bagian yang memiliki perbedaan namun mereka setingkat secara tingkatan hirarki, maka atribut pembagian administratif mengambil peran disini, inilah yang menentukan suatu data terkait apakah merupakan kota tersendiri maupun merupakan kabupaten. (turunan dari kota, dimana *scope* nya lebih kecil dan antara kota dengan masing-masing kabupaten dikepalai oleh walikota yang berbeda)

Pada *entity* RW digambarkan bahwa setiap desa/kelurahan dapat memiliki banyak RW, dimana masing-masing RW dapat memiliki banyak RT. RW disini pada umumnya merupakan pembagian/pembatasan wilayah komplek atau perkampungan sedemikian rupa dengan kantor desa/kelurahan sebagai batasnya. Sementara batasan masing-masing cakupan wilayah RT merupakan rumah penghuni dari ketua RT masing-masing.

Pada *entity* kodepos terdapat dua bagian penting yang tersimpan didalamnya, yaitu nomor kodepos secara utuh dan masing-masing informasi yang terkandung pada setiap digit nya. Keduanya memiliki jenis data integer namun memiliki perlakuan yang berbeda karena pembacaan data dan perolehan informasi dari alur relasi yang berbeda pula. (antara kodepos 60234 dengan 60236 dianggap, dan memang dalam implementasi penentuan daerahnya berbeda, meskipun masih mengandung informasi yang sama hingga tingkat kecamatan) Pada contoh ini, 60234 mencakup propinsi Jawa Timur, kota Surabaya, kecamatan Gayungan, namun pada kelurahan Menanggal. Sementara kodepos 60236 mengarah pada kelurahan Siwalankerto. Perbedaan tersebut menggambarkan dua daerah yang berbeda (dua data) meskipun masih tercakup pada cakupan wilayah yang sama.

4. HASIL DAN PEMBAHASAN

4.1. *System Requirements*

Chromium-*based browsers*, yaitu Chrome for Android ataupun Firefox for Android. (hanya perlu memilih salah satu) Kelebihan Chrome antara lain PWA *will retain its functionality as long as user always keep its dependencies up-to-date* (Android System WebView) serta meningkatkan kenyamanan penggunaan secara keseluruhan karena sudah berupa *native app*.

Sementara kelemahannya antara lain Chrome mempunyai *web rendering components* secara terpisah, namun untuk fungsi PWA secara optimal wajib ada Android System WebView (masing-masing komponennya punya kegunaannya tersendiri) dan kehadiran Chrome secara otomatis menonaktifkan WebView.

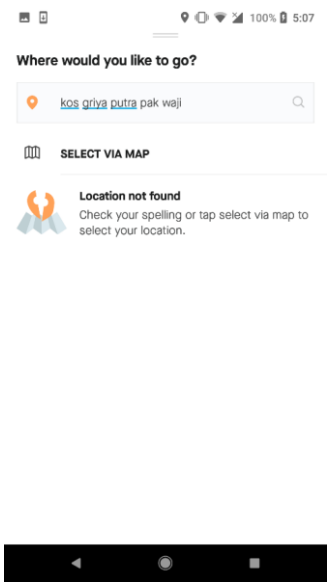
Pada umumnya Chrome sudah dikondisikan oleh *device manufacturers* sebagai *system app* dimana tanpa hak akses khusus (dimana disini yang dimaksud adalah *root access*) tidak dapat disiasati dengan *selective workaround method* yang ada. Dengan kondisi seperti ini yang saling meniadakan, maka PWA tidak dapat berfungsi sebagaimana mestinya dan hanya diperlakukan sebagai *regular webpage* (karena PWA akan berfungsi secara optimal apabila WebView dan Chrome berjalan secara bersamaan)

4.2. Pemasangan Awal dan *Initial Page Load as* PWA

Mula-mula *user* membuka *web* ini pada *browser*, dimana ada sedikit perbedaan UI antara *desktop version* dengan *mobile version*. Posisi *install* pada *desktop* cukup sulit dijangkau karena terlalu kecil dan penempatannya yang kurang strategis. (posisi cenderung mendekati ujung kanan *address bar*) Sementara untuk langkah pemasangan PWA pada *device* lebih *intuitive*. *Browser* menyediakan slot tersendiri diatas *virtual capacitive buttons* pada bagian bawah *browser*. (gambar tidak dapat ditampilkan karena kurangnya bahan dan pemasangan PWA sifatnya tidak dapat diulang) Setelah PWA berhasil dipasang pada tampilan *mobile*, *user* akan disuguhkan dengan menariknya tampilan *splash screen* yang kemudian membawa *user* pada tampilan awal PWA pertama kali dibuka.

