Jurnal ICT : Information Communication & Technology
Vol. 21, N0.1, Juli 2022, pp. 61-67
p-ISSN: 2302-0261, e-ISSN: 2303-3363, DOI:

Sistem *Data Bridge* Pemantauan Gunungapi Berbasis *Application Protocol Interface* (Studi Kasus di Provinsi Jawa Barat)

Rudi Kurniawan [1], Hamdan Hanafi [2]
[1,2] Program Studi Teknik Informatika STMIK "AMIKBANDUNG"
Jln. Jakarta No. 28 Bandung 40272 INDONESIA
Email: [1]rudikurniawan@stmik-amikbandung.ac.id, [2]hamdanhanafi90@gmail.com

INFORMASI ARTIKEL

Histori artikel:

Naskah masuk, 1 Juni 2022
Direvisi, 28 Juni 2022
Diiterima, 29 Juni 2022

Kata Kunci:

Sistem Data Bridge
Gunungapi
Geospasial
Agile
API PVMBG
API BMKG

ABSTRAK

Abstract- The country of Indonesia has around 127 active volcanoes spread over 7 kilometers and the area is inhabited by around 4 million people, so the Center for Volcanology and Geological Hazard Mitigation (PVMBG) of the Geological Agency of the Ministry of Energy and Mineral Resources (ESDM) conducts 24-hour surveillance of the area. information about the condition of volcanoes by the community is an important problem that must be solved. The research covers the state of volcanoes in the province of West Java with the aim of creating a system that provides complete information about volcanoes. The development of the system used is the Agile methodology (Planning, Implementation, Software test, Documentation, Deployment, Software maintenance). Data bridge system is an API which can be consumed by frontend (website) to get complete information about an object. Data bridge API system with REST API architecture is used for communicating nikasi through the programming language Go as the main engine. In displaying information about volcanoes, data communication is carried out from two services (third party), namely the PVMBG API service and the BMKG API service. After being tested, the data bridge system shows that it can be a solution to solve the problem of the lack of information on volcanoes in Indonesia. Thus, early prevention of disasters in the form of volcanic eruptions or other things can be anticipated.

Abstrak- Negara Indonesia memiliki sekitar 127 gunungapi aktif yang tersebar sepanjang 7 kilometer dan diarea tersebut dihuni sekitar 4 juta jiwa sehingga pihak Pusat Vulkanologi dan Mitigasi Bencana Geologi (PVMBG) Badan Geologi Kementerian Energi dan Sumber Daya Mineral (ESDM melakukan pengawasan 24 jam terhadap daerah tersebut. Ketidaktahuan informasi mengenai kondisi gunungapi oleh masyarakat merupakan sebuah masalah penting yang harus dicari pemecahannya. Penelitian mencakup keadaan gunungapi yang ada di provinsi Jawa Barat dengan tujuan untuk membuat sebuah sistem yang menyediakan informasi lengkap mengenai gunungapi. Pengembangan sistem yang digunakan yaitu metodologi *Agile* (Perencanaan, Implementasi, Tes perangkat lunak, Dokumentasi, *Deployment*, Pemeliharaan perangkat lunak). Sistem *data bridge* adalah sebuah *API* yang dapat dikonsumsi oleh *frontend* (*website*) untuk mendapatkan informasi lengkap mengenai suatu objek. Sistem *data bridge API* dengan arsitektur *REST API* digunakan untuk berkomunikasi melalui bahasa pemrograman *Go* sebagai *engine* utama. Dalam menampilkan informasi mengenai gunungapi dilakukan komunikasi data dari dua *service* (*third party*), yaitu *service API* PVMBG dan *service API* BMKG. Setelah diujicoba, sistem *data bridge* menunjukan dapat menjadi solusi untuk memecahkan masalah mengenai kurangnya informasi gunungapi di Indonesia. Dengan demikian pencegahan dini pada bencana berupa letusan gunungapi atau hal-hal yang lain dapat diantisipasi.

***Penulis Korespondensi**:*

Rudi Kurniawan
Program Studi Teknik Informatika,
STMIK "AMIKBANDUNG"
Jl. Jakarta No. 28 Bandung, Jawa Barat, Indonesia
Email: rudikurniawan@stmik-amikbandung.ac.id

1. Pendahuluan

Berdasarkan data dari PVMBG (Pusat Vulkanologi dan Mitigasi Bencana Geologi) Indonesia memiliki sekitar 127 gunungapi aktif yang tersebar di jalur gunungapi sepanjang 7 kilometer, yang mana jalur gunungapi tersebut adalah daerah rawan bencana dan dihuni kurang lebih 4 juta jiwa. Oleh karena itu pihak PVMBG terus melakukan pengawasan 24 jam terhadap gunungapi yang aktif agar jika terjadi letusan maka bisa di antisipasi[10].

