PEMBUATAN *WEB SERVICE* SISTEM CARI KERJA PERGURUAN TINGGI DENGAN METODE REST

Dudi Iskandar, Fitrah Satrya Fajar Kusumah, Hersanto Fajri
Teknik Informatika, Universitas Ibn Khaldun Bogor
Jl. Sholeh Iskandar Kedungbadak, Kota Bogor, Indonesia
dudyiskandar325@gmail.com

ABSTRAK

Perguruan Tinggi dituntut harus memberikan kontribusi yang nyata dalam meningkatkan kualitas dan daya saing mahasiswa di dunia kerja oleh pihak-pihak yang memiliki kepentingan (*stakeholders*), peningkatan kualitas dan daya saing yang dimana sudah diatur dalam Undang-undang No. 12 Tahun 2012 tentang Pendidikan Tinggi maka Pusat Karir merupakan satuan kerja yang mengemban tugas memangkas masa tunggu lulusan dalam memperoleh pekerjaan, pusat karir membuat aplikasi cari kerja berbasis *website* dan *mobile*, untuk memudahkan mahasiswa sebagai pencari kerja mendapatkan pekerjaan yang sesuai dengan keahliannya, sistem yang dibuat juga terintegrasi dengan sistem internal yang sudah berjalan seperti sitem gamifikasi. Tujuan utama penelitian ini adalah membuat *web service* cari kerja untuk menghubungkan aplikasi berbasis *website*, *mobile* dan sistem yang lainnya. Penelitian ini menggunakan metode *waterfall* sebagai metode perancangan aplikasi dan menggunakan bahasa pemrograman *typescript* dengan *framework Node js serta Express js.* Hasil penelitian ini adalah implementasi *web service* cari kerja sebagai mekanisme interaksi antar sistem yang mendukung agar dapat berkolaborasi. Dengan adanya *web service* ini diharapkan dapat menjadi jembatan penghubung antara aplikasi *website* dan *mobile.*

Kata kunci : *Cari Kerja, Web Service, REST API*

1. PENDAHULUAN

Perguruan Tinggi dituntut harus memberikan kontribusi yang nyata dalam meningkatkan kualitas dan daya saing mahasiswa di dunia kerja oleh pihak-pihak yang memiliki kepentingan (stakeholders), peningkatan kualitas dan daya saing yang dimana sudah diatur dalam Undang-undang No. 12 Tahun 2012 tentang Pendidikan Tinggi maka Pusat Karir merupakan satuan kerja yang mengemban tugas memangkas masa tunggu lulusan dalam memperoleh pekerjaan, menyelenggarakan survei alumni, mengembangkan jaringan informasi lowongan kerja dan menyelenggarakan Bursa Kerja Industri bidang seperti pendidikan, perbankan, komunikasi, dan bisnis sudah banyak di rasakan manfaatnya. Dengan perkembangan teknologi saat ini, hal ini tentunya akan berdampak positif bagi perusahaan yang membutuhkan karyawan, karena perusahaan dapat menyebarkan pekerjaan secara online dan menyaring calon karyawan tanpa kertas, karena calon karyawan mengunggah file yang dibutuhkan perusahaan secara online [1].

Manfaat lain dari teknologi dan informasi yaitu sebagai alat yang tujuannya adalah untuk mengembangkan dan memajukan organisasi [2].

Karena permasalahan di atas maka diperlukan teknologi web service sebagai mekanisme interaksi antar sistem, yang mendukung kolaborasi antara dua atau lebih program atau aplikasi dengan lingkungan sistem operasi, bahasa pemrograman, serta basis data berbeda yang memproses dan mengolah data tertentu [3], juga dapat digunakan oleh berbagai pihak melalui jaringan internet menggunakan perangkat berbeda yang dimiliki oleh setiap pengguna [4]

2. TINJAUAN PUSTAKA

Dalam bab kajian teoritis tentang teori-teori yang mendukung tema penelitian. Berikut adalah penjelasan tentang teori-teori yang mendukung penelitian.

2.1. Studi Literatur

Studi literatur merupakan penelitian terdahulu yang telah dilakukan sebelumnya dan dipublikasi. Berikut adalah studi literatur terkait dengan penelitian yang akan dibuat.

Pada penelitian yang dilakukan oleh Bagas Yusuf F., Hindriyanto Dwi P. Tahun 2022 yang berjudul "Perancangan Aplikasi Cari Kerja Berbasis *Website* Menggunakan Laravel". Penelitian ini membahas tentang pembuatan aplikasi cari kerja berbasis *website* [5].

