ARTIKEL ILMIAH PENELITIAN

"RANCANG BANGUN GAME SEDERHANA DENGAN REST WEB SERVICE YANG DITERAPKAN PADA SISTEM OPERASI ANDROID"

Disusun Oleh :

Nama : Budiawan

NIM : A11.2008.04482

Program Studi : Teknik Informatika

FAKULTAS ILMU KOMPUTER

UNIVERSITAS DIAN NUSWANTORO

SEMARANG

2013

RANCANG BANGUN GAME SEDERHANA DENGAN REST WEB SERVICE YANG DITERAPKAN PADA SISTEM OPERASI ANDROID BUDIAWAN

Program Studi Teknik Informatika - S1, Fakultas Ilmu

Komputer, Universitas Dian Nuswantoro Semarang

URL : http://dinus.ac.id/

Email : 111200804482@mhs.dinus.ac.id

ABSTRAK

Perkembangan aplikasi game saat ini sangat pesat, bermula dari aplikasi dengan *user* pada perangkat tunggal, kemudian muncul bentuk permainan dengan banyak *user* melalui banyak perangkat pada permainan yang sama melalui internet atau biasa disebut dengan game online. Salah satu bagian terpenting dari game online adalah distribusi dan sinkronisasi data antara *client* dan *server*. Untuk dapat menerapkan sinkronisasi data yang terintegrasi tersebut, dapat menggunakan *Web Service*. Penggunaan *Web Service* dimaksudkan agar proses *update* data dilakukan sekali pada sisi *server* sehingga lebih hemat waktu dan tenaga. Sedangkan untuk distribusi data dapat menggunakan salah satu arsitektur dari *web service* yaitu REST *web service.* Penerapan *REST web service* mendukung penggunaan XML dan JSON yang mempercepat pertukaran data pada sistem sehingga aplikasi akan lebih cepat dan efisien. Aplikasi game ini dapat dimainkan di perangkat mobile dengan sistem operasi android. Diperlukan koneksi internet untuk menghubungkan server dengan aplikasi game ini. Karena aplikasi ini harus diakses secara online maka koneksi internet mempengaruhi jalannya aplikasi ini. Pembuatan aplikasi game ini menggunakan Eclipse disisi client dan PHPStorm disisi server. Sistem Operasi Android yang digunakan untuk menjalankan aplikasi game ini dengan Android *Gingerbread* 2.3.0. Metode pengembangan perangkat lunak yang digunakan dalam pembuatan aplikasi game ini adalah metode *Extreme Programming*. Pada perkembangan lebih lanjut, aplikasi ini memerlukan sistem enkripsi data untuk masalah keamanan yang dapat mendukung REST *web service*.

Kata Kunci = Game, Rest Web Service, Rest, Web Service, Android

1. PENDAHULUAN

1.1 Latar Belakang

Perkembangan teknologi saat ini khususnya teknologi mobile sudah sangat pesat. Banyak vendor berlomba-lomba meramaikan pasar perangkat *mobil*, mereka memproduksi dan menjual produk tersebut dengan keunggulan pada masing-masing perangkat *mobile*. Mayoritas masyarakat saat ini pun sudah memiliki perangkat mobile. Menurut Nielsen Company (2013), di Amerika Serikat saat ini 60% masyarakat menggunakan perangkat *mobile smartphone*. Bahkan di Hong Kong

pengguna *smartphone* sangat tinggi yaitu mencapai nilai 87%, sedangkan di Indonesia walaupun terbilang masih sedikit yaitu 23% tapi angka tersebut masih terus meningkat mengingat perkembangan perangkat *mobile* dan antusias masyarakat sangat tinggi[1]. Perkembangan pasar *mobile* di seluruh dunia yang mengalami peningkatan yang cukup pesat memacu banyak pengembang aplikasi untuk membuat aplikasi yang menarik pada perangkat *mobile* seperti *smartphone*.