Masyarakat yang bertempat tinggal di sekitar jalur gunungapi berada pada daerah yang kurang aman, karena ancaman bencana gunungapi bisa terjadi secara tiba-tiba, dan masyarakat dalam keadaan tidak siap saat bencana terjadi atau tidak mengetahui informasi mengenai kondisi gunungapi[5].

Ketidaktahuan mengenai informasi terhadap kondisi gunungapi oleh masyarakat merupakan sebuah masalah penting yang harus dicarikan pemecahannya. Oleh karena itu sebagai alternatif solusi, diperlukan sebuah sistem informasi yang dapat memberikan informasi mengenai kondisi sebuah gunungapi. Dimana sistem tersebut dapat di akses oleh masyarakat umum secara mudah, informatif dan mudah dipahami. Sistem tersebut berupa aplikasi yang di akses oleh masyarakat melalui jaringan internet, sehingga masyarakat siap menghadapi sebuah bencana letusan jika suatu saat akan terjadi. Dampaknya melalui informasi yang sudah diketahui oleh masyarakat maka akan meminimalisir kepanikan, sehingga masyarakat bisa lebih mandiri dalam melakukan evakuasi dan akan cepat tanggap dalam menghadapi bencana letusan gunungapi tersebut[6].

Pada sistem yang dibangun untuk mengambil informasi dari aplikasi di Pusat Vulkanologi dan Mitigasi Bencana Geologi komunikasinya digunakan *library http client* bernama *resty*[13]. *Resty* adalah salah-satu *library* di dalam bahasa *Go* yang biasanya digunakan untuk keperluan berkomunikasi atau mengambil data dari *service* lain atau *third-party* berbasis API[8].

Aplikasi sistem informasi yang dibangun memiliki fungsi untuk memberikan informasi mengenai sejarah letusan, aktifitas vulkanik, data cuaca sekitar gunungapi, data pemantauan dan dampak dari letusan. Dengan demikian dengan adanya informasi tersebut maka langkah pencegahan korban bencana dari gunungapi bisa diantisipasi dan dihindari[6].

Penelitian sebelumnya yang berkaitan dengan topik yang dibahas yaitu: Penelitian yang dilakukan oleh [1] yang berjudul "Pemetaan Zona Bahaya Aliran Piroklastik Gunung Merapi, Jawa Tengah dan Sekitarnya menggunakan Aplikasi Titan2D". Pada penelitian tersebut dibahas mengenai simulasi aliran piroklastik (material vulkanik) sebagai salah satu upaya antisipasi terhadap erupsi suatu gunungapi dan efeknya pada zona atau daerah yang berpotensi terdampak. Hasil penelitian yaitu berupa simulai dari perangkat lunak Titan2D dengan input berupa data DEM (*Digitalized Elevation Model*) yang merepresentasikan bentuk morfologi Gunung Merapi yang memanfaatkan aplikasi Sistem Informasi Geografis (SIG)[1].

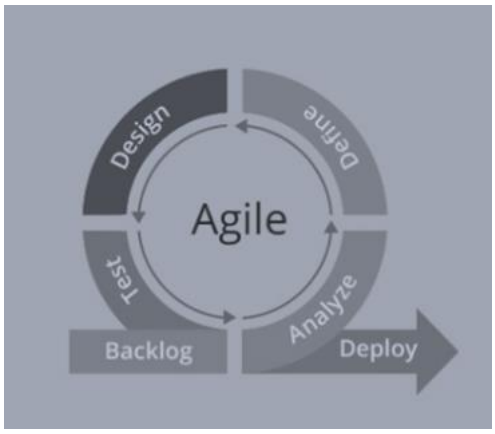
Penelitian yang dilakukan oleh [2] dengan judul "Rancang Bangun Rest API Aplikasi Weshare Sebagai Upaya Mempermudah Pelayanan Donasi Kemanusiaan" menyampaikan bahwa telah dikembangkan sistem transaksi donasi berbasis *Application Programming Interface* (API) sebagai *Backend Development* dan diimplementasikan untuk Mobile dan Website berbasis Android. Penelitian ini menghasilkan sistem berbasis API dengan arsitektur REST dalam hal *Backend Development* untuk memudahkan masyarakat dalam memberikan transaksi donasi dan diterapkan pada aplikasi Android dan website sebagai *user interface.*