Penelitian dengan judul "Sistem Informasi Lowongan Kerja Kota Tangerang Berbasis *Android* dan *Web Service"* yang dilakukan oleh Maesaroh S., Otto F., Muhammad N. pada tahun 2019. Penelitian ini membahas tentang aplikasi lowongan kerja berbasis *android* yang dibangun mampu menjembatani hubungan antara pemberi kerja dengan pencari kerja lebih efisien. Hal ini disebabkan adanya fasilitas informasi perusahaan dari aplikasi *android* [6].

Penelitian dengan judul "RESTful *Web Service* untuk Integrasi Sistem Akademik dan Perpustakaan Universitas Perjuangan" yang dilakukan oleh Randi Rizal dkk, pada tahun 2019. Penelitian ini membahas tentang Penerapan teknologi *web service* dengan menggunakan arsitektur REST pada sistem informasi akademik dan sistem informasi perpustakaan mampu mengintegrasikan kedua sistem tersebut. Sehingga proses *input* dan verifikasi data hanya dilakukan satu

kali, hal tersebut mengatasi terjadinya duplikasi data dan mengurangi perkerjaan *input* data. Format pertukaran data antar sistem menggunakan format JSON yang tidak bernegara (*stateless*) [7].

2.2. *Web Service*

Web service adalah mekanisme interaksi antar sistem yang mendukung interoperabilitas untuk tujuan integrasi data. Pada dasarnya *web service* memiliki karakteristik khusus berupa *uniform resource location* (URL) yang seragam seperti halnya halaman *web* pada umumnya. Namun, URL layanan *web* hanya berisi sekumpulan informasi, perintah, dan konfigurasi untuk tujuan pembuatan fungsi aplikasi tertentu. Selain itu, *web service* dapat diimplementasikan dalam berbagai bahasa pemrograman dan dapat diakses oleh berbagai *platform*, menjadikannya dapat digunakan sebagai mekanisme transaksi data di berbagai *platform* yang berbeda [8].

2.3. *Database*

Database atau basis data adalah kumpulan data yang dikelola sedemikian rupa atas dasar aturan-aturan tertentu yang saling terkait sehingga mudah untuk mengelolanya. Dengan kemudahannya pengguna bisa mendapatkan mencari, menyimpan, dan memproses informasi dengan mudah. Basis data dapat dipahami sebagai kabinet elektronik yang terorganisir dengan baik [9].

2.4. *JavaScript Object Notation* (JSON)

JavaScript Object Notation (JSON) adalah format pertukaran data yang ringan. Sangat mudah bagi manusia untuk membaca dan menulis. Mudah bagi mesin untuk mengurai dan menghasilkan. Ini didasarkan pada *subset* dari Standar Bahasa Pemrograman JavaScript ECMA-262 Edisi ke-3 - Desember 1999. JSON adalah format teks yang sepenuhnya bebas bahasa tetapi menggunakan konvensi yang akrab bagi pemrogram dari keluarga bahasa C, termasuk C, C++, C#, Java, JavaScript, Perl, Python, dan banyak lainnya. Properti ini menjadikan JSON sebagai bahasa pertukaran data yang ideal [10].

2.5. *Unified Modeling Language* (UML)

Unified modelling language (UML) merupakan salah satu media untuk menggambarkan perancangan sistem berorientasi objek atau dapat dikenal sebagai *blueprint* sebuah software. UML diharapkan mampu untuk mempermudah pengembangan sistem serta dapat digunakan sebagai jembatan penerjemah antara pengembang sistem dengan pengguna [11].

2.6. *Postman*

Postman adalah sebuah aplikasi yang berfungsi sebagai *REST Client* untuk uji coba REST *API*. Postman menyederhanakan setiap langkah siklus hidup *API* dan merampingkan kolaborasi sehingga Anda dapat membuat *API* yang lebih baik—lebih cepat [12].

2.7. *Representational State Transfer* (REST)

REST (*Representational State Transfer*) Merupakan *standard* dalam arsitektur *web* yang menggunakan *Protocol* HTTP untuk pertukaran data. Konsep REST pertama kali diperkenalkan oleh Roy Fielding pada tahun 2000. REST *server* menyediakan jalur untuk akses *resource* atau data, sedangkan REST *client* melakukan akses *resource* dan kemudian menampilkan atau menggunakannya. *Resource* yang dihasilkan sebenarnya berupa teks, namun formatnya bisa bermacam – macam tergantung keinginan *developer*, umumnya adalah JSON dan XML. Metode REST yang umum digunakan untuk melakukan operasi *create, read, update, delete* dengan menggunakan metode (*GET, PUT, DELETE, POST*) [13].

