RANCANG BANGUN SISTEM ANTREAN MULTI FUNGSI YANG RESTFUL DENGAN MENGGUNAKAN *CREDIT CARD-SIZED COMPUTER*

Sadr Lufti Mufreni *)

Program Studi Teknologi Informasi Fakultas Sains dan Teknologi Universitas 'Aisyiyah Yogyakarta,
Jalan Siliwangi (Ring Road Barat) 63, Mlangi, Nogotirto, Gamping, Sleman,
D.I. Yogyakarta, 55292, Indonesia

*)*E-mail: sadr@unisayogya.ac.id*

Abstrak

Universitas 'Aisyiyah Yogyakarta menerima mahasiswa baru yang mengalami peningkatan dari tahun ke tahun. Pada tahun 2017 terdapat 18.000 pendaftar dengan jumlah yang diterima sebesar 1829. Banyaknya pendaftar menimbulkan kesulitan untuk mengantre, tidak ada informasi kapan antrean diproses, dan fokus petugas akademik terganggu. Penelitian ini dibuat untuk mengatasi masalah tersebut. Sistem antrean ini mudah dimodifikasi, bersifat modular, dan berbiaya rendah. Penelitian menggunakan metode *waterfall model* sebagai *software process model* dan bahasa analisis *Unified Modeling Language*(UML) untuk membantu pembuatan dokumentasi, spesifikasi, dan analisis sistem. Digunakan perangkat keras Raspberry Pi dengan Raspbian sebagai sistem operasi. Representation State Transfer (REST) dan JavaScript Object Notation (JSON) digunakan sebagai standar komunikasi antara sistem antrean dan pengguna. Implementasi menggunakan bahasa pemrograman Java, basis data Hypersonic SQL, dan Restlet yang merupakan *framework* REST di Java. Sistem diuji secara fungsional dengan skenario pendaftaran mahasiswa baru menggunakan Hypertext Markup Language (HTML) dan Javascript. Pengujian berhasil bekerja menyelesaikan masalah antrean. Sistem antrean membantu proses penerimaan mahasiswa baru sehingga pihak Universitas dapat memberikan layanan yang prima untuk calon mahasiswanya. Selain itu biaya pengadaan komputer untuk sistem antrean dapat ditekan. Diharapkan dengan kemudahan modifikasi, penelitian ini bisa dijadikan langkah awal pembuatan Sistem Informasi Kesehatan di Universitas 'Aisyiyah Yogyakarta.

Kata kunci: sistem, antrean, dimodifikasi, murah, REST

Abstract

University of 'Aisyiyah Yogyakarta accepts new students that increases each year. In 2017 the registered students were 18.000 and 1829 were qualified. The registrants' large number causes problems, i.e. difficulty in queue, lack of queue information and the disrupted focus. This research address the queue problem. This queue system is easy to modify, modular, and low cost. The research uses waterfall model as the software process model and the Unified Modeling Language (UML) as the standard analysis language in documentation, specification, and system analysis. Raspberry Pi is used with Raspbian as the operating system. Representation State Transfer (REST) and JavaScript Object Notation (JSON) are used as communication standards between queuing systems and clients. Implementation uses Java, Hypersonic SQL database, and Restlet as the REST framework in Java. Blackbox testing is used with student enrollment scenarios using Hypertext Markup Language (HTML) and Javascript. Testing works successfully to resolve queue issues. The queuing system assists the admission to process registration, thus the University can provide excellent service to its prospective students. Moreover, computer procurement cost for the queue system can be minimized. With easy modification feature, this research is useful for developing Health Information System in the University of 'Aisyiyah Yogyakarta.

