JUST TI: (Jurnal Sains Terapan Teknologi Informasi) 14, 2 (Juli , 2022): 1 - 9
Website: e-journal.polnes.ac.id/index.php/justi
ISSN: 2579-4510(online) ISSN: 2085-6458(print)

IMPLEMENTASI WEB SERVICE MENGGUNAKAN REST API UNTUK INTEGRASI DATA TATA NASKAH DINAS PADA APLIKASI OFC UNIVERSITAS HALU OLEO

Natalis Ransi[1)], Muhammad Alfian Izzah[2)], Arman[3)], La Ode Saidi[4)], Andi Tenriawaru[5)], La Surimi[6)], Edi Cahyono[7)]
[1,2,4,5,6]Studi Ilmu Komputer, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Halu Oleo
[3,7]Studi Matematika, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Halu Oleo
Jl. H.E.A Mokodompit, Kambu, Kendari, Sulawesi Tenggara
natalis.ransi@uho.ac.id [1)], alfianizzah6@gmail.com[2], arman.mtmk@uho.ac.id[3)], lm.saidi@yahoo.co.id[4)]
atenriawaru36@gmail.com[5)], lasurimi@uho.ac.id[6], edi.cahyono@uho.ac.id[7]

Abstrak **-** Universitas Halu Oleo (UHO) melalui Lembaga Pengembangan Sistem Informasi (LPSI) saat ini sedang membangun sistem informasi yang disebut sebagai *One File Cabinet* (OFC). OFC terdiri dari beberapa sub sistem, salah satunya adalah manajemen data tata naskah dinas. Salah satu kendala yang dihadapi pengembang OFC adalah pada saat akan mengembangkan sub sistem baru, pengembang akan selalu membuat komponen dari awal. Sebagai contohnya adalah komponen basis data. Tentu ini akan berimplikasi pada proses pembuatan sub sistem yang kurang efisien dan juga dapat terjadi duplikasi data yang tidak perlu. *Web Service* adalah metode atau cara komunikasi antara dua aplikasi atau perangkat elektronik melalui *world wide web*. Terdapat beberapa arsitektur *web service* diantaranya *Representative State Transfer* (REST). REST adalah sebuah *web service* dengan penerapan konsep perpindahan *state* yang bernavigasi melalui *link* HTTP dalam melakukan aktivitas tertentu. Pada penelitian ini dilakukan pengembangan sub sistem tata naskah dinas pada aplikasi OFC dengan mengimplementasikan *web service* menggunakan REST API. Pengujian dilakukan dengan metode *Behaviour Driven Development* (BDD). REST API yang dihasilkan akan menjadi jembatan penghubung antara sistem yang ada serta sistem baru yang akan dikembangkan nantinya. Berdasarkan pengujian yang telah dilakukan diperoleh kesimpulan yaitu perilaku pengguna saat berinteraksi dengan sistem dan menguji fungsi yang telah dipilih, semua fungsi tersebut telah dijalankan minimal satu kali dan melewati seluruh statement sehingga berhasil dan memperoleh nilai coverage sebesar 100%.
Kata Kunci : *Web Service, REST API, One File Cabinet*

1. PENDAHULUAN

Website adalah salah satu layanan yang bisa digunakan untuk mencari informasi, layanan *website* sangat dibutuhkan untuk keperluan pengguna dalam pencarian informasi, serta layanan berupa *download*, *upload*, *email* dan lainnya. *Website* sangat membutuhkan internet, tanpa internet *website* tidak akan dapat digunakan secara *online* (Sulistiyanto, 2012).

Menurut (Bora dan Bezboruah, 2015) *Web service* adalah *software* yang dibangun untuk membangun komunikasi antara sistem atau perangkat yang terpisah melalui internet. *Web service* membuat kemampuan internet dengan menambahkan kemampuan *transactional web*, yaitu kemampuan *web* untuk saling berkomunikasi dengan pola *program* to *program* (Asiyah, dkk, 2020).

Menurut (Kurniawan, dkk, 2013) Ada dua jenis *web service* diantaranya REST (*Representative State Transfer*) dan SOAP (*Simple Object Acces Protocol*). SOAP merupakan protol yang digunakan untuk bertukar data antara aplikasi dalam bentuk format XML (*Extensible Markup Language*) sedangkan REST adalah sebuah *web service* dengan penerapan konsep perpindahan *state* yang ber*navigasi* melalui *link* HTTP (*Hypertext Transfer Protocol*) dalam melakukan aktivitas tertentu dengan format JSON (*JavaScript Object Notation*).