Penelitian lainnya [5] yang berjudul "*Web Design and Application Programming Interface (API) Smart Farming Application*" membahasa permasalahan mengenai aktifitas di bidang pertanian yang dimulai dari proses pembibitan hingga proses pemberian pakan hingga proses pemanenan, yang dipengaruhi oleh pandemi Corona sehingga dengan membangun suatu sistem teknologi berupa *smart farming technology* yang dapat membantu proses pengelolaan pertanian serta mampu mengumpulkan data dan menyajikan data tanaman hidroponik selama proses penanaman sehingga dapat segera diambil tindakan yang tepat. Teknologi dibangun berupa Web dan API (*Application Programming Interface*)[8]. Dengan demikian melalui aplikasi *smart farming* sudah mampu membuat proses pengelolaan penanaman menjadi lebih efektif dan efisien.

2. Metode Penelitian

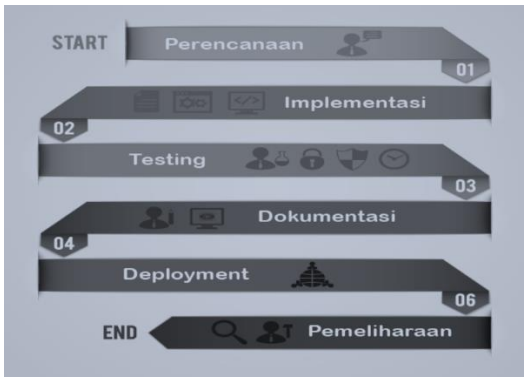
Pada penelitian ini digunakan metode pengembangan *software* yang dilakukan secara bertahap dan berulang (iterasi) yaitu *Agile*. Di mana

metode ini merupakan alternatif dari metode *Waterfall* yang linear dan tak bisa diubah di tengah proses pengembangan. Pada Gambar 1 dapat dilihat alur penelitian yang dipakai dalam penelitian ini.



Gambar 1. *Agile modelling*

Pengembangan perangkat lunak Agile memiliki beberapa tahapan dalam mengembangkan suatu proyek/perangkat lunak. Tahapan-tahapan tersebut diperlihatkan seperti yang terlihat pada Gambar 2.



Gambar 2. Tahapan metoda *Agile*

Tahapan-tahapan tersebut adalah sebagai berikut. a. Perencanaan, di mana Pengembang dan Klien membuat rencana tentang kebutuhan dari perangkat lunak yang akan dibuat; b. Implementasi, pada proses ini *programmer* melakukan pengkodean perangkat lunak; c. Tes perangkat lunak, di sini perangkat lunak yang telah dibuat di tes oleh bagian kontrol kualitas agar *bug* yang ditemukan bisa segera diperbaiki dan kualitas perangkat lunak terjaga; d. Dokumentasi, yaitu proses membuat dokumentasi untuk mempermudah proses *maintenanance* kedepannya; e. *Deployment*, yaitu proses yang dilakukan oleh penjamin kualitas untuk menguji kualitas sistem. Setelah sistem memenuhi syarat maka perangkat lunak siap di-*deployment*; Pemeliharaan, suatu perangkat lunak idak ada yang 100% bebas dari *bug*, oleh karena itu perangkat lunak dipelihara secara berkala untuk terus disempurnakan.

3. Hasil dan Pembahasan

3.1. Analisis dan *Define*

Penulis melakukan penelitian pada aplikasi MAGMA Indonesia yang dibuat oleh PVMBG (Pusat Vulkanologi dan Mitigasi Bencana Geologi). Aplikasi MAGMA Indonesia (*Multiplatform Application for Geohazard Mitigation and Assessment in Indonesia*), dikutip dari website official magma yaitu https://magma.vsi.esdm.go.id/, MAGMA Indonesia adalah sebuah aplikasi multiplatform (*web & mobile*) yang berisikan informasi dan rekomendasi kebencanaan geologi terintegrasi yang mencakup gunungapi, gempabumi, tsunami dan Gerakan tanah yang disajikan secara kuasi-realtime dan interaktif.