Keywords: system, queue, customizable, low cost, REST

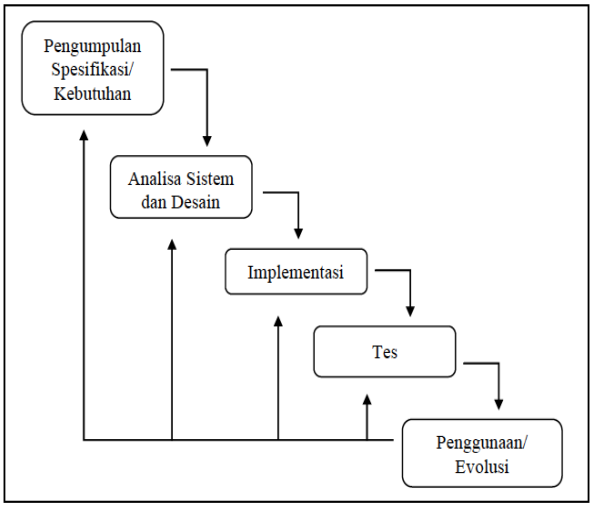
1. Pendahuluan

Universitas 'Aisyiyah Yogyakarta (UNISA) merupakan salah satu universitas swasta di Yogyakarta yang berdiri pada tahun 1963. Jumlah mahasiswa UNISA meningkat dari tahun ke tahun. Jumlah mahasiswa pada tahun 2016 adalah 1.851, dan pada tahun 2017 adalah 1.829 dengan pendaftar sejumlah 18.000. Jumlah pendaftar yang banyak menimbulkan permasalahan, yaitu pendaftar mengalami kesulitan untuk mengantre, pendaftar tidak mengetahui kapan data pendaftar akan diproses, dan fokus petugas akademik dalam memproses pendaftaran sering terganggu

karena harus melayani pendaftar yang menanyakan kapan data mereka akan diproses. Untuk mengatasi permasalahan di atas diperlukan sistem antrean.

Penelitian sistem antrean yang sudah ada [1][2] berfokus pada penggunaan di tempat tertentu (seperti Unit Pelayanan Mahasiswa atau antrean pasien di rumah sakit). Penelitian menggunakan PC sebagai server, dan SMS *gateway* dan gawai (*client*) untuk berkomunikasi dengan pengguna. Namun, penelitian [1][2] memiliki beberapa kekurangan, antara lain sistem antrean perlu dibuat ulang untuk setiap kebutuhan yang berbeda.Selain itu, terdapat relasi *client-server* yang erat, sehingga jika ada perubahan di server perlu mengubah *client*. Sistem tersebut juga menggunakan PC sebagai server, yang biayanya relatif mahal.

Dengan melihat kekurangan yang ada, penelitian ini bertujuan untuk membangun sistem antrean yang dapat digunakan untuk kebutuhan yang berbeda. Relasi *client-server* didesain renggang (modular), sehingga perubahan di server tidak perlu mengubah client. Selain itu, penggunaan Raspberry Pi sebagai server dipilih yang biayanya relatif lebih murah dari PC.



Gambar 1. Waterfall Model

2. Metode

Metode menggunakan beberapa tahap berdasarkan *waterfall model*[3], seperti ditunjukkan pada Gambar 1. Tahapannya adalah mengumpulkan spesifikasi atau kebutuhan dari sistem, diteruskan dengan analisis sistem dan desain keseluruhan sistem. Selanjutnya, dilakukan implementasi dari analisis dan desain. Setelah itu, dilakukan tes pada sistem yang dibuat. Pada tahap akhir, penggunaan sistem dan evolusi dari sistem jika dibutuhkan.

2.1. Pengumpulan Spesifikasi

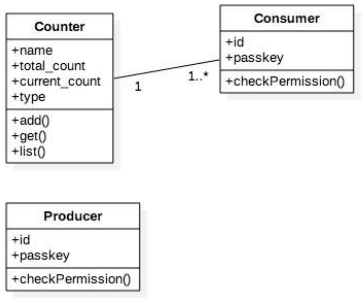
Pengumpulan spesifikasi dilakukan dengan 2 cara :
1. Pengamatan langsung di lapangan dan di beberapa tempat yang mempunyai masalah yang sama (seperti bank, rumah sakit).
2. Wawancara dengan pihak akademik

2.2. Analisis sistem dan desain

Analisis menggunakan bahasa UML[4]. *Class Diagram* menggambarkan kelas yang terlibat dalam pembuatan sistem, *use case* menggambarkan fungsi sistem dan *sequence diagram* menggambakan interaksi di dalam sistem.

