OBJECT ORIENTED PROGRAMMING (EL 5121)

Dokumentasi Program

Sistem Disposisi Pekerjaan Berbasis Web Service

(Studi Kasus BPS Kabupaten Luwu Utara)

Diajukan Dalam Rangka Memenuhi Tugas Mata Kuliah Object Oriented Programming (El 5121)
Pada Magister Elektro Opsi Layanan Teknologi Informasi
Institut Teknologi Bandung

Disusun Oleh:

Aris Ardiansyah (232 17 143)

Magister Elektro Opsi Layanan Teknologi Informasi
Sekolah Tinggi Elektro dan Informatika
Institut Teknologi Bandung
2017

Pendahuluan

Sistem yang dibuat merupakan modifikasi sementara (belum final) terhadap sistem lama yang masih menggunakan arsitektur berorientasi objek. Seperti halnya sistem yang lama, sistem ini berfungsi sebagai sistem internal kantor (bukan untuk tujuan publik). Dengan menggunakan arsitektur berorientasi layanan, diharapkan pengembangan sistem ini ke depannya dapat lebih luas dari sisi wilayah akses sistem yang tidak dibatasi jaringan internal kantor dan kemungkinan penggunaan bahasa pemrograman yang berbeda dalam pengembangan aplikasi klien.

Sistem berfungsi sebagai pengelola disposisi pekerjaan internal di lingkup BPS Kabupaten Luwu Utara, dengan adanya sistem ini diharapkan penggunaan sumber daya manusia dalam menyelesaikan daftar tugas yang ada dapat berjalan lebih efisien dan merata. Kepala atau PPK pada awalnya mengisi daftar pekerjaan sesuai dengan output dan kegiatan yang dilimpahkan kepada BPS Kabupaten Luwu Utara, *Subject Matter* (Kepala Seksi) kemudian membagi mesing-masing kegiatan ini menjadi paket-paket pekerjaan baik berdasarkan wilayah maupun kriteria lainnya. Paket pekerjaan kemudian didistribusikan kepada para pegawai, pegawai yang memperoleh disposisi akan menerima notifikasi baik dalam bentuk pesan singkat *SMS* maupun *Email* dari sistem.

Server yang dibangun terbagi atas dua jenis, pertama yang menangani *request* dari klien dan yang kedua menangani pengiriman notifikasi ke pegawai. Untuk komunikasi diantara keduanya, sistem pertama akan membuat file notifikasi pada yang akan dibaca oleh aplikasi pengirim notifikasi dan mengirimkan isi dari file notifikasi tersebut dalam bentuk *SMS* atau *Email*.

Server pertama dibangun dengan menggunakan bahasa pemrograman Go-lang menggunakan *tools* Visual Studio Code 2017 dan pengirim notifikasi dibangun dengan menggunakan bahasa pemrograman Visual Basic.NET dengan *tools* Visual Studio Community 2017. Untuk DBMS menggunakan SQL Server 2012 dengan tools pembuatan database menggunakan SQL Server Management Studio dan untuk manajemen *service* database menggunakan SQL Server Configuration Manager.

Aplikasi Klien yang dibuat (sebagai sampel aplikasi yang akan mengkonsumsi layanan dari sistem pertama dan membantu dalam proses *debugging*) berfungsi sebagai aplikasi untuk melakukan *input* dan operasi lain dengan cara mengirim permintaan kepada sistem pertama dalam bentuk *HTTP Request* dan menerima respon dalam bentuk *HTTPResponse*. Aplikasi klien dibangun dengan menggunakan bahasa Visual Basic.NET.