Konsep sistem berbasis *Application Programing Interface* (API) dengan gaya arsitektur REST merupakan teknologi yang memungkinkan komunikasi di antara berbagai *platform* lain (Redhat, 2022). Menurut (Kurniawan, dkk, 2013) REST adalah *web service* yang menerapkan konsep perpindahan *state* melalui *link* HTTP (*Hypertext Transfer Protocol*) untuk melakukan aktivitas tertentu. Rest yang menyediakan *resource* sebagai layanannya.

Webservice adalah Metode atau cara komunikasi antara dua aplikasi atau perangkat elektronik melalui *world wide web*. *Web service* terdiri dari dua jenis SOAP dan REST (Safitri, R.K., dan Putro, 2021). *Web service* menyimpan informasi dalam format JSON atau XML dan dapat dipanggil oleh aplikasi lain dengan bantuan HTTP (Zaman, 2017). Menurut (Arianto, dkk, 2016) *web services* memungkinkan

http://dx.doi.org/10.46964/justti.v14i2.1547
Received: 03 September 2021; Revised: 09 September 2021 ; Accepted: 22 Januari 2022

sebuah fungsi dalam *web services* dapat dipinjam oleh aplikasi lain tanpa perlu mengetahui pemrograman detail didalamnya.

Representational State Transfer Application Programing Interface yang biasa di sebut sebagai REST API dan dikenal sebagai RESTful API adalah antaramuka pemrograman yang sesuai dengan batasan gaya arsitektur REST dan memungkinkan interaksi dengan layanan *web* (Redhat, 2022). Layanan *web* RESTful dibuat bekerja paling baik di *Web*, dalam gaya arsitektur REST, data dan fungsionalitas dianggap sebagai sumber daya dan diakses menggunakan *Uniform Resource Identifiers* (URI), biasanya tautan di *Web* (Oracle and/or, 2022), Sedangkan Menurut (Kurniawan, dkk, 2013) REST adalah *web service* yang menerapkan konsep perpindahan *state* melalui *link* HTTP untuk melakukan aktivitas tertentu. REST yang menyediakan *resource* sebagai layanannya.

Universitas Halu Oleo melalui Lembaga Pengembangan Sistem Informasi saat ini sedang membangun sistem informasi yang disebut sebagai *One File Cabinet* (OFC). OFC terdiri dari beberapa sub sistem, salah satunya adalah manajemen data tata naskah dinas. Salah satu kendala yang dihadapi pengembang OFC adalah pada saat akan mengembangkan sub sistem baru, pengembang akan selalu membuat komponen dari awal. Sebagai contohnya adalah komponen basis data. Tentu ini akan berimplikasi pada proses pembuatan sub sistem yang kurang efisien dan juga dapat terjadi duplikasi data yang tidak perlu.

2. METODE

2.1 Lokasi Penelitian

Penelitian REST API untuk integrasi data tata naskah dinas dilakukan pada bulan Maret sampai Mei 2022 di Lembaga Pengembangan Sistem Informasi Universitas Halu Oleo. Dengan rincian kegiatan dapat dilihat pada Tabel 1.

Tabel 1 Rincian Kegiatan Penelitian

No	Uraian	Mei 2022				Juni 2022				July 2022			
		I	II	III	IV	I	II	III	IV	I	II	III	IV
1.	SPRINT 1: Analisis Permasalahan												
2.	SPRINT 2: Perancangan Aplikasi												
3.	SPRINT 3: Pembuatan Aplikasi												
4.	SPRINT 4: Testing dan Evaluasi												
5.	SPRINT 5: Simpulan dan Saran												

2.2 Teknik Pengumpulan Data.

Metode pengumpulan data digunakan untuk memperoleh data untuk keperluan penelitian adalah sebagai berikut

1. Observasi

Metode yang dilakukan dengan secara langsung di Universitas Halu Oleo untuk mendapatkan berbagai data yang terkait dengan keperluan tata naskah dinas.

2. Wawancara

Kami melakukan wawancara langsung kepada pengelola data tata naskah dinas di Lembaga Pengembangan Sistem Informasi Universitas Halu Oleo mengenai hal yang berkaitan dengan penelitian.