2.2.1. Class Diagram

Gambar 2 berikut ini merupakan kelas dan interaksi dalam sistem yang dibuat.



Gambar 2. Class Diagram Dari Sistem

2.2.2. Use Case

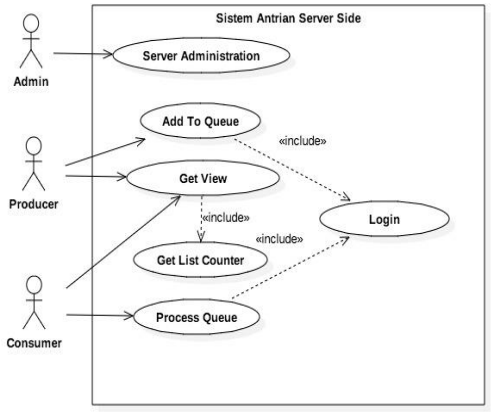
Berdasarkan pengumpulan spesifikasi, dihasilkan *use case* sebagai berikut (Gambar 3):

Pengguna(*client*) dibagi menjadi 3 yaitu :
1. Admin : mengelola konfigurasi ang dibutuhkan untuk menjalankan server
2. Consumer : pihak universitas
3. Producer: calon mahasiswa baru/orang tua mahasiswa/wali mahasiswa

- *Use Case* Server Administration
  Skenario : admin mengelola server dengan konfigurasi yang sudah ditentukan
  Alternatif : konfigurasi yang digunakan salah
- *Use Case* Get List Counter
  Skenario : sistem menampilkan semua counter yang telah didefinisikan pada konfigurasi
- *Use Case* Login
  Skenario : producer/consumer ingin menggunakan fungsi tertentu yang memerlukan hak akses, sistem meminta konfirmasi dengan menanyakan pengguna dan kata kunci

Alternatif : pengguna dan kata kunci tidak sesuai

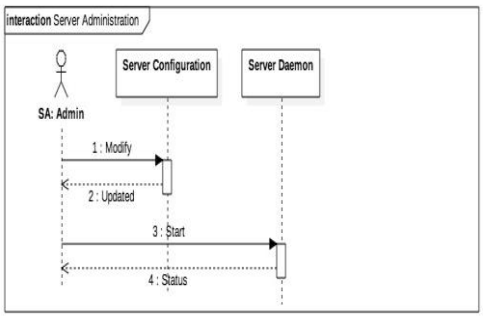
- *Use Case* Get Producer View
  Skenario : producer mengakses tampilan Producer dengan menggunakan url tertentu
- *Use Case* Add To Queue
  Skenario : producer ingin menambahkan antrean pada salah satu counter yang terdapat pada use case Get List Counter
- *Use Case* Get Consumer View
  Skenario : consumer mengakses tampilan Consumer dengan menggunakan url tertentu
- *Use Case* Process Queue
  Skenario : consumer mengambil antrean dari salah satu counter yang terdapat pada use case Get List Counter