Saat ini terdapat berbagai jenis aplikasi pada perangkat *smartphone*, seperti aplikasi game, hiburan, berita, dan sebagainya. Namun menurut survey Nielsen Company (2011), aplikasi game merupakan aplikasi yang paling populer di kalangan pengguna *smartphone*. Berdasarkan hasil riset Nielsen, 64% pengguna *handphone* memainkan aplikasi game dalam kurun waktu 30 hari. Aplikasi terpopuler berikutnya adalah aplikasi cuaca yang digunakan 60% pengguna *handphone*, disusul aplikasi *social media* dengan 56% dan aplikasi peta dengan 51% pengguna [2]. Hal ini menunjukkan aplikasi game merupakan aplikasi yg paling sering digunakan oleh pengguna *smartphone*, dan merupakan jenis aplikasi yg paling populer di antara berbagai jenis aplikasi pada perangkat *mobile* baik yang diunduh dengan gratis atau berbayar.

Perkembangan aplikasi game saat ini sangat pesat, bermula dari aplikasi dengan *user* pada perangkat tunggal, kemudian muncul bentuk permainan dengan banyak *user* melalui banyak perangkat pada permainan yang sama melalui internet atau biasa disebut dengan game online. Salah satu bagian terpenting dari game online adalah distribusi dan sinkronisasi data antara *client* dan *server*. Bentuk pemrosesan data haruslah dirancang sedemikian rupa sehingga dapat didistribusikan melalui jaringan yang ada kepada seluruh *user* yang ada.

Untuk dapat menerapkan sinkronisasi data yang terintegrasi tersebut, dapat menggunakan *Web Service*. *Web service* dapat dibangun dengan menggunakan bahasa pemrograman apa saja dan juga dapat diimplementasikan pada platform manapun. Karena Web Service merupakan aplikasi logika yang dapat diakses dan dipublikasikan menggunakan standar internet yang dideskripsikan dalam format XML dan diidentifikasi dengan *Universal Resource Identifier* (URI) [3]. Penggunaan *Web Service* dimaksudkan

agar proses *update* data dilakukan sekali pada sisi *server* sehingga lebih hemat waktu dan tenaga.

Sedangkan untuk distribusi data dapat menggunakan salah satu arsitektur dari *web service* yaitu REST *web service*. REST *web service* mendukung penggunaan XML maupun JSON sebagai format pertukaran data [4]. Penelitian lain yang telah dilakukan sebelumnya menunjukkan bahwa format JSON cocok untuk digunakan pada REST *web service* untuk lingkungan *mobile*. Nurzhan Nurseitov, Michael Paulson, Randall Reynolds, dan Clemente Izurieta (2009) telah melakukan penelitian untuk mengukur performa XML dan JSON sebagai format untuk pertukaran data. Hasilnya menunjukkan bahwa JSON lebih cepat dan membutuhkan lebih sedikit resource daripada XML [5]. Penggunaan arsitektur REST *Web Service* dimaksudkan agar *user* dapat terhubung ke *server* dan mempercepat proses pertukaran data.

Dengan kelebihan dari REST *web service* diatas maka dapat dikembangkan sebuah aplikasi game kuis sederhana dengan mengimplementasikan sistem kerja yang ada pada REST *web service* untuk melakukan *update* data dan menggunakan smartphone berbasis sistem operasi Android untuk media pengimplementasian aplikasi ini.

1.2 Tujuan

Berkaitan dengan hal-hal yang telah dijelaskan di atas, maka dpat dikembangkan sebuah aplikasi game kuis sederhana menggunakan sistem operasi Android dengan menerapkan REST web service agar proses *update* data dilakukan lebih efisien.

2. TINJAUAN PUSTAKA

2.1 Web Service

Dalam Microsoft (2000) dinyatakan bahwa *web-service* merupakan tahapan ketiga dari tahapan evolusi ASP (*Application Service Provider*) dimana pada tahapan pertama ditekankan pada penyediaan aplikasi *desktop* sedangkan pada tahapan kedua ditekankan pada penyediaan aplikasi berbasis *client-server*. Pada tahapan ketiga ini, komponen-komponen atau *building blocks software* disediakan sebagai *service* dan disebarkan lewat jaringan internet untuk diintegrasikan dengan aplikasi-aplikasi lain [6].