3. Metode pustaka

Penulis memperoleh data dari bebergai sumber-sumber seperti buku-buku, jurnal dan referensi yang berkaitan dengan penelitian.

2.3. Metode Pengembangan Sistem

Adapun metode pengembangan sistem yang kami gunakan adalah *Scrum agile.* Metode *Scrum Agile* terdiri dari 4 tahapan yang dapat dilihat pada Table 2.

Tabel 2 *Tahapan metode Scrum Agile*

Fase Scrum	Pengertian
Sprint Planning.	*Sprint planning* atau perencanaan *sprint* adalah tahapan untuk menentukan item mana yang akan dikerjakan selama sprint berjalan. rencana yang ditetapkan merupakan hasil diskusi atau Kerjasama dari seluruh anggota scrum team
Daily Scrum.	*Daily scrum* adalah tahapan untuk evaluasi, tahapan ini berguna untuk memeriksa progress yang terjadi pada *sprint goal*, menyesuaikan rencana rencana kerja yang akan datang dan bentuk evaluasi lainnya. Tahapan ini dilakukan setiap hari selama sprint berjalan..
Sprint Review	*Sprint review* adalah tahapan untuk

	memeriksa Increment dan mengadaptasi *Produk Backlog* jika diperlukan. Pada tahapan ini semua tim yang terlibat berkolaborasi tentang apa yang dilakukan di *sprint*.
Sprint Retrospective	*Sprint Retrospective* merupakan tahapan yang digunakan *scrum team* untuk memeriksa dirinya sendiri dan membuat perencanaan mengenai peningkatan kualitas dan efektivitas yang dilakukan dalam sprint berikutnya.

2.4. Rancangan Design API.

Rancangan Design API yang ditentukan akan menjadi basis dari penerapan API dengan metode REST berdasarkan hasil dari analisis sistem. Desain ini bertujuan untuk memfokuskan pengembangan sistem supaya tidak keluar dari lingkup pengembangan API yang telah direncanakan. Untuk mempermudah desain API, penulis membagi modul berdasarkan model sistem. Diantaranya:

1) Modul User.

Pada modul user telah ditentukan API berdasarkan hal-hal yang berhubungan dengan user. Hal ini mencakup seluruh proses autentifikasi user. Modul user dapat dilihat pada Tabel 3.

Tabel 3 Modul *user*

No.	PROSES	API	
		METHOD	ENDPOINT
1.	Register	POST	/api/register
2.	Login	POST	/api/login
3.	*Logout*	POST	/api/*logout*

2) Modul Mahasiswa.

Pada modul mahasiswa telah ditentukan API berdasarkan hal hal yang berhubungan dengan mahasiswa. Hal ini mencakupi profile dan data mahasiswa. Modul Mahasiswa dapat dilihat pada Tabel 4.

Tabel 4 Modul mahasiswa

No.	PROSES	API	
		METHOD	ENDPOINT
1.	Menyimpan *profile* mahasiswa	POST	/api/mahasiswa/update-*profile*
2.	Mengupdate *profile* mahasiswa	PUT	/api/mahasiswa/update-*profile*/{id}
3.	Membuat data mahasiswa	POST	/api/mahasiswa
4.	Mengubah data mahasiswa	PUT	/api/mahasiswa/{id}
5.	Menghapus data mahasiswa	DELETE	/api/mahasiswa/{id}
6.	Melihat data mahasiswa	GET	/api/mahasiswa
7.	Melihat detail mahasiswa	GET	/api/mahasiswa/{id}

3) Modul Dosen.

Modul dosen, pada modul dosen telah ditentukan API berdasarkan hal hal yang berhubungan dengan mahasiswa. Hal ini mencakupi profile, data dosen. Modul Dosen dapat dilihat pada Tabel 5.

Tabel 5 Modul dosen

NO	PROSES	API	
		METHOD	ENDPOINT
1.	Simpan *profile* dosen	POST	/api/dosen/update-*profile*
2.	mengubah *profile* dosen	PUT	/api/dosen/update-*profile*/{id}
3.	menambahkan data dosen	POST	/api/dosen
4.	mengubah data dosen	PUT	/api/dosen/{id}
5.	menghapus data dosen	DELETE	/api/dosen/{id}
6.	Lihat data dosen	GET	/api/dosen
7.	Lihat detail dosen	GET	/api/dosen/{id}

4) Modul Admin.