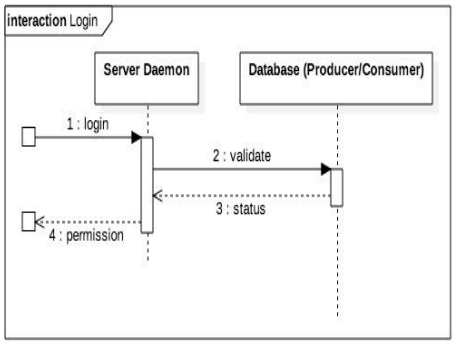
Gambar 3. Use Case Dari Sistem

2.2.3. Sequence Diagram

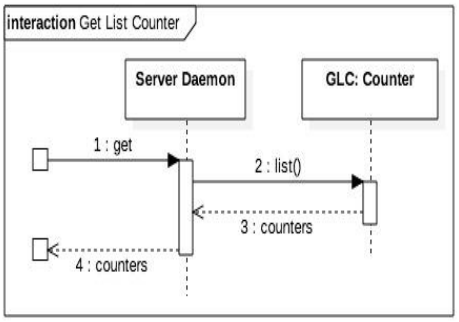
Berikut penjabaran dari *use case* dalam bentuk *sequence diagram*, yang ditunjukkan pada gambar 4 – 9.



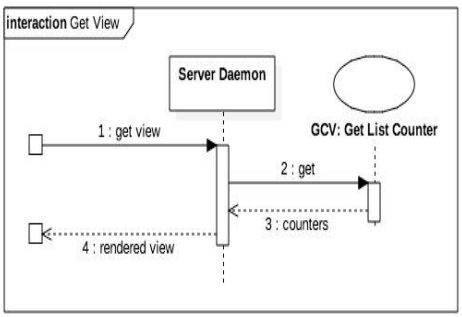
Gambar 4. Sequence Diagram Dari Server Administration



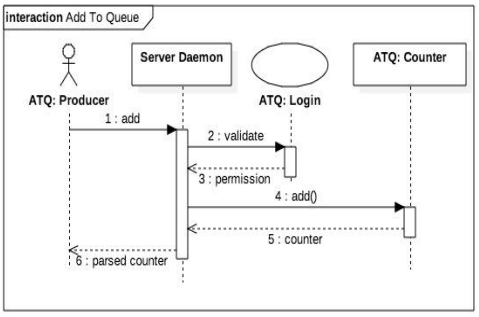
Gambar 5. Sequence Diagram Dari Login



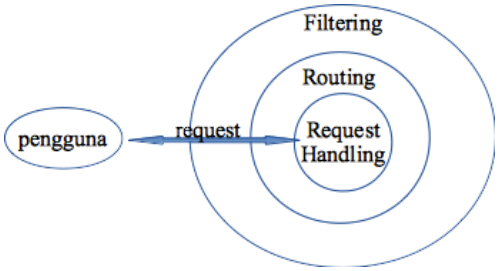
Gambar 6. Sequence Diagram Dari Get List Counter



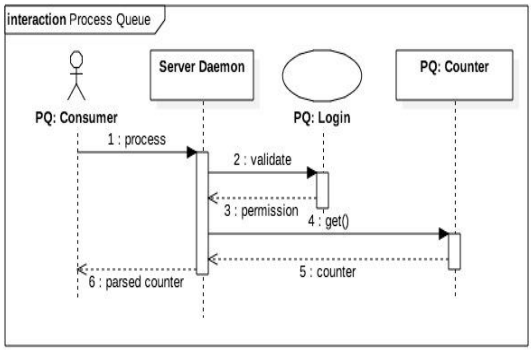
Gambar 7. Sequence Diagram Dari Get View



Gambar 8. Sequence Diagram Dari Add To Queue



Gambar 9. Interaksi REST dengan pengguna di system



Gambar 10. Sequence Diagram Process Queue

2.3. Implementasi Sistem

Implementasi terbagi dalam beberapa sub-seksi.