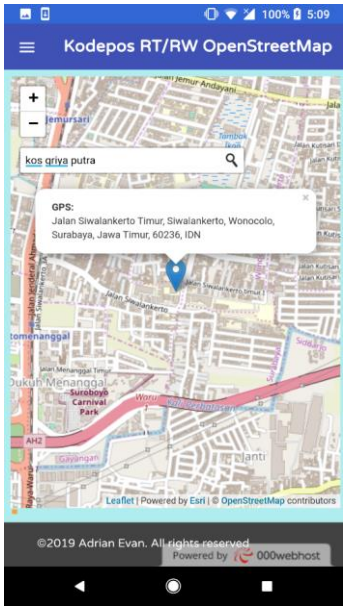
4.3. Perbandingan Performa antara Google Maps, OpenStreetMap dengan Firebase

Penulis dimudahkan dengan banyaknya *app* yang sudah ada yang menerapkan Google Maps untuk fungsi utama petanya. Dengan hal ini, penulis dapat dengan mudah membuat perbandingan untuk tujuan pengujian. Penulis memilih GOJEK untuk rujukan Google Maps, dan masing-masing dilakukan proses *address search* pada kedua *platform*. Tempat yang sudah tercatat akan dilewati agar lebih dapat difokuskan untuk tempat yang belum ada. Kemudian dilakukan kembali metode pencarian alamat yang serupa pada OpenStreetMap, dengan harapan variasi database lebih beragam karena sifatnya merupakan *open source* dan dapat dengan lebih mudah diperbaharui oleh sesama pengguna. Namun hasilnya nihil juga persis seperti metode Google Maps.



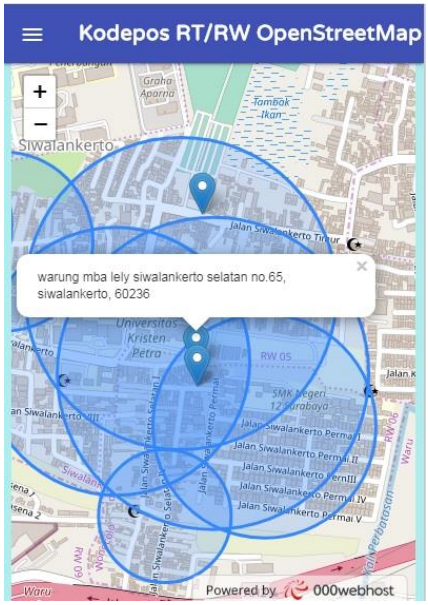
Gambar 2. hasil pencarian alamat via GOJEK

Pada Gambar 2 ditampilkan proses *address search* lewat GOJEK sebagai parameter pembanding dengan krosm PWA. Kemudian dilakukan kembali metode pencarian alamat yang serupa pada OpenStreetMap, dengan harapan variasi database lebih beragam karena sifatnya merupakan *open source* dan dapat dengan lebih mudah diperbaharui oleh sesama pengguna. Namun hasilnya nihil juga persis seperti metode Google Maps.



Gambar 3. hasil pencarian alamat menggunakan OpenStreetMap API secara *embedded* pada PWA

Pada Gambar 3 terlihat hasil pencarian dari Firebase *external database storing and searching*. Untuk kondisi seperti inilah Firebase dapat mengambil peranan didalamnya. Dengan bantuan *reverse geocoding* pada ESRI framework, *user* dapat memproyeksikan secara lebih tepat peletakkan titik koordinat berdasarkan ancar-ancar (*wayguide*) sesuai arah yang dilewati sampai di tempat tujuan. Penulis mengusahakan sebaik mungkin untuk mengurangi *pinpoint complexity* dengan mempersempit *path to destination* dengan mengambil jalur yang lebih sering dilewati. (seperti yang sudah disinggung diawal, penulis tidak mengusulkan NLP dan AI-*based methods* untuk penelitian ini)



Gambar 4. proses pemetaan titik ke tujuan pada Firebase *external search*

Pada Gambar 4 ditunjukkan sistematika pengujian serta pengukuran tingkat akurasi. Pada kasus ini dilakukan dua kali proses pemetaan titik ke tujuan (karena tepat dua titik tempat dan posisinya agak terhimpit masuk gang) dimana untuk kasus seperti ini tingkat akurasinya ½ * 100% = 50% dimana nilai 1 disini merupakan kondisi pencarian yang ideal hanya sekali sementara nilai 2 merupakan jumlah proyeksi titik sampai ditemukannya tempat yang dimaksud.