Pada modul admin telah ditentukan API berdasarkan hal hal yang berhubungan dengan admin. Modul Admin dapat dilihat pada Tabel 6.

Tabel 6 Modul Admin

NO	PROSES	API	
		METHOD	ENDPOINT

1.	Get Detail data user	GET	/api/getuser

5) Modul Model Surat.

Pada modul model surat telah ditentukan API berdasarkan hal hal yang berhubungan dengan Model pembuatan surat. Hal ini mencakupi data model surat. Modul surat dapat dilihat pada Tabel 7

Tabel 7 Modul model surat

NO	PROSES	API	
		METHOD	ENDPOINT
1.	Melihat model surat	GET	/api/model_surat
2.	Create model surat	POST	/api/model_surat
3.	Update model surat	PUT	/api/model_surat /{kd}
4.	Delete model surat	DELETE	/api/model_surat /{kd}

6) Modul Persyaratan Surat.

Pada modul persyaratan surat telah ditentukan API berdasarkan hal hal yang berhubungan dengan persyaratan surat. Modul persyaratan surat dapat dilihat pada Tabel 8.

Tabel 8 Modul persyaratan surat

NO.	PROSES	API	
		METHOD	ENDPOINT
1.	Melihat Persyaratan surat	GET	/api/require_surat
2.	Create Persyaratan surat	POST	/api/require_surat
3.	Update Persyaratan surat	PUT	/api/require_surat/{kd}
4.	Delete Persyaratan surat	DELETE	/api/require_surat/{kd}

7) Modul Pembuatan Surat.

Pada modul pembuatan surat telah ditentukan API berdasarkan hal hal yang berhubungan dengan pembuatan surat mahasiswa. Modul pembuatan surat dapat dilihat pada Tabel 9.

Tabel 9 Modul pembuatan surat

No.	PROSES	API	
		METHOD	ENDPOINT
1.	Melihat surat yang pernah dibuat	GET	/api/surat
2.	Create surat	POST	/api/surat
3.	Delete surat	DELETE	/api/surat/{kd}

8) Modul Isian Surat.

Pada modul isian surat telah ditentukan API berdasarkan hal hal yang berhubungan dengan proses pengisian surat yang telah dibuat. Hal ini mencakupi pengelolaan surat milik user. Modul isian surat dapat dilihat pada Tabel 10.

Tabel 10 Modul isian surat

No.	PROSES	API	
		METHOD	ENDPOINT
1.	Melihat isian surat	GET	/api/isi_require_surat
2.	Create isian surat	POST	/api/isi_require_surat
3.	Update isian surat	PUT	/api/isi_require_surat/{kd}
4.	Delete isian surat	DELETE	/api/isi_require_surat/{kd}

9) Modul Izin Surat.

Pada modul izin surat telah ditentukan API berdasarkan hal hal yang berhubungan dengan proses pengizinan surat, apakah surat disetujui atau tidak. Hal ini mencakupi pengelolaan izin surat oleh dosen. Modul izin surat dapat dilihat pada Tabel 11.

Tabel 11 Modul izin surat

No.	PROSES	API	
		METHOD	ENDPOINT
1.	Melihat Izin surat	GET	/api/izin-surat
2.	Mengisi izin surat	PUT	/api/izin-surat/{kd}

10) Modul Prodi.

pada modul Prodi telah ditentukan API berdasarkan hal hal yang berhubungan dengan data prodi. Hal ini mencakupi pengelolaan data prodi. Modul Prodi dapat dilihat pada Tabel 12.

Tabel 12 Modul prodi

NO	PROSES	API	
		METHOD	ENDPOINT
1.	Melihat prodi	GET	/api/prodi

2.	Create prodi	POST	/api/prodi
3.	Update prodi	PUT	/api/prodi/{kd}
4.	Delete prodi	DELETE	/api/prodi/{kd}

11) Modul Absen.

Pada modul absen telah ditentukan API berdasarkan hal hal yang berhubungan dengan data absen. Hal ini mencakupi pengelolaan data absen serta pengabsenan oleh user. Modul Absen dapat dilihat pada Tabel 13.

Tabel 13 Modul Absen

No.	PROSES	API	
		METHOD	**ENDPOINT**
1.	Melihat absen	GET	/api/absen
2.	Create absen	POST	/api/absen
3.	Update absen	PUT	/api/absen/{kd}
4.	Delete absen	DELETE	/api/absen/{kd}
5.	Melihat daftar hadir absen	GET	/api/absen/{kd}/list-present
6.	Mengisi daftar hadir	POST	/api/present/absen/{absen}

12) Modul Lainnya.