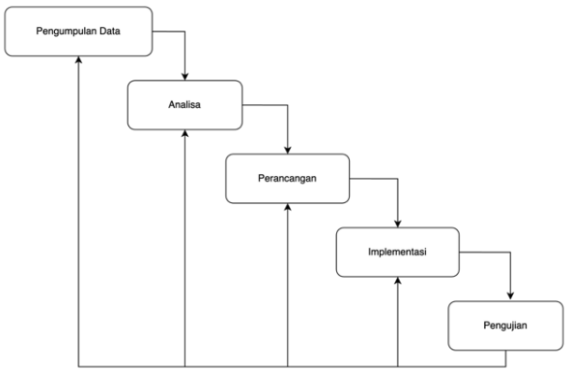
2.8. *Waterfall*

Metode *Waterfall* merupakan salah satu metode pengembangan sistem yang sering digunakan oleh para pengembang, karena mudah digunakan serta terstruktur sesuai dengan siklus pengembangan yang ada. Metode *waterfall* juga dikenal sebagai metode pengembangan yang mengadaptasi alur seperti air mengalir ke bawah (air terjun) melewati tahapan – tahapan seperti pengumpulan data, analisis, perancangan, implementasi, dan pengujian. Jika tahapan 1 belum selesai, maka tahapan 2 tidak dapat berjalan, begitu pun seterusnya [11].

2.9. Node JS

Sebagai *runtime JavaScript* berbasis peristiwa asinkron, Node Js dirancang untuk membangun aplikasi jaringan yang dapat diskalakan. Node Js bebas dari kekhawatiran mengunci proses, karena tidak ada kunci. Hampir tidak ada fungsi di Node Js yang secara langsung melakukan I/O, jadi prosesnya tidak pernah diblokir kecuali saat I/O dilakukan menggunakan metode sinkron dari pustaka standar Node Js. Node js mirip dalam desain, dan dipengaruhi oleh, sistem seperti *Ruby's Event Machine dan Python's Twisted* [14].

3. METODE PENELITIAN



Gambar 1. Metode Penelitian *Waterfall*

Metode penelitian ini menggunakan metode *Waterfall* menurut Roger S. Pressman E.7. Meliputi pengumpulan data, analisis, perancangan, pengkodean, pengujian. Metode penelitian ditunjukkan pada Gambar 1.

3.1. Pengumpulan Data

Pada tahap ini dilakukan proses pencarian serta pengumpulan data untuk mempermudah proses analisis terkait penelitian. Adapun data yang dibutuhkan dibagi menjadi dua bagian, yaitu:

a. Data Primer
   Data primer yaitu data mahasiswa Universitas Ibn Khaldun Bogor, dengan kriteria mahasiswa angkatan 2020 sampai dengan 2023 dan 50 mahasiswa dari setiap fakultas.
b. Data Sekunder
   Data sekunder yaitu data yang diperoleh dari sumber lain seperti jurnal, *e-book, website* dan dokumen pemerintah dengan cara studi literatur terkait dengan cari kerja, gamifikasi, dan *web service*.

3.2. Analisis (*Requirement Definition*)

Pada tahap ini dilakukan proses analisis dengan tujuan pembuatan yang dilakukan untuk mendefinisikan kebutuhan-kebutuhan sistem serta analisis pengolahan data yang telah diperoleh meliputi analisis kebutuhan fungsional sistem, kebutuhan non – fungsional, arsitektur sistem, sistem yang berjalan, sistem yang akan dibuat.

3.3. Perancangan (*System and Software Design*)

Pada tahap ini dilakukan proses perancangan desain sistem yang digambarkan melalui UML, perancangan *database* yang didapatkan pada tahap analisis, merancang pemasangan *web service* dengan metode REST. Perancangan yang akan disajikan yaitu, perancangan *database*, perancangan *use case* diagram, perancangan *component diagram*, dan perancangan *deployment diagram*.

3.4. Implementasi (*Implementation and Unit Testing*)

Tahap ini dilakukan proses implementasi ke dalam bentuk yang dapat dipahami oleh komputer menggunakan bahasa pemrograman berbasis *Javascript* dengan *framework Node JS* serta *Express JS*. Setelah dilakukan proses implementasi, maka akan dilakukan proses pengujian sistem yang telah dibuat secara bertahap.

3.5. Pengujian (*Integration and System Testing*)

Pada tahap ini dilakukan proses pengujian sistem dengan tujuan untuk mengetahui kekurangan ataupun kesalahan yang terdapat dalam sistem yang telah dibuat. Pengujian sistem ini menggunakan metode *black box* untuk memeriksa fungsi dari setiap *web service* yang telah dibuat. Pengujian dilakukan menggunakan *software* Postman.