Menurut Kreger (2001) *web-service* diartikan sebagai sebuah antar muka (*interface*) yang menggambarkan sekumpulan operasi-operasi yang dapat diakses melalui jaringan, misalnya

internet, dalam bentuk pesan XML [7]. *Web-service* dapat dibangun dengan menggunakan bahasa pemrograman apa saja dan juga dapat diimplementasikan pada platform manapun. Hal ini dimungkinkan karena *web-service* berkomunikasi menggunakan sebuah standar format data yang universal yaitu XML dan menggunakan protokol SOAP. Karena *web-service* menggunakan format data XML, maka *web-service* juga mewariskan sifat *multitier* dari XML sehingga memungkinkan terjadinya integrasi antar *web-service* atau aplikasi. Pada sistem *multi-tier*, aplikasi maupun dokumen XML dapat dilewatkan ke pihak lain dan diolah oleh pihak tersebut.

Dalam sistem ini dimungkinkan suatu aplikasi dapat mengambil data dari satu sumber tanpa harus tahu bahwa sebenarnya data tersebut dihasilkan melalui proses pengolahan oleh sistem lain sehingga dapat terjadi integrasi data maupun aplikasi yang sering disebut dengan A2A (*application to application*). Dalam Kreger (2001) dinyatakan juga bahwa model dari sebuah *web-service* didasarkan pada interaksi antara 3 komponen yang berperan dalam *web-service*, yaitu: *service provider, service registry* dan *service requestor/consumer* [7]. Interaksi yang terjadi antara ketiga komponen tersebut juga melibatkan operasi *publish, find* dan *bind*. *Service provider* menyediakan service yang dapat diakses melalui jaringan komputer, misalnya internet. Kemudian, *service provider* mendeskripsikan service yang dibangun dan mem-*publish*-kan *service description* tersebut ke *service registry* atau secara langsung ke *service consumer*. *Service requestor/consumer* menggunakan operasi *find* untuk mendapatkan *service description* secara lokal maupun melalui *service registry*. *Service description* yang diperoleh itu kemudian digunakan untuk mem-*bind service provider* dan berinteraksi dengan implementasi *web-service* yang akan digunakan tersebut.

Al Shahwan dan Moessner (2010) dan HostBridge Technology (2009) mengelompokkan *web service* berdasarkan arsitektur yang digunakan dalam implementasinya yaitu REST dan SOAP-based *web service*[5].

2.2 REST Web Service

Istilah REST yang merupakan singkatan dari *Representational State Transfer* pertama kali digunakan oleh Roy Thomas Fielding, salah seorang pelopor proyek *web server* Apache, pada disertasi doktornya yang berjudul *Architectural Styles and the Design of*

Network-based Software Architectures di University of California pada tahun 2000. Fielding (2000) mengidentifikasi empat prinsip (*constraints*) dalam REST, yaitu [8]:

1. *Resource Identification*

   Semua *resource* (serta *state*nya) yang berhubungan dengan aplikasi diberikan *identifier* yang unik dan *identifier* tersebut harus bersifat global. Konsep *resource* disini bukan hanya hal statis yang langsung berhubungan dengan aplikasi namun juga termasuk informasi yang dibutuhkan seperti dokumen transaksi.

   REST *resource* adalah semua hal yang bisa diakses dan ditransfer melalui web antara *client* dan server. Dan karena protokol yang digunakan untuk berkomunikasi adalah HTTP, berbagai macam tipe file bisa ditransfer, teks file, *flash movie*, gambar dll. Sehingga dalam REST *system representasi* dari *resource* tergantung dari tipe yang diminta *client* (*MIME type*) yang didefinisikan didalam protokol *request*.