Telah ditentukan API berdasarakan hal-hal yang yang berhubungan dengan hal tersebut. Modul Lainnya dapat dilihat pada Tabel 13.

Tabel 14 Modul lainnya

No.	PROSES	API	
		METHOD	**ENDPOINT**
1.	Refresh PDF	GET	/api/refresh-pdf/{PembuatanSurat}

3. HASIL DAN PEMBAHASAN

3.1 Hasil.

1) Use Case Diagram

Use case ini digunakan untuk menjelaskan bagaimana tahapan atau langkah-langkah yang dikerjakan oleh sistem kepada orang yang berada diluar sistem.

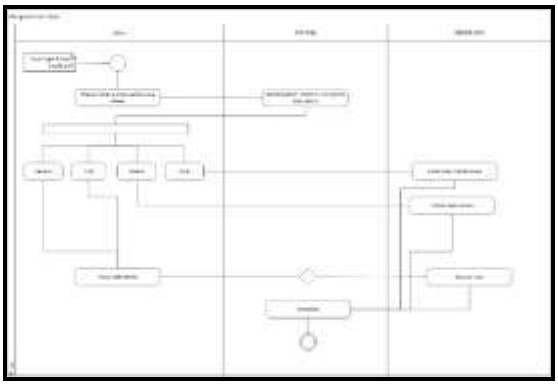
Gambar 1 *Use case diagram*

Use Case Diagram di atas menggambarkan aktivitas apa saja yang dilakukan oleh admin, mahasiswa dan dosen di dalam sistem. semua role aktor harus melakukan login untuk mendapatkan token agar bisa mengakses sistem. Mahasiswa dapat membuat surat, mengisi absen(daftar hadir) dan mengelola profile, dosen dapat membuat persetujuan surat, dan admin dapat mengakses semua action yang ada. Semua action ini dapat dilakukan setelah melakukan action login.

2) Activity Diagram

Activity diagram memberikan gambaran tentang aktifitas yang terjadi pada sistem. Dari pertama sampai akhir, diagram ini menunjukkan langkah-langkah dalam proses kerja sistem yang kita buat. Berikut merupakan *activity diagram* yang digunakan oleh peneliti.

- *Activity Diagram* Kelola Absen



Gambar 2 *Activity diagram* kelola absen

Activity diagram di atas menjelaskan Alur kerja *Actor* (*admin*) yang bertujuan untuk mengelola data absen. Proses ini dimulai dengan pengecekan apakah *user* sudah *login* dan *token* masih aktif yang kemudian akan menampilkan *list* daftar absen yang dapat dipilih dan di Kelola.

3) *Sequence Diagram*

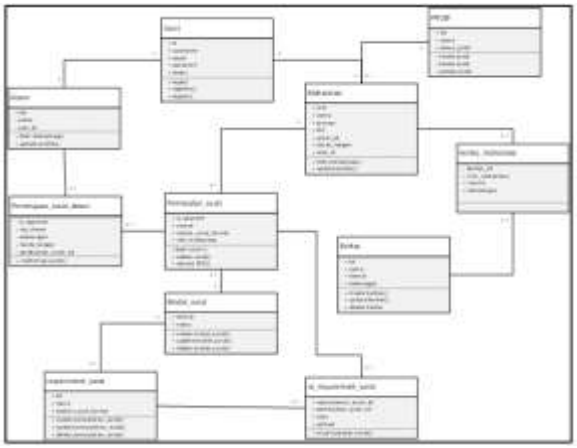
Sequence diagram ini digunakan untuk menggambarkan interaksi antara objek di dalam dan disekitar sistem, termasuk pengguna, display dan sebagiannya berupa *massage* yang digambarakan terhadap waktu.

Gambar 3 *Sequence diagram* mengisi absen.

Sequence diagram di atas mengambarkan interaksi antara *actor, tampilan absen,* API, *controller,* dan *model* antara *actor.* Mengisi absen dimulai dengan memasukan data pada form absen kemudian akan diproses API kemudian di kirim ke *controller,* selanjutnya *controller* akan mengirimkan data ke *model* absen_user. Jika data yang dimasukan sesuai dengan maka akan ditampilkan pesan berhasil, namun jika data yang dimasukkan sesuai dengan model absen_user, maka akan ditampilkan pesan berhasil dan absen berhasil di isi

4) *Class Diagram*

Class diagram dapat dilihat pada gambar 1. Desain ERD menggambarkan sturktur dari kelas dalam sistem dan menggambarkan *atribut, method* dan hubungan antara kelas. yang digunakan sebagai acuan dalam pembangunan aplikasi.