4. HASIL DAN PEMBAHASAN

4.1. Analisis Kebutuhan Sistem

Pada tahapan analisis akan dilakukan proses analisis kebutuhan sistem diantara-Nya analisis kebutuhan fungsional sistem, analisis kebutuhan non - fungsional sistem, dan analisis arsitektur sistem.

4.1.1. Analisis Kebutuhan Fungsional

Analisis kebutuhan fungsional merupakan analisis kebutuhan pada sistem berupa layanan *web service* yang disediakan untuk proses pencarian kerja serta pengelolaan sistem manajemen. Berdasarkan kebutuhan pencarian kerja maka modul – modul yang akan disediakan oleh *web service* dapat dilihat pada Table 1:

Tabel 1. Analisis Kebutuhan Fungsional

No	Modul Kebutuhan Fungsional	Dekripsi
1	Fungsi Login	Fungsi ini digunakan pengguna untuk masuk ke sistem sesuai dengan *role* nya, serta otorisasi yang dapat dilakukan di sistem, jika *role* sebagai kandidat dapat mengelola dan melakukan proses melamar ke pekerjaan yang tersedia, kemudian untuk *recruiter* dapat mengelola pekerjaan, dan melihat kandidat yang telah melamar pada pekerjaan.
2	Fungsi Register Kandidat	Fungsi ini digunakan kandidat yang belum memiliki akun untuk membuat akun baru di sistem cari kerja.
3	Fungsi Register *Recruiter*	Fungsi ini digunakan *recruiter* untuk mendaftarkan perusahaanya pada sistem, agar *recruiter* dapat menambahkan dan mempublikasi pekerjaan.
4	Fungsi Manajemen Profile	Fungsi ini digunakan untuk user mengubah data *profile*, seperti nama, email, nomor handphone dan foto *profile*.
5	Fungsi Manajemen Pekerjaan	Fungsi ini digunakan *recruiter* untuk menambahkan, mengubah, menghapus dan mempublikasi pekerjaan.
6	Fungsi *Apply* Pekerjaan	Fungsi ini digunakan oleh kandidat untuk melakukan *apply* pekerjaan sampai dengan proses pembuatan *file pdf cover-letter* yang dapat di unduh oleh *recruiter* ataupun kandidat
7	Fungsi Integrasi Gamifikasi	Fungsi ini berjalan di belakang layar untuk melakukan sinkronisasi data *level* gamifikasi mahasiswa dengan sistem TIAS (Teknik Informatika Akademik Sistem), pada modul ini hanya mendapatkan data total *score* saja tidak ada proses perhitungan gamifikasi.

4.1.2. Analisis Kebutuhan Non. - Fungsional

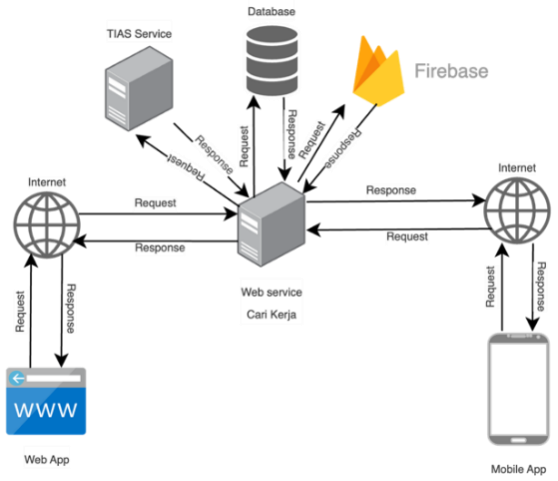
Analisis kebutuhan non – fungsional merupakan analisis yang digunakan untuk mendefinisikan hal – hal yang berkaitan dengan sistem yang sedang berjalan. Berikut adalah kebutuhan non – fungsional sistem dapat dilihat pada Table 2.

Table 2. Analisis Kebutuhan Non – Fungsional

No.	Kebuthan Non - Fungsional	Deskripsi
1.	*Correctness*	*Web service* memberikan data yang benar sesuai dengan data yang di *request* oleh *end user*
2.	*Integrity*	*Web service* memiliki kemampuan untuk dapat mengawasi akses dari setiap pengguna terhadap data..
3.	*Reability*	*Web service* dapat berjalan sesuai dengan waktu yang telah ditentukan.
4.	*Interoperability*	*Web service* dapat digunakan oleh *platform* dan *operating* sistem yang berbeda.
5.	*Portability*	*Web service* dapat diakses melalui jaringan internet.