2. *Uniform Interface*

   Semua interaksi dibangun dengan antarmuka yang seragam. REST *web service* menampilkan semua *resource* dan interaksinya dengan *interface* yang seragam, tidak seperti RPC yang menampilkan fungsi yang ada melalui *method* yang bisa dipanggil secara *remote*. Dalam REST *web service* untuk *uniform interface* ini menggunakan *Uniform Resource Identifier*(URI). URI pada REST *web service* berupa *hyperlink* terhadap *resource* meskipun REST *constraint* tidak menyatakan URI harus berupa *hyperlink*, namun karena teknologi yang digunakan pada web service adalah web sehingga URI berupa *hyperlink*. Jika menggunakan teknologi lain, REST URI tentu akan berupa hal yang berbeda, namun tetap berupa *address* terhadap sebuah *resource*.

3. *Self-Describing Message*

   Untuk setiap interaksi dengan *resource* melalui antarmuka yang seragam, REST membutuhkan representasi dari *resource* yang menggambarkan semua aspek penting yang dimiliki oleh *resource* tersebut. Representasi dari *resource* sendiri adalah semua hal yang dikirim antara *cilent* dan *server*. Representasi merupakan *state*

sementara dari data sebenarnya yang terletak di suatu tempat penyimpanan. Dengan kata lain representasi merupakan *stream* biner besama *metadata* yang menjelaskan bagaimana *stream* tersebut digunakan baik untuk *client* maupun untuk *server*. Bisa terdapat banyak jenis *client* yang me-*request resource* yang ada, oleh karena itu representasi setiap *client* pun dapat berbeda. Representasinya dapat berupa gambar, text file, *stream* XML atau *stream* JSON, tapi kesemua representasi tersebut harus tersedia melalui URI yang sama. Untuk kasus *request* yang dilakukan oleh manusia (*human user*) biasanya representasi berupa laman web sehingga menjadi bentuk representasi yang dapat dibaca.

4. *Stateless Interaction*

Setiap interaksi antara *client* dan *server* harus memiliki *state* sendiri (atau dengan kata lain tidak dipengaruhi *session client*). Jadi *server* hanya akan memantau *resource state* bukan *client session*.

2.3 Android

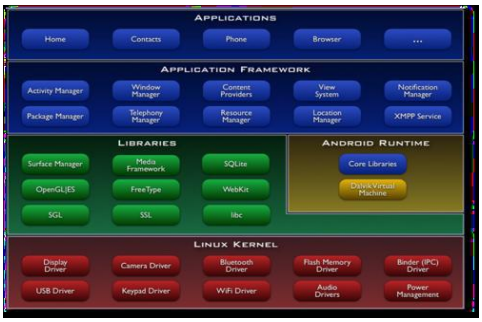
Android adalah sistem operasi untuk telepon seluler yang berbasis Linux. Android menyediakan *platform* terbuka bagi para pengembang buat menciptakan aplikasi mereka sendiri untuk digunakan oleh bermacam peranti bergerak. Awalnya, Google Inc. membeli Android Inc., pendatang baru yang membuat peranti lunak untuk ponsel. Kemudian untuk mengembangkan Android, dibentuklah *Open Handset Alliance*, konsorsium dari 34 perusahaan peranti keras, peranti lunak, dan telekomunikasi, termasuk Google, HTC, Intel, Motorola, Qualcomm, T-Mobile, dan Nvidia.

Pada Juli 2000, Google bekerjasama dengan Android Inc., perusahaan yang berada di Palo Alto, California Amerika Serikat. Para pendiri Android Inc. bekerja pada Google, di antaranya Andy Rubin, Rich Miner, Nick Sears, dan Chris White. Pada saat perilisan perdana android, 5 November 2007, Android bersama *Open Handset Alliance* menyatakan mendukung pengembangan standar terbuka pada perangkat seluler. Di lain pihak, Google merilis kode–kode Android di bawah lisensi Apache, sebuah lisensi perangkat lunak dan standar terbuka perangkat seluler

Di dunia ini terdapat dua jenis distributor sistem operasi android. Pertama yang mendapat dukungan penuh dari Google atau *Google Mail*

Services (GMS) dan kedua adalah yang benar-benar bebas distribusinya tanpa dukungan langsung Google atau dikenal sebagai *Open Handset Distribution* (OHD).