2.3.1. Raspberry Pi

Raspberry merupakan komputer mini seukuran kartu kredit yang dapat menjalankan sistem operasi seperti layaknya komputer berukuran "normal". Untuk menjalankan sistem operasi tetap dibutuhkan layar monitor, papan ketik, dan tetikus[5]. Model yang digunakan adalah Raspberry Pi Model B+ dengan spesifikasi prosesor ARM 700 MHz, 512MB RAM, 100MB Ethernet, 4xUSB[6]. Harganya sekitar Rp. 750.000,00 pada saat penelitian ini dilakukan

2.3.2. Raspbian

Raspbian adalah sistem operasi berbasis Linux yang berawal dari Debian. Sistem operasi ini sudah dioptimasi sehingga dapat berjalan baik pada Raspberry Pi[6].

2.3.3. Java

Java adalah bahasa pemrograman berbasis objek. Java dirancang oleh James Gosling dan dibuat oleh Sun Microsystem (sekarang dimiliki oleh Oracle Corporation). Keunggulan dari Java adalah *write once, run everywhere* [7], tentunya dapat berjalan dengan baik pada Raspbian.

2.3.4. Hyper SQL Database (Hsqldb)

Hsqldb adalah basis data relasional yang dibuat menggunakan Java[8]. Kelebihan dari basis data ini adalah

1. Mendukung SQL 2011.
2. Ukuran yang relatif kecil.
3. Dapat dibenamkan ke dalam aplikasi.
4. Dapat berjalan secara *in-memory* atau disimpan di file.

Berdasarkan kelebihan di atas, hsqldb dipilih untuk implementasi dari sistem.

2.3.5. Representation State Transfer (REST)

Untuk membuat sistem yang modular, diperlukan standar untuk berkomunikasi. Standar komunikasi tersebut adalah REST. Keunggulan dari REST adalah menggunakan standar internet yang sudah ada yaitu Hypertext Transport Protocol (HTTP) untuk berkomunikasi. Fokus REST adalah interaksi komponen dan datanya, dan tidak membatasi terhadap implementasi dan protokol dari komponen[9]. RESTful adalah aplikasi yang memenuhi persyaratan REST.

Sistem mengimplementasikan REST berdasarkan *use case* dengan alamat sebagai berikut :

1. counter/add
2. counter/get
3. counter/list

2.3.5.1. Restlet

Restlet adalah implementasi REST dengan menggunakan bahasa Java. Restlet mempunyai struktur[10] sebagai berikut, seperti ditunjukkan pada Gambar 10 :

1. *Filtering* berfungsi menyaring *request*.
2. *Routing* berfungsi mengarahkan *request*.
3. *Handling* berfungsi memproses request.

2.3.6. JavaScript Object Notation (JSON)

Standar komunikasi supaya efektif memerlukan format data. Format data harus dimengerti oleh server maupun pengguna. Format data yang digunakan adalah JSON. JSON merupakan format data de facto di Web[11]. Implementasi JSON untuk *request handling* REST adalah

1. counter/add

```
"content":
{
      "total_count": 2,
      "current_count": 0,
      "counter_name": "cs",
      "type": 0
}
```

2. counter/get

```
"content": {
      "total_count": 2,
```

```
    "current_count": 2,
    "counter_name": "cs",
    "type": 0
}
```


3. counter/list

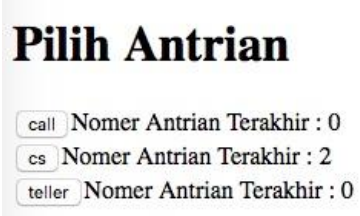

```
"content":
[
    {
      "total_count": 0,
      "current_count": 0,
      "counter_name": "call",
      "type": 3
    },
    {
      "total_count": 1,
      "current_count": 0,
      "counter_name": "cs",
      "type": 0
    },
    {
      "total_count": 0,
      "current_count": 0,
      "counter_name": "teller",
      "type": -1
    }
]
```