Aplikasi tersebut dibangun dan dikembangkan oleh PVMBG (Pusat Vulkanologi dan Mitigasi Bencana Geologi) menggunakan teknologi berbasis *open-source*. Adapaun informasi yang ditampilkan didalam aplikasi MAGMA Indonesia terdiri dari: 1) Gunungapi; 2) Gempabumi dan Tsunami; 3) Gerakan Tanah; 4) *Press Release* dan 5) Lapor Bencana. Adapun fitur-fitur yang terdapat pada aplikasi MAGMA Indonesia yaitu 1) Peta Interaktif, menampilkan informasi yang disajikan oleh MAGMA Indonesia; 2) *Press Release*, menampilkan berita laporan dari pemantauan; 3) *Live Seismogram*, menampilkan informasi hasil dari alat seosmograf secara langsung.

Setelah menganalisa aplikasi MAGMA Indonesia, ditemukan beberapa hal yang menurut Penulis merupakan kekurangan dari aplikasi MAGMA Indonesia. Kekurangan dari aplikasi MAGMA Indonesia, sebagai berikut: 1) Informasi gunungapi kurang lengkap, tidak ada informasi mengenai keadaan cuaca dan lahar; 2) Tidak ada informasi mengenai dampak letusan gunungapi hanya ada peta KRB; 3) Tidak terfokus pada gunungapi saja.

3.1.1. Gambaran Sistem Lama

Aplikasi MAGMA Indonesia adalah sebuah sistem informasi yang menampilkan informasi mengenai gunungapi seluruh Indonesia dan rekomendasi kebencanaan geologi terintegrasi yang mencakup gunungapi, gempabumi, tsunami dan gerakan tanah yang disajikan secara *quasi-realtime* dan interaktif.

Aplikasi MAGMA Indonesia tidak berfokus pada informasi gunungapi saja, melainkan hal-hal yang lain seperti yang dijelaskan di paragraf sebelumnya. Oleh karena itu menurut Penulis dari hasil analisa terdapat kekurangan pada informasi yang disajikan di aplikasi tersebut, yaitu tidak ada informasi mengenai gunungapi khususnya di Jawa Barat.

Pada Tabel 1 merupakan kebutuhan informasi pada Sistem *Data Bridge.*

Tabel 1. Daftar Kebutuhan Informasi

Informasi yang Dibutuhkan	Tujuan	Frekuensi
Gunungapi	*client-side/frontend*	Setiap melakukan *request* melalui *API* sistem *data bridge*
Cuaca disekitar gunungapi	*client-side/frontend*	Setiap melakukan *request* melalui *API* sistem *data bridge*
Letak geografis gunungapi	*client-side/frontend*	Setiap melakukan *request* melalui *API* sistem *data bridge*
History Data Gunungapi (*time series*)	*client-side/frontend*	Setiap melakukan *request* melalui *API* sistem *data bridge*

3.1.2. Kebutuhan fungsional

Kebutuhan fungsional yang harus tersedia pada Sistem *Data Bridge*, dijelaskan pada Tabel 2 di bawah ini.

Tabel 2. Daftar Kebutuhan Fungsional

No	Kode Kebutuhan	Deskripsi
client-side/frontend		
1	SD-KF-01	*API* Daftar gunungapi berbasis *GeoJSON* untuk digunakan di *google maps*
2	SD-KF-02	*API* Detail informasi gunungapi berbasis *GeoJSON* dengan properti-properti tambahan
3	SD-KF-03	*API* histori (*Time Series*) data gunungapi
4	SD-KF-04	*Sebuah* aplikasi *Worker* untuk mengambil data gunungapi setiap 5 menit sekali dan disimpan

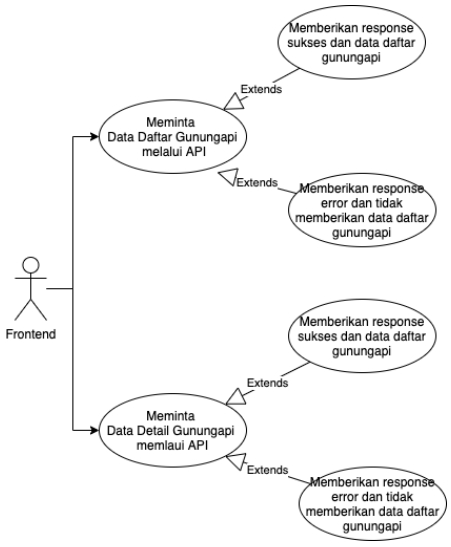
Sedangkan kebutuhan non fungsional pada Sistem *Data Bridge*, dijelaskan pada Tabel 3 di bawah ini.