4.1.3. Analisis Arsitektur Sistem

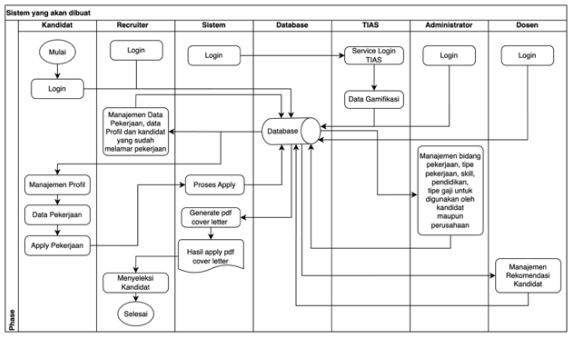
Analisis arsitektur sistem merupakan analisis yang digunakan untuk merancang arsitektur suatu *web service* sesuai dengan model *client-server*. *Web service* menerapkan komunikasi dua arah, dimana *client* dapat melakukan *request* ke *server* dengan parameter tertentu, kemudian *request* tersebut masuk ke *server* diolah dan disajikan sebagai *response*. *Client* dan *server* tidak terkait langsung tetapi menggunakan perantara *web service*. dalam penelitian, format data yang digunakan adalah JSON. Arsitektur sistem yang dibuat ditampilkan pada Gambar 2.



Gambar 2. Arsitektur Sistem

4.1.4. Analisis Sistem yang Akan Dibuat

Analisis sistem akan buat dengan menjelaskan bagaimana sistem atau alur proses pelaksanaan kegiatan yang akan dibuat. Sistem yang akan diterapkan pada penelitian ini adalah menyediakan *web service* untuk digunakan beberapa langkah seperti login, proses pengelolaan data seperti pengelolaan data pekerjaan, data kandidat yang sudah melamar pekerjaan, pengelolaan data manajemen profil kandidat dan recruiter, pengelolaan data master bidang pekerjaan, tipe pekerjaan, skill, pendidikan, tipe gaji untuk digunakan oleh kandidat maupun perusahaan dan integrasi data gamifikasi dengan sistem TIAS, serta sistem akan menyediakan file pdf cover letter sebagai bukti hasil apply. Gambar analisis sistem yang akan dilakukan dapat dilihat di Gambar 3.

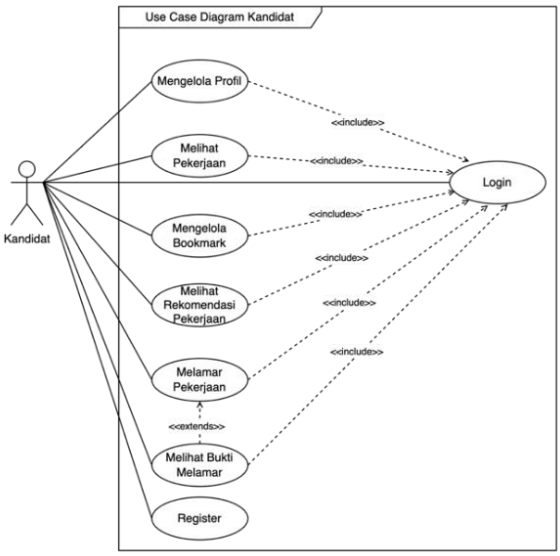


Gambar 3. Analisis Sistem yang akan dibuat

4.2. Perancangan

Perancangan pada penelitian ini terbagi menjadi 5 jenis yaitu perancangan *use case* diagram, perancangan *component* diagram, perancangan *deployment* diagram dan perancangan *database*.

4.2.1. *Use Case* Diagram



Gambar 4. Use Case Kandidat

Use case diagram digunakan untuk menggambarkan interaksi antara *actor* dengan sistem. Perancangan *use case* diagram dibagi menjadi tiga

bagian, yaitu *use case* diagram kandidat, *use case* diagram *recruiter*, dan *use case* diagram administrator. Use case diagram dapat dilihat pada Gambar 4.

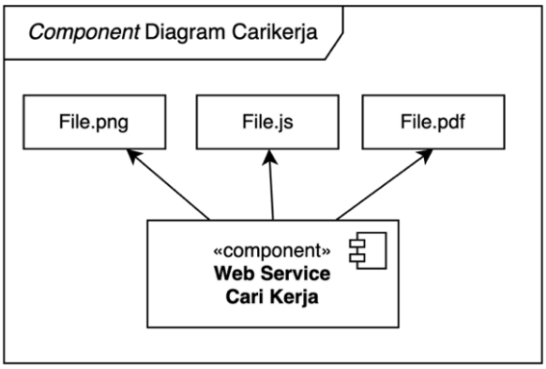
4.2.2. *Class* Diagram

Class diagram digunakan untuk menggambarkan himpunan kelas, antarmuka, kolaborasi dan relasi antara kelas serta antar muka. *Class* diagram dapat dilihat pada Gambar 5.