Gambar 4 *Class diagram*

3.2 Pembahasan

1) Implementasi Basis Data

Class diagram menggambarkan sturktur dari kelas dalam sistem dan menggambarkan *atribut, method* dan hubungan antara kelas. yang digunakan sebagai acuan dalam pembangunan sistem. Dalam penelitian ini terdapat beberapa atribut yang saling berhubungan digunakan sebagai acuan dalam pembuatan Sistem Informasi Tata Naskah Dinas, seperti *User*s, Dosen, Mahasiswa, Prodi, Persetujuan_surat_dosen, Pembuatan_Surat, Berkas, Berkas_Mahasiswa, Model_surat Requirment_surat, dan Isi_Requirment_surat.Class diagram dapat dilihat pada Gambar 5.

Gambar 5 Basis data

2) Implementasi REST API

- Implementasi API Login

API *login* dibutuhkan sebagai proses otentikasi akun sistem. REST API mewajibkan *developer*/*user* untuk *login* terlebih dahulu sebelum melakukan pengaksesan beberapa endpoint API. Proses *login* memerlukan data masukan berupa *email* dan *password*. Data masukan tersebut akan diproses oleh sistem (API) dan mengembalikan *token*. Koding API Login dapat dilihat pada Gambar 6.

```
if (!$token = auth()->attempt(request()->only(['email', 'password']))) {

        return response()->json([
            "success" => 401,
            "message" => "Invalid email or password",
            'error' => 'Unauthorized'
        ], 401);
    }

    $user = User::with(['mahasiswa', "dosen"])->where('email', request('email'))->firstOrFail();
    $user->tokens()->delete();

    $token = $user->createToken('auth_token')->plainTextToken;

    return response()->json([
        "status" => 200,
        "message" => "Login Success",
        'access_token' => $token,
        "data" => (new UserResource($user))
    ], 200);
}
```

Gambar 6 Koding login

Hasil *Response* yang dikirimkan ke API *Login* dan Ketika *user* berhasil di *Authentication* maka API akan mengirimkan *token* dan data *user* kepada *client.* Response API login dapat dilihat pada Gambar 7.

Gambar 7 Response API LOGIN

3) Implementasi Client REST API

Gambar 8 Halaman dashboard client REST API

Halaman di atas pertama kali muncul ketika user berhasil mengakses sistem informasi. Pada halaman ini terdapat beberapa tampilan menu profile, prodi, dosen , mahasiswa, surat dan halaman lainnya, yang dimana halaman-halaman tersebut data dan fungsinya terhubung langsung pada REST API yang dibuat.

4) Implementasi Pengujian Behavior Driven Development

Tabel 15 Hasil Pengujian BDD

No.	Scenario	Hasil yang diharapkan	Hasil
Feature: Login testing			
1.	*Error on empty fields*	Mengembalikan status 400 dan mengirimkan pesan *error*	sesuai
2.	*Wrong password*	Mengembalikan status 401 dan mengirimkan pesan *error*	sesuai
3.	*Login with valid token*	Mengembalikan status 200 dan mengirimkan token	susuai
Feature: Logout testing			
4.	*Logout with valid token*	Mengembalikan status 200 dan mengirimkan pesan berhasil	Sesuai