5. KESIMPULAN DAN SARAN

5.1. Kesimpulan

Berdasarkan hasil pengujian yang sudah dilakukan, dapat disimpulkan bahwa *app* ini memberikan koreksi alamat yang sesuai selama penggunanya memiliki pengetahuan akan sistem pemetaan dan kondisi geografis dari tempat-tempat yang bersangkutan, baik pada posisi tempat yang sebenarnya sampai tingkat *all possible directions* yang mungkin untuk menuju tempat tersebut dapat melalui *shortcuts* ataupun jalan tikus, sehingga *app* ini kedepannya dapat berfungsi pula sebagai *shortest*/*fastest route finder* juga *traffic*/*riot detection* untuk mencapai kenyamanan dan keamanan berkendara secara keseluruhan.

5.2. Saran

- Sistem kodepos seharusnya diterapkan secara menyeluruh untuk dapat lebih meningkatkan ketepatan hingga tingkat RT/RW yang sifatnya sering mengalami perubahan dan sudah cukup berdekatan, karena penulis sementara hanya menggunakan sistem ini sebagai batasan/*boundary* pada daerah/fokus pencarian/analisa penelusuran tempat-tempat baru.

- Menerapkan metode *Natural Language Processing* serta AI-*based methods* lainnya untuk penerapan *app* secara lebih sempurna, seperti pendeteksi kondisi jalan, semua rute yang memungkinkan untuk sampai ke tujuan, serta perubahan bentuk/penempatan lokasi hingga tingkat fisik/blok petak jalan.

6. DAFTAR PUSTAKA

[1] Agarwal, N. 2019. "Progressive Web Apps-Prime Time to Change". WildNet Technologies. Retrieved from URI = https://www.wildnettechnologies.com/

[2] Arya, W., 2017."Apa itu ZIP code", Lifeder Indonesia., URI =https://id.lifeder.com/zip-code-kode-pos/

[3] Buczkowski, A. 2015, October. geoawesomeness. "Why would you use OpenStreetMap if there is Google Maps?" URI=https://geoawesomeness.com/

[4] Chiu, CS. and Wang, D. "Weighted coordinates transformation method for map overlay with non-homogeneous space partition", Computer & Geosciences, vol. 29, 2003.

[5] de Croos, P. 2018. When You Should (and Shouldn't) use Firebase. codementor community. Retrieved from URI = https://www.codementor.io/cultofmetatron/when-you-should-and-shouldn-t-use-firebase-f62bo3gxv

[6] Kusuma, M.E. dan Yanto Budisusanto. "Aplikasi Google Maps API dalam pengembangan Geographic Information System pariwisata berbasis web pada daerah Sidoarjo", GEOID, vol.10, no.2, 2015.

[7] Lenaghan, J. 2014. Location Precision: the Good, the Bad and the Ugly. URI = https://adexchanger.com/data-driven-thinking/location-precision-the-good-the-bad-and-the-ugly/

[8] Lukwira, A.L. 2016. "Pengalaman Buruk Naik Grab dan Gojek". Kompasiana. Retrieved from URI = https://www.kompasiana.com/lukwirandre/56e8d83d45afbd8a0b350132/pengalaman-buruk-naik-grab-dan-gojek?page=all

[9] McDonough, M. 2013. "Google Map Maker vs. OpenStreetMap: Which mapping service rules them all?" In: Eyers D., Schwan K. (eds) Digital Trends. Retrieved from URI = https://www.digitaltrends.com/computing/google-map-maker-vs-openstreetmap-id-editor/

[10] Prima, E. 2018. Surabaya Kembangkan Aplikasi GoBis untuk Pantau Suroboyo Bus. Retrieved from URI = https://tekno.tempo.co/read/1158855/surabaya-kembangkan-aplikasi-gobis-untuk-pantau-suroboyo-bus/full&view=ok

[11] Quinn, S. dan John A. Dutton. Why tiled maps? URI =https://www.e-education.psu.edu/geog585/node/706

[12] Sebastian, N., Vadim Savenkov and Alex Polleres, "Geo-Semantic Labelling of Open Data", Procedia Computer Science, vol. 17, 14th International Conference on Semantic Systems, 2018.

[13] Van Diggelen. F., Want, R. and Wang, W. 2018. How to achieve 1meter accuracy in Android. Retrieved from URI = https://www.gpsworld.com/

[14] Verma, S. 2018. APIs versus web services. Retrieved from URI = https://blogs.mulesoft.com/dev/api-dev/apis-versus-web-services/

[15] Wang, N., Zhang, H., and Men, C. "A high capacity reversible data hiding method for 2D vector maps based on virtual coordinates", Elsevier, 2014.

[16] Zuniar, S.A., Prastyo, R. dan Listiyorini, T. "Pemanfaatan Google Maps API untuk pencarian jalur lokasi SPBU terdekat di kota Jepara dan Kudus dengan teknologi Node.JS", SELISIK 2016.