Diagram berikut menunjukan komponen utama dari sistem operasi Android :



Gambar 2.1: Arsitektur Android

1. Linux Kernel

   Seperti dapat dilihat pada gambar Linux Kernel menyediakan *driver* layar, kamera, *keypad*, Wifi, *flash memory*, *audio*, dan IPC (*interprocess communication*) untuk mengatur aplikasi dan keamanan. Kernel juga bertindak sebagai lapisan abstrak antara *hardware* dan *software*.

2. *Libraries*

   Android menyertakan *libraries* C / C++ yang digunakan oeleh berbagai komponen dari sistem android.

3. Android *Runetime*

   Android terdiri dari satu set *core libraries* yang menyediakan sebagian besar fungsi yang sama dengan yang terdapat pada *core libraries* bahasa pemrograman Java. Setiap aplikasi menjalankan prosesnya sendiri dalam android, dengan mesin virtual Dalvik (Dalvik VM).

4. Aplikasi *Framework*

   Pengembang memiliki akses penuh menuju API *framework* yang sama dengan yang digunakan oleh aplikasi inti. Arsitektur aplikasi dirancang agar komponen dapat digunakan kembali (reuse) dengan mudah. Setiap aplikasi dapat memanfaatkan kemampuan ini (sesuai dengan batasan keamanan yang didefinisikan *framework*). Mekanisme yang sama memungkinkan komponen diganti oleh pengguna.

5. *Application*

   Android telah menyertakan aplikasi inti (*native*) seperti *email client*, *map*, SMS, kalender, dan lainnya. Semua aplikasi tersebut ditulis menggunakan bahasa pemrograman Java. Pada *layer* inilah *developer* menempatkan aplikasi yang dibuat.

Saat ini operasi android memiliki beberapa versi, versi terbaru (sampai tulisan ini dibuat) adalah versi 4.3 (*Jelly Bean*).

3. METODE PENGUMPULAN DATA

Dalam Pengumpulan data dilakukan dengan cara kajian pustaka mengenai teori – teori dan data-data yang terkait berupa buku, jurnal, dan artikel dalam web. Teori tersebut diantaranya mengenai perancangan dan dasar – dasar penerapan REST *web service* pada game android.

4. HASIL DAN PEMBAHASAN

4.1 Analisa Kebutuhan User

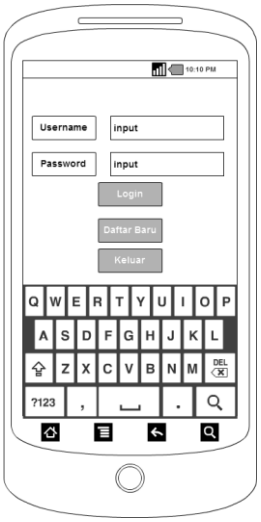
Dalam pembangunan aplikasi Game kuis sederhana yang menerapkan REST *web service* berbasis Android tentunya diperlukan analisa apa saja yang dibutuhkan oleh user dalam hal ini masyarakat yang menggunakan aplikasi ini. User memerlukan suatu sistem yang mencakup:

1. Mengambil data pertanyaan dan skor dari *server*.
2. Menampilkan pertanyaan seputar sejarah Indonesia.
3. Menampilkan daftar pertanyaan dan jawabannya setelah game berakhir.
4. Menampilkan nilai skor tertinggi dari *server*.

4.2 Desain

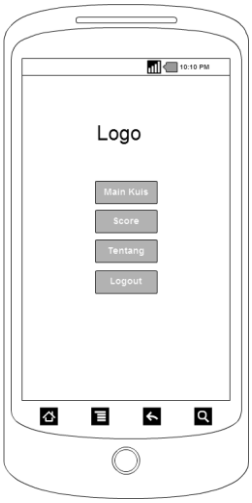
Proses desain akan menerjemahkan syarat kebutuhan ke sebuah perancangan perangkat lunak yang dapat diperkirakan sebelum dibuat *coding*. Proses ini berfokus pada : struktur data, arsitektur perangkat lunak, representasi *interface*, dan detail (algoritma) prosedural. Tahapan ini akan menghasilkan dokumen yang disebut *software requirment*. Dokumen inilah yang akan digunakan *proggammer* untuk melakukan aktivitas pembuatan sistemnya.