Berikut adalah dokumentasi dari Sistem pertama:

Task

Berfungsi untuk menangani CRUD ke tabel task (tabel yang menyimpan daftar tugas dari Kepala atau PPK)

1. Get Task

Berfungsi untuk memperoleh melakukan read tabel task

Request

Method	URL	Fungsi
GET	/task	Memperoleh daftar seluruh task dari sistem, nilai kembalian JSON
GET	/task/{seksi}	Memperoleh daftar task menurut {seksi} dari sistem, nilai kembalian JSON

Response

Code	Hasil
200	Response berupa daftar objek yang terdiri atas daftar tugas dalam format JSON atau nilai null [{ "ID": <id>, "Kegiatan": <kegiatan>, "DeskKegiatan": <deskripsi kegiatan>, "Jumlah": <jumlah>, "Mulai": <tanggal mulai>, "Selesai": <tanggal selesai> "Seksi": <seksi>, "Deskripsi": <deskripsi> }] Contoh: [{ "ID": 1018, "Kegiatan": 8, "DeskKegiatan": "", "Jumlah": 100, "Mulai": "2017-10-01T00:00:00Z", "Selesai": "2017-10-31T00:00:00Z", "Seksi": 5, "Deskripsi": "Pendataan Lapangan Nilai Tukar Petani" }]
404	Page Not Found
500	Internal Server Error

2. Insert Task

Berfungsi melakukan write ke tabel task

Request

Method	URL	Fungsi
POST	/task?[parameter]	Melakukan insert ke database

Parameter

?kegiatan=<kegiatan>&jumlah=<jumlah>&mulai=<mulai>&selesai=<selesai>&seksi=<seksi>&deskripsi=<deskripsi>

Contoh:

http://localhost:8181/task?kegiatan=9&jumlah=100&mulai=2017-10-03&selesai=2017-11-03&seksi=5&deskripsi=Pendataan lapangan UMK/UMB

Response

Code	Hasil

200	Response berupa objek pesan dari database dalam format JSON, field pesan berisi id hasil insert
	{
	"Status": "Success",
	"Message": <id>
	}
	Contoh:
	{
	"Status": "Success",
	"Message": "1019"
	}

404	Page Not Found
405	Method not allowed
500	Internal Server Error

3. Update Task

Berfungsi melakukan update ke tabel task

Request

Method	URL	Fungsi
PUT	/task/{id}?[parameter]	Melakukan update ke database

Parameter

?kegiatan=<kegiatan>&jumlah=<jumlah>&mulai=<mulai>&selesai=<selesai>&seksi=<seksi>&deskripsi=<deskripsi>

Contoh:

http://localhost:8181/task/1022?kegiatan=9&jumlah=100&mulai=2017-10-03&selesai=2017-11-05&seksi=5&deskripsi=Pendataan lapangan UMK/UMB

Response

Code	Hasil

200	Response berupa objek pesan dari database dalam format JSON, field pesan berisi id hasil update
	{
	"Status": " Updated Successfully ",
	"Message": <jumlah baris>
	}
	Contoh:
	{
	"Status": "Updated Successfully",
	"Message": "1 row"
	}

404	Page Not Found/Data dengan id = <id> tidak ditemukan
405	Method not allowed
500	Internal Server Error

4. Delete Task

Berfungsi melakukan Delete ke tabel task

Request

Method	URL	Fungsi
DELETE	/task/{id}	Melakukan delete ke database

Contoh:

http://localhost:8181/task/1022

Response

Code	Hasil
200	Response berupa objek pesan dari database dalam format JSON, field pesan berisi id hasil insert {

Code	Hasil

	"Status": " Deleted Successfully ",
	"Message": Berhasil menghapus ID = <id>, dengan jumlah = <jumlah baris>
	}
	Contoh:
	{
	"Status": "Deleted Successfully",
	"Message": "Berhasil menghapus ID = 1022, dengan jumlah = 1"
	}

404	Page Not Found/Data dengan id = <id> tidak ditemukan
405	Method not allowed
500	Internal Server Error

TaskDesc

Berfungsi untuk menangani CRUD ke tabel taskdesc (tabel yang menyimpan daftar tugas yang telah dialokasikan oleh kepala seksi)