3 Hasil dan Analisis

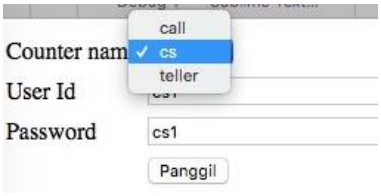
Penelitian sistem antrean merupakan aplikasi server yang berbasis REST. Pengujian dilakukan secara fungsionalitas dengan membuat aplikasi pengguna yang berbasis Hypertext Markup Language (HTML) dan Javascript. Aplikasi dijalankan dengan menggunakan web server. Dengan menggunakan aplikasi tersebut dilakukan tes uji coba dengan consumer sebagai berikut :

1. call untuk antrean pemrosesan data pendaftar
2. cs untuk antrean informasi tentang pendaftaran
3. teller untuk antrean pemrosesan keuangan

3.1. Hypertext Markup Language (HTML)



Gambar 11. Tampilan Producer/Pendaftar



Gambar 12. Tampilan Consumer/Akademik

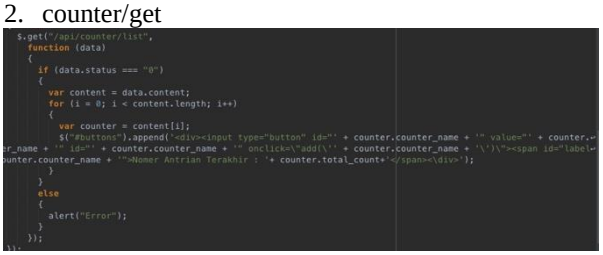
3.2. Javascript

1. counter/add

```
$.post("/api/counter/add",
  {
    counter_name: counter_name,
    user_id: "1",
    passkey: "1"
  },
  function (data)
  {
    if (data.status === "0")
    {
      var content = data.content;
      $("#label" + content.counter_name).html("Nomer Antrian Terakhir : " + content.total_count);
      if (content.type === 0)
      {
        alert("Antrian no : " + content.total_count + ", Menunggu Antrian : " + (content.total_count - conten
nt_count));
      }
      else
      {
        alert("Antrian no : " + content.total_count);
      }
    }
  }
);
```

Gambar 13. Fungsi untuk memanggil REST dengan fungsi add

2. counter/get



```
$.get("/api/counter/list",
  function (data)
  {
    if (data.status === "0")
    {
      var content = data.content;
      for (i = 0; i < content.length; i++)
      {
        var counter = content[i];
        $("#buttons").append('<div><input type="button" id="' + counter.counter_name + '" value="' + counter.
r_name + '" id="' + counter.counter_name + '" onclick=\"add(\'' + counter.counter_name + '\')\"><span id="label
ounter.counter_name + '"">Nomer Antrian Terakhir : '+ counter.total_count+'</span><\div>');
      }
    }
    else
    {
      alert("Error");
    }
  });
});
```

Gambar 14. Fungsi untuk memanggil REST dengan fungsi get

3. counter/list

```
$.get("/api/counter/list",
  function (data)
  {
    if (data.status === "0")
    {
      var content = data.content;
      for (i = 0; i < content.length; i++)
      {
        var counter = content[i];
        $("#buttons").append('<div><input type="button" id="' + counter.counter_name + '" value="' + counter.
r_name + '" id="' + counter.counter_name + '" onclick=\"add(\'' + counter.counter_name + '\')\"><span id="label
ounter.counter_name + '"">Nomer Antrian Terakhir : '+ counter.total_count+'</span><\div>');
      }
    }
    else
    {
      alert("Error");
    }
  });
});
```

Gambar 15. Fungsi untuk memanggil REST dengan fungsi list

3.3. Uji Coba

Uji coba dilakukan menggunakan Raspberry-Pi sebagai server sekaligus web server untuk aplikasi pengguna. Gambar 16-17 berikut merupakan hasil dari salah satu percobaan tersebut :

Gambar 16. Hasil Dari Add To Queue

Gambar 17. Hasil Dari Process Queue

Sistem sudah berhasil bekerja dan mengatasi masalah antrean. Keunggulan sistem ini dibanding penelitian [1][2] adalah

1. Dapat digunakan untuk kebutuhan berbeda dengan mengganti konfigurasi daftar Consumer tanpa perlu membuat ulang sistem.
2. Aplikasi client/pengguna dapat dibuat bebas asalkan merujuk pada spesifikasi REST dari sistem antrean. Contoh pengujian penelitian menggunakan web server, HTML, dan Javascript.
3. Penggunaan Raspberry Pi sebagai server sehingga biaya dapat ditekan.