		logout	
5.	*Logout with invalid token*	Mengembalikan status 401 dan mengirimkan pesan error	Sesuai
6.	*Logout with empty token*	Mengembalikan status 400 dan mengirimkan pesan error	sesuai
Feature: *Register testing*			
7.	*Register Error on empty fields*	Mengembalikan status 400 dan mengirimkan pesan *error*	sesuai
8.	*Register Error on invalid email*	Mengembalikan status 400 dan mengirimkan pesan *error*	Sesuai
9.	*Register Error with email already exist*	Mengembalikan status 400 dan mengirimkan pesan *error*	Sesuai
10.	*Register with valid data*	Mengembalikan status 201 dan mengirimkan pesan berhasil Register	Sesuai
Feature: Mengelola data dosen			
11.	*Create* data dosen *with empty fields*	Mengembalikan status 422 dan mengirimkan pesan error	Sesuai
12.	*Create* data dosen *with valid field*	Mengembalikan status 201 dan mengirimkan pesan berhasil.	
13.	*Update* data dosen *with empty fields*	Mengembalikan status 422 dan mengirimkan pesan error	Sesuai
14.	*Update* data dosen *with valid id*	Mengembalikan status 200 dan mengirimkan pesan berhasil.	Sesuai
15.	*Delete data dosen with valid id*	Mengembalikan status 200 dan mengirimkan pesan berhasil.	Sesuai
16.	*Get* data Dosen	Mengembalikan status 200 dan mengirimkan pesan berhasil.	Sesuai
Feature: Mengisi Absen			
17.	*Failed filled in attendance, because of invalid data*	Mengembalikan status 500 dan mengirimkan pesan error	Sesuai
18.	*Successfully filled in attendance, because of valid data*	Mengembalikan status 200 dan mengirimkan pesan berhasil.	Sesuai
19.	*Failed to fill in attendance, because attendance already exists*	Mengembalikan status 409 dan mengirimkan pesan error	Sesuai
20.	*Failed to fill in attendance, because attendance already exists*	Mengembalikan status 403 dan mengirimkan pesan error	Sesuai

4 KESIMPULAN

Setelah melakukan analisis, perancangan dan melakukan implementasi *web service* dengan metode REST untuk integrasi tata naskah dinas Universitas Halu Oleo, dapat diperoleh kesimpulan diantaranya:

Penelitian ini menghasilkan sebuah *web service* dengan metode REST API yang dapat digunakan dalam integrasi data tata naskah dinas Universitas Halu Oleo dan berdasarkan hasil pengujian dengan menggunakan metode pengujian sistem *Behavior Driven Development* berdasarkan perilaku pengguna saat berinteraksi dengan sistem, semua fungsi berhasil berhasil melewati seluruh *statement* dan memperoleh nilai *coverage* sebesar 100%.

REFERENSI

[1] Affiliates, O. and/or its. (2022). *REST API for Oracle Service Cloud 18A - About the REST APIs*. https://docs.oracle.com/en/cloud/saas/service/18a/cxsvc/index.html

[2] Arianto, M.A., Munir, S., dan Khotimah, K. (2016). Analisis Dan Perancangan Representational State Transfer (Rest) Web Service Sistem Informasi Akademik Stt Terpadu Nurul Fikri Menggunakan Yii Framework. *Jurnal Teknologi Terpadu*, *2*(2), 1–4.

[3] Asiyah, L.N., Sulistyanto, M.P.T., dan Aziz, A. (2020). Penerapan Restful Web Service Untuk Optimalisasi Kecepatan Akses Pada Aplikasi Berbasis Android. *JOINTECS (Journal of Information Technology and Computer Science)*, *5*(2), 129. https://doi.org/10.31328/jointecs.v5i2.1260

[4] Bora, A., dan Bezboruah, T. (2015). A Comparative

Investigation on Implementation of RESTful versus SOAP based Web Services. *International Journal of Database Theory and Application*, *8*(3), 297–312. https://doi.org/10.14257/ijdta.2015.8.3.26

[5] Kurniawan, K.Y., Oslan, Y., dan Kristanto, H. (2013). Implementasi Rest - Api Untuk Portal Akademik Ukdw Berbasis Android. *Jurnal EKSIS*, *6*, 29–40.

[6] Redhat. (2022). *What is a REST API?* https://www.redhat.com/en/topics/api/what-is-a-rest-api

[7] Safitri, R.K., dan Putro, H. P. (2021). Implementasi REST API untuk Komunikasi Antara ReactJS dan NodeJS (Studi Kasus : Modul Manajemen User Solusi247). *Automata*, *2*(1), 0–4. https://journal.uii.ac.id/AUTOMATA/article/view/17381

[8] Sulistiyanto. (2012). Sulistiyanto Jurusan Teknik Informatika – STT Nurul Jadid Paiton. *APLIKASI SISTEM INFORMASI PAITON RESORT HOTEL ONLINE*, *6*.

[9] Zaman, G. A. P. (2017). Perancangan Dan Implementasi Web Service Sebagai Media Pertukaran Data Pada Aplikasi Permainan. *Jurnal Informatika*, *11*(2), 22–30. https://doi.org/10.26555/jifo.v11i2.a6252