1. Get Task Desc

Berfungsi untuk memperoleh melakukan read tabel taskdesc

Request

Method	URL	Fungsi
GET	/taskdesc/{field}?nilai=<nilai>	Memperoleh daftar seluruh deskripsi tugas dari sistem berdasarkan field tertentu dan nilai tertentu, nilai kembalian JSON

Response

Code	Hasil

200	Response berupa daftar objek yang terdiri atas daftar deskripsi tugas dalam format JSON atau nilai null
	[
	{
	"ID": <id>,
	"TaskID": <taskid>,
	"Author": <author>,
	"Judul": <judul>,
	"Deskripsi": <deskripsi>,
	"Jumlah": <jumlah>,
	"IsDelegated": <isdelegated>
	}
	]
	Contoh:
	[
	{
	"ID": 12,
	"TaskID": 1008,
	"Author": 3,
	"Judul": "Desa Cenning Blok 003B",
	"Deskripsi": "Pencacahan Lapangan Susenas Cenning 003B",
	"Jumlah": 1,
	"IsDelegated": 1
	}
	]

404	Page Not Found
500	Internal Server Error

2. Insert TaskDesc

Berfungsi melakukan write ke tabel task desc

Request

Method	URL	Fungsi
POST	/taskdesc?[parameter]	Melakukan insert ke database

Parameter

?taskid=<taskid>&author=<author>&judul=<judul>&deskripsi=<deskripsi>&jumlah=<jumlah>

Contoh:

http://localhost:8181/taskdesc?taskid=1009&author=7&judul=Kelurahan Baliase 003B&deskripsi=Entri Data Susenas Semester I Kel Baliase 003B&jumlah=10

Response

Code	Hasil

200	Response berupa objek pesan dari database dalam format JSON, field pesan berisi id hasil insert
	{
	"Status": "Success",
	"Message": <id>
	}
	Contoh:
	{
	"Status": "Success",
	"Message": "13"
	}

404	Page Not Found
405	Method not allowed
500	Internal Server Error

3. Update TaskDesc

Berfungsi melakukan update ke tabel task desc

Request

Method	URL	Fungsi
PUT	/taskdesc/{id}?[parameter]	Melakukan update ke database

Parameter

?taskid=<taskid>&author=<author>&judul=<judul>&deskripsi=<deskripsi>&jumlah=<jumlah>

Contoh:

http://localhost:8181/taskdesc/13?taskid=1009&author=7&judul=Kelurahan Baliase 003B&deskripsi=Entri Data Susenas Semester I Kel Baliase 003B&jumlah=10

Response

Code	Hasil

200	Response berupa objek pesan dari database dalam format JSON, field pesan berisi id hasil insert
	{
	"Status": " Updated Successfully ",
	"Message": <jumlah baris>
	}
	Contoh:
	{
	"Status": "Updated Successfully",
	"Message": "1 row"
	}

404	Page Not Found/Data dengan id = <id> tidak ditemukan
405	Method not allowed
500	Internal Server Error

4. Delete TaskDesc

Berfungsi melakukan Delete ke tabel task desc

Request

Method	URL	Fungsi
DELETE	/taskdesc/{id}	Melakukan delete ke database

Contoh:

http://localhost:8181/taskdesc/13

Response

Code	Hasil
200	Response berupa objek pesan dari database dalam format JSON, field pesan berisi id hasil insert { "Status": " Deleted Successfully ",

Code	Hasil

	"Message": Berhasil menghapus ID = <id>, dengan jumlah = <jumlah baris>
	}
	Contoh:
	{
	"Status": "Deleted Successfully",
	"Message": "Berhasil menghapus ID = 13, dengan jumlah = 1"
	}

404	Page Not Found/Data dengan id = <id> tidak ditemukan
405	Method not allowed
500	Internal Server Error

TaskList

Berfungsi untuk menangani CRUD ke tabel tasklist (tabel yang menyimpan daftar tugas yang telah dialokasikan kepada pegawai oleh kepala seksi)