Gambar 5. *Class* Diagram

4.2.3. *Component* Diagram

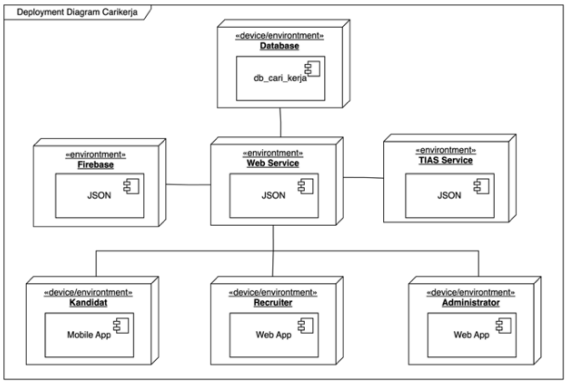
Component diagram menggambarkan struktur dan relasi antar komponen serta ketergantungan antara komponen lainnya dengan sistem. Component diagram dapat dilihat pada Gambar 6.



Gambar 6. *Component* Diagram

4.2.4. *Deployment* Diagram

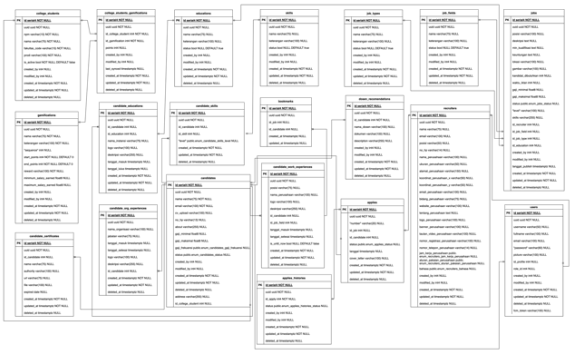
Deployment diagram menggambarkan hubungan antara komponen *software* dan komponen hardware serta menunjukkan konfigurasi setiap komponen pada saat dijalankan. *Deployment* diagram dapat dilihat pada Gambar 7.



Gambar 7. *Deployment* Diagram

4.2.5. *Peracncangan Database*

Database yang akan dibuat memiliki 20 tabel dan dapat dilihat pada Gambar 8.



Gambar 8. Perancangan Database

4.3. Hasil Implementasi

Berdasarkan analisis dan perancangan yang telah dilakukan, maka dibangun *web service* untuk transaksi data antar aplikasi cari kerja serta untuk mendukung *interoperabilitas* antar sistem. *Web service* yang telah dihasilkan akan diuji dengan menggunakan software Postman. Berikut *web service* yang dibangun dalam aplikasi cari kerja:

4.3.1. Melamar Pekerjaan

Melamar pekerjaan merupakan *service* yang berfungsi untuk kandidat melamar pekerjaan yang tersedia, *method* yang digunakan untuk *service* ini yaitu *POST*. Berikut *source code* dari *service* melamar pekerjaan.

```
@Post('/candidate')
@Version('1')
@AppSecure()
@Auth([RoleTypesId.CANDIDATE])
@Validation(AppliesCreateDto)
async create({ body, user }):
Promise<ResponseSuccessArrayDto<AppliesDto>> {
  try {
    body.id_candidate = user.id_profile;
    const job = await this.jobSvc.getByUUID(body.uuid_job, {
      shouldNotFoundException: false,
      isPlain: true,
    });
    delete body.uuid_job;
    if (job) {
```

```
      body.id_job = job.id;
    } else {
      throw new BadRequestException('failed.job.not_found');
    }
    const isApplied = await
this.appliesvc.getByCandidateAndJob(
      user.id_profile,
      job.id,
      {
        shouldNotFoundException: false,
      },
    );
    let candidate;
    if (body.uuid_candidate && user.role_id ==
RoleTypesId.RECRUITER) {
      candidate = await
this.candidateSvc.getByUUID(body.uuid_candidate, {
        shouldNotFoundException: false,
        isPlain: true,
      });
      delete body.uuid_candidate;
      if (candidate) {
        body.id_candidate = candidate.id;
      } else {
        throw new BadRequestException('failed.job.not_found');
      }
    } else {
      candidate = await
this.candidateSvc.getById(user.id_profile);
    }
    const status = 'submitted';
    const created_at = today();
    const FE_HOST = this.apiConfigSvc.appConfig.fe_url;
    const number = `AP/${
      new Date().getUTCMilliseconds() + today('MMHHmmss')
    }/${today('YYYY')}`;
    let c_id = '';
    if (user.role_id == RoleTypesId.CANDIDATE) {
      c_id = user.id_profile;
    } else if (user.role_id == RoleTypesId.RECRUITER) {
      c_id = candidate?.id?.toString();
    }
```

```
    const coverLetter = {
      nama_rekruter: job.recruiter?.dataValues?.nama,
      judul_pekerjaan: job.posisi,
      lokasi_pekerjaan: job.lokasi,
      bidang_pekerjaan: job.job_field?.dataValues?.nama,
      nama_kandidat: candidate.nama,
      apply_number: number,
      candidate_id: c_id,
      apply_date: today('DD MMMM YYYY'),
      lokasi_kandidat: 'Bogor',
      id: c_id,
      qr: `${FE_HOST}/${c_id}_${job.id}`,
    };
    const coverLetterPDF = await
this.coverLetterSvc.generatePDF(coverLetter);

    if (!coverLetterPDF.status) {
      throw new Error(
        'Gagal melakukan proses pelamaran, silahkan coba lagi
nanti!',
      );
    }
    const data = {
      id_job: job.id,
      number,
      status,
      tanggal: created_at,
      id_candidate: c_id,
      cover_letter:
`${coverLetterPDF.data.fullpath}/${coverLetterPDF.data.filena
me}`,
    };
    body = { ...body, ...data };
    const applied = await this.appliesvc.create(body);
    return new ResponseSuccessObjectDto(applied);
  } catch (error) {
    console.log(error);
    return new ResponseErrorObjectDto(error);
  }
}
```

Adapun dokumentasi *service* melamar pekerjaan dapat dilihat pada Tabel 6.

Tabel 6. *Service* Melamar Pekerjaan

Melamar Pekejran	
URL	http://localhost:3000/api/v1/applies/candidate
Method	*POST*
Parameter	uuid_job

Success Response	"statusCode": 200,
	"status": "success",
	"message": "Proses berhasil dilakukan!",
	"data": {
	"uuid": "235ff929-62a8-426b-9082-3bd30452bb98",
	"created_at": "2023-08-25T02:54:48.383Z",
	"updated_at": "2023-08-25T02:54:48.383Z",
	"number": "AP/20008095448/2023",
	"status": "submitted",
	"tanggal": "2023-08-25T02:54:48.000Z",
	"cover_letter": "public/candidate/1/cover-letter/AP_20008095448_2023.pdf"
	}

4.4. Pengujian Sistem

Pengujian dimaksudkan untuk mengamati hasil menjalankan melalui data uji dan untuk memvalidasi fungsionalitas perangkat lunak. Pengujian dalam penelitian ini menggunakan metode *blackbox*. Metode *blackbox* melakukan pengecekan validasi terhadap hasil yang diberikan oleh sistem pada saat sistem menerima perintah. Pengujian pencarian kerja ditunjukkan pada Tabel 7.

Tabel 7. Pengujian Sistem

No	URL	Hasil Pengujian	Ketarangan	Status
1	http://localhost:3000/api/v1/auth/login	"status": 200, "success": true, "message": "Login Berhasil"	*Login*	Berhasil

2	http://localhost:3000/api/v1/users?limit=10&page=1&sortBy=username:asc&criteria=username:admin	"statusCode": 200,
		"status": "success",
		"message": "Proses berhasil dilakukan!",
		"data": [
		{
		"uuid": "00e10982-93ee-4663-b6db-75e821b56ba8",
		"created_at": "2023-08-12T17:37:56.533Z",
		"updated_at": "2023-08-12T17:37:56.533Z",
		"username": "admin",
		"fullname": "Super Adminisrator",
		"email": "admin@carikerja.id",
		"picture": null
		}
		]
			Melihat Data *User*	Berhasil

3	http://localhost:3000/api/v1/candidates/registration	"statusCode": 200,
		"status": "success",
		"message": "Pendaftaran kandidat berhasil!",
		"data": {
		"uuid": "ab274324-e4f5-4704-a2cb-716cf9ccfb37",
		"nama": "Kandidat A",
		"address": null,
		"email": "kandidat_a@gmail.com",
		}
			Pendaftaran Kandidat	Berhasil

4	http://localhost:3000/api/v1/jobs	"statusCode": 200, "status": "success", "message": "Proses berhasil dilakukan!",	Melihat Data Pekerjaan	Berhasil