Dalam proses ini juga akan menghasilkan *desain interface* yang nantinya digunakan untuk membuat tampilan dari aplikasi game yang akan dibuat. Berikut adalah tampilan *desain interface*-nya :

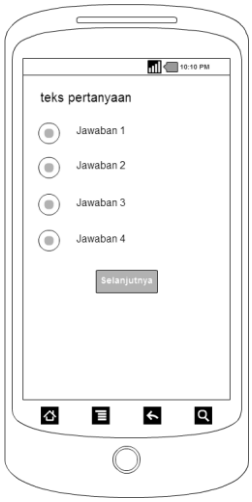


Gambar 4.1 : Desain Halaman Login

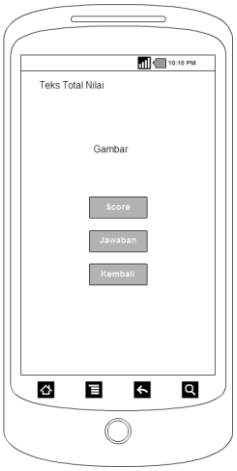
Gambar 4.2 : Halaman Registrasi



Gambar 4.3 : Menu Utama



Gambar 4.4 : Halaman Main Kuis



Gambar 4.5 : Selesai Game

Gambar 4.6 : Halaman Score

5. PENUTUP

5.1 Kesimpulan

Setelah melakukan analisis merancang dan mengimplementasikan REST *Web Service* pada aplikasi game kuis sederhana pada *handphone* Android, diperoleh kesimpulan bahwa diterapkannya REST *Web Service* membuat aplikasi dapat diakses secara online. Selain itu mempercepat pendistribusian data dari server ke client. Tetapi karena aplikasi ini harus

diakses secara online maka koneksi internet mempengaruhi jalannya aplikasi ini.

5.2 Saran

Berdasarkan kesimpulan diatas, maka ada beberapa saran yang dapat diberikan dalam pengembangan aplikasi ini, yaitu :

1. Untuk pengembangan lebih lanjut dalam proses penyimpanan data, diperlukan sistem enkripsi data untuk masalah keamanan yang dapat mendukung REST *web service*. Sehingga data yang terdapat pada server dapat terjamin keamanannya.
2. Aplikasi game sederhana ini hanya dapat digunakan pada sistem operasi Android saja. Diharapkan pada pengembangan berikutnya, aplikasi ini dapat dikembangkan pada sistem operasi mobile lainnya jelas.

DAFTAR PUSTAKA

[1] http://www.nielsen.com/us/en/newswire/2013/whos-winning-the-u-s-smartphone-market-.html diakses pada 21 oktober 2013.

[2] http://www.nielsen.com/us/en/newswire/2011/games-most-popular-mobile-app-category.html diakses pada tanggal 25 September 2013.

[3] Prasetyo, Hendro J. 2005. *Implementasi Service Oriented Architecture (SOA) Menggunakan Teknologi Web Service.* Skripsi. Universitas Widya Dharma.

[4] Kumar Pavan, Otti P. 2011. On the Design of Web Services: SOAP vs REST University of North Florida.

[5] HostBridge Technology. 2009. *SOAP and REST : Choosing formal and informal Web services for CICS integration*. JSON team. Intoducing to JSON.

[6] Microsoft Corp. (2000). *Application Service Provider: Evolution and Resources*. White Paper, USA.

[7] Kreger, H. (2001). *Web-services Conceptual Architecture (WSCA 1.0).* IBM Software Group, USA.

[8] Fielding, Roy Thomas (2000). *Architectural Styles and the Design of Network-based Software Architectures*. Disertasi Doktoral. University of California, Irvine.

[9] Adams, Ernest (2010). *Fundamentals of game design 2nd edition*. Berkeley: New Riders Publishing.

[10] Scheel, Jesse. (2008). *The art of game design: a book of lenses*. Burlington: Morgan Kaufmann Publishers