4. Kesimpulan

Dengan hasil uji coba yang sudah dilakukan, dapat disimpulkan bahwa sistem yang dibuat sudah memenuhi tujuan dari penelitian yaitu :

1. Membangun sistem antrean yang dapat digunakan untuk kebutuhan yang berbeda.
2. Relasi *client-server* yang renggang (modular), sehingga perubahan di server tidak perlu mengubah client.
3. Menggunakan Raspberry Pi sebagai server, yang biayanya relatif lebih murah dari PC.

Sistem dapat digunakan sehingga manfaat dari penelitian juga terpenuhi. Manfaat tersebut adalah :

1. Calon mahasiswa baru dapat mengetahui urutan kapan data akan diproses.
2. Petugas akademik dapat lebih fokus memproses data mahasiswa baru.
3. Menekan biaya pembuatan sistem antrean

Untuk pengembangan lebih lanjut, dapat ditambahkan multimedia dan *thermal printer*. Penambahan multimedia dapat menarik minat dari pengguna mesin antrean. Selain menampilkan informasi antrean juga dapat memberikan informasi berguna lainnya, contohnya profil dan berita terkini dari pemilik sistem atau informasi produk yang dimiliki oleh pemilik sistem. *Thermal printer* digunakan untuk mencetak bukti antrean, sehingga pemilik sistem tidak perlu menyiapkan kartu antrean. Selain mencetak, *printer* dapat digunakan untuk menampilkan informasi berguna lainnya, seperti tanggal antrean, dan nomor telepon dari pemilik sistem.

Referensi

[1] M. Ridwan lubis, “Sistem Layanan Antrian Dengan SMS Pada Unit Pelayanan Mahasiswa (Studi Kasus : AMIK Tunas Bangsa Pematangsiantar),” *J. Penelit. Tek. Inform.*, vol. 1, pp. 42–44, 2016.

[2] S. B. Aziz, T. A. Riza, and R. Tulloh, “Pasien Pada Dokter Umum Berbasis Android Dan Sms Gateway Design and Implementation Queue System Application for Patient of General Practitioner Using Android and Sms,” *J. Elektro Telekomun. Terap.*, pp. 71–82, 2015.

[3] I. Sommerville, *Software Engineering: Global Edition, Tenth Edition*. 2016.

[4] A. Dennis, B. H. Wixom, and D. Tegarden, *Systems Analysis and Design : An Object-Oriented Approach with UML*, 5th ed. Wiley Publishing, Inc, 2015.

[5] S. Dhang, “Credit Card Size Computer,” *The Thecy*, 2017. [Online]. Available: http://www.thetechy.com/credit-card-size-computer/. [Accessed: 25-Feb-2018].

[6] Future Publishing, “The Ultimate Raspberry Pi Handbook,” pp. 1–1037, 2016.

[7] P. Deitel and H. Deitel, *Java How to Program, 10th Edition, Late Objects Version*. Pearson Education, Inc., 2015.

[8] T. H. D. Group, “HyperSQL,” *The HSQL Development Group.*, 2018. [Online]. Available: http://hsqldb.org. [Accessed: 25-Mar-2018].

[9] R. T. Fielding, “Architectural Styles and the Design of Network-based Software Architectures,” *Building*, vol. 54, p. 162, 2000.

[10] J. Louvel, T. Templier, and T. Boileau, *Restlet In Action*. 2012.

[11] B. Smith, *Beginning JSON*. Apress, 2015.