5	http://localhost:3000/api/v1/applies/candidate	"statusCode": 200,
		"status": "success",
		"message": "Proses berhasil dilakukan!",
		"data": {
		"uuid": "235ff929-62a8-426b-9082-3bd30452bb98",
		"created_at": "2023-08-25T02:54:48.383Z",
		"updated_at": "2023-08-25T02:54:48.383Z",
		"number": "AP/20008095448/2023",
		"status": "submitted",
		"tanggal": "2023-08-25T02:54:48.000Z",
		"cover_letter": "public/candidate/1/cover-letter/AP_20008095448_2023.pdf"
		}
			Melamar Pekerjaan	Berhasil

6	http://localhost:3000/api/v1/applies/cover-letter/download?number=AP/35808171336/2023	*File pdf*	Mengunduh *Cover Letter*	Berhasil

5. KESIMPULAN DAN SARAN

Berdasarkan hasil perancangan dan pengujian *web service* cari kerja dengan metode REST berhasil di implementasikan pada aplikasi cari kerja baik di *mobile* ataupun *website* menggunakan Node JS sebagai teknologi untuk pengembangannya. *Web service* membuat komunikasi dan transaksi data antar aplikasi cari kerja berbasis *mobile* dan aplikasi cari kerja berbasis *website* dapat berjalan dengan baik. Dengan *web service* berhasil melakukan komunikasi dengan *service* lain seperti TIAS dengan baik, untuk melakukan sinkronisasi data antara data dari sistem cari kerja dan sistem TIAS.

DAFTAR PUSTAKA

[1] A. Sugiarto, "Efektifitas Pemanfaatan Teknologi Informasi dan Komunikasi terhadap Tata Persuratan Elektronik (Paperless Office System) (Studi Kasus: Universitas Islam Negeri Raden Fatah Palembang) Pusat Teknologi Informasi dan Pangkalan Data, UIN Raden Fatah Palembang," *JUSIFO*, vol. 6, pp. 45–54, 2020, doi: https://doi.org/10.19109/jusifo.v6i1.5632.

[2] U. Seleksi *et al.*, "Rancang Bangun E-Recruitment," *Journal Speed-Sentra Penelitian Engineering dan Edukasi*, vol. 12, no. 2, pp. 30–35, 2020.

[3] M. M. Amin, "Interoperabilitas perangkat lunak menggunakan restful web service," *Register:*

Jurnal Ilmiah Teknologi Sistem Informasi, vol. 4, no. 1, pp. 14–22, Jan. 2018, doi: 10.26594/register.v4i1.1129.

[4] E. Sutanta and K. Mustofa, "KEBUTUHAN WEB SERVICE UNTUK SINKRONISASI DATA ANTAR SISTEM INFORMASI DALAM E-GOV DI PEMKAB BANTUL YOGYAKARTA," 2012, [Online]. Available: http://bantulkab.go.id/

[5] Faqih Yusuf Bagas and Purnomo Dwi Hindriyanto, "Perancangan Aplikasi Cari Kerja Berbasis Website Menggunakan Laravel," *Jurnal Teknik Informatika dan Sistem Informasi*, vol. 9, pp. 566–581, 2022.

[6] Maisaroh Siti, Fajarianto Otto, and Nasir Muhammad, "Sistem Informasi Lowongan Kerja Kota Tangerang Berbasis Android dan Web Service," *JURNAL SISFOTEK GLOBAL*, vol. 9, pp. 112–117, 2019.

[7] R. Rizal and A. Rahmatulloh, "RESTful Web Service untuk Integrasi Sistem Akademik dan Perpustakaan Universitas Perjuangan," *Jurnal Ilmiah Informatika*, vol. 7, 2019.

[8] R. C. Buwono, "Web Services Menggunakan Format JSON".

[9] A. Silberschatz, H. F. Korth, and S. Sudarshan, *Database system concepts*.

[10] json.org, "Pengenalan JSON." Accessed: Aug. 14, 2023. [Online]. Available: https://www.json.org/json-en.html

[11] Roger S. Pressman, *Software Engineering: A Practitioner's Approach*. McGraw-Hill, 2010. [Online]. Available: www.mhhe.com/pressman.

[12] Postman Inc, "Apa itu Postman?" Accessed: Aug. 14, 2023. [Online]. Available: https://www.postman.com/product/what-is-postman/

[13] "RESTful Web Services Tutorial | Tutorialspoint." Accessed: Aug. 14, 2023. [Online]. Available: https://www.tutorialspoint.com/restful/

[14] nodejs.org, "Abot Node Js." Accessed: Aug. 14, 2023. [Online]. Available: https://nodejs.org/en/about