1. Get Task List

Berfungsi untuk memperoleh melakukan read tabel tasklist

Request

Method	URL	Fungsi
GET	/tasklist	Memperoleh daftar seluruh daftar tugas terdelegasi nilai kembalian JSON
GET	/tasklist/{field}?nilai=<nilai>	Memperoleh daftar seluruh deskripsi tugas dari sistem berdasarkan field tertentu dan nilai tertentu, nilai kembalian JSON

Response

Code	Hasil

200	Response berupa daftar objek yang terdiri atas daftar deskripsi tugas dalam format JSON atau nilai null
	[
	{
	"ID": <id>,
	"TaskID": <taskid>,
	"Pegawai": <nip pegawai>,
	"Parent": <seksi>,
	"Mulai": <tanggal mulai>,
	"Selesai": <tanggal selesai>,
	"Status": <status>
	}
	]Contoh:
	[
	{
	"ID": 6,
	"TaskID": 7,
	"Pegawai": "198704062009121002",
	"Parent": 7,
	"Mulai": "2017-04-03T00:00:00Z",
	"Selesai": "2017-12-06T00:00:00Z",
	"Status": 1
	}
	]

404	Page Not Found
500	Internal Server Error

2. Insert TaskList

Berfungsi melakukan write ke tabel tasklist

Request

Method	URL	Fungsi
POST	/tasklist?[parameter]	Melakukan insert ke database

Parameter

?taskid=<taskid>&pegawai=<pegawai>&parent=<parent>&mulai=<mulai>&selesai=<selesai>&status=<status>

Contoh:

http://localhost:8181/tasklist?taskid=14&pegawai=198704062009121002&parent=7&mulai=2017-03-31&selesai=2017-04-03>&status=1

Response

Code	Hasil

200	Response berupa objek pesan dari database dalam format JSON, field pesan berisi id hasil insert
	{
	"Status": "Success",
	"Message": <id>
	}
	Contoh:
	{
	"Status": "Success",
	"Message": "12"
	}

404	Page Not Found
405	Method not allowed
500	Internal Server Error

3. Update TaskList

Berfungsi melakukan update ke tabel task list

Request

Method	URL	Fungsi
PUT	/tasklist/{id}?[parameter]	Melakukan update ke database

Parameter

?taskid=<taskid>&pegawai=<pegawai>&parent=<parent>&mulai=<mulai>&selesai=<selesai>&status=<status>

Contoh:

http://localhost:8181/tasklist?taskid=14&pegawai=198704062009121002&parent=7&mulai=2017-03-31&selesai=2017-04-05>&status=1

Response

Code	Hasil

200	Response berupa objek pesan dari database dalam format JSON, field pesan berisi id hasil insert
	{
	"Status": " Updated Successfully ",
	"Message": <jumlah baris>
	}
	Contoh:
	{
	"Status": "Updated Successfully",
	"Message": "1 row"
	}

404	Page Not Found/Data dengan id = <id> tidak ditemukan
405	Method not allowed
500	Internal Server Error

4. Delete TaskList

Berfungsi melakukan Delete ke tabel task list

Request

Method	URL	Fungsi
DELETE	/tasklist/{id}	Melakukan delete ke database

Contoh:

http://localhost:8181/tasklist/14

Response

Code	Hasil
200	Response berupa objek pesan dari database dalam format JSON, field pesan berisi id hasil insert {

Code	Hasil

	"Status": " Deleted Successfully ",
	"Message": Berhasil menghapus ID = <id>, dengan jumlah = <jumlah baris>
	}
	Contoh:
	{
	"Status": "Deleted Successfully",
	"Message": "Berhasil menghapus ID = 13, dengan jumlah = 1"
	}

404	Page Not Found/Data dengan id = <id> tidak ditemukan
405	Method not allowed
500	Internal Server Error