Tabel 3. Daftar Kebutuhan Non Fungsional

No	Kode Kebutuhan	Deskripsi
client-side/frontend		
1	SD-KNF-01	Sistem *Data Bridge* hanya menggunakan metode *API* untuk berkomunikasi
2	SD-KNF-02	Sistem *Data Bridge* hanya menggunakan *REST* untuk arsitektur
3	SD-KNF-03	Sistem *Data Bridge* harus menyediakan *output* data yang mudah dipahami dan dibaca oleh *frontend*/*client-side*
4	SD-KNF-04	Sistem *Data Bridge* hanya menggunakan *JSON* sebagai bentuk *response* ketika memberikan data
5	SD-KNF-05	Sistem *Data Bridge* menggunakan standar format *encoding GeoJSON* untuk mengolah data berbasis spasial

3.2. *Design*

Berikutnya pada Gambar 2 diperlihatkan bahwa aktor *client-side/frontend* dapat melakukan 2 jenis *request* melalui *API* kepada Sistem *Data Bridge.*



Gambar 1. *Case Diagram client-side/frontend* dengan sistem *Data Bridge*

Dua jenis *request* tersebut yaitu: 1) Meminta Data Daftar Gunungapi, ketika *client-side/frontend* melakukan *request* untuk meminta daftar gunungapi maka sistem *Data Bridge* akan memberikan data daftar gunungapi di Jawa Barat lengkap dengan letak

geografisnya. 2) Meminta Data Detail Gunungapi, ketika *client-side/frontend* melakukan *request* untuk meminta data detail gunungapi maka sistem *data bridge* akan memberikan data informasi lengkap gunungapi.

3.3. Implementasi

Karena sistem *data bridge* berinteraksi atau berkomunikasi dengan 2 *third-party*, dalam kasus ini adalah *service API* PVMBG dan *service API* BMKG, maka Penulis membuat 2 *service*, yang pertama adalah *service* untuk berinteraksi dengan *API* PVMBG dengan nama file magma.go dan yang kedua adalah *service* untuk berinteraksi dengan *API* BMKG dengan nama file bmkg.go. Ke dua file tersebut diletakan di dalam folder *services*. Bentuk *script* pada file magma.go dan bmkg.go dapat dilihat pada Gambar 3 dan Gambar 4.

```
//VolcanicBasic method
func (m *MagmaServices) VolcanicBasic() *VolcanicBasicResponse {
    http := lib.HTTPClient{}
    volcanincBasicResponse := &VolcanicBasicResponse{}
    _, err := http.Get("/home/gunung-api", volcanincBasicResponse)
    if err != nil {
        fmt.Print(err)
    }
    return volcanincBasicResponse
}

// MagmaVar method
func (m *MagmaServices) MagmaDetail(GaCode string) *MagmaDetailResponse {
    http := lib.HTTPClient{}
    magmaDetailResponse := &MagmaDetailResponse{}
    _, err := http.Get("/home/gunung-api/var/"+GaCode, magmaDetailResponse)
    if err != nil {
        fmt.Println(err)
    }
    return magmaDetailResponse
}
```

Gambar 2 *Script* pada file magma.go (*service* PVMBG)

Di dalam *file* magma.go terdapat dua *method*, sebagai berikut: 1) *VolcanicBasic*, *method* ini digunakan untuk mendapatkan data daftar gunungapi, 2) MagmaDetail, *method* ini digunakan untuk mendapatkan data detail gunungapi.

Kedua *method* tersebut memiliki fungsi untuk berkomunikasi atau melakukan *request* ke *service API* PVMBG. Kedua *method* tersebut mengembalikan nilai berupa *response* berbentuk *JSON* yang didapat dari service *API* PVMBG.

Agar dapat melakukan *request* atau berkomunikasi, maka kedua *method* tersebut sama-sama menggunakan *library http client* bernama *resty*. *Resty* adalah salah-satu *library* di dalam *go* yang biasanya digunakan untuk keperluan berkomunikasi atau mengambil data dari *service* lain atau *third-party* berbasis *API*.

```
// MagmaServices struct
type BMKGServices struct{}

// ForcastWeathers method
func (bmkg *BMKGServices) WeatherForcest() (*resty.Response, error) {
    client := resty.New()

    resp, err := client.R().
        EnableTrace().
        Get(viper.GetString("bmkg.api_url"))

    return resp, err
}
```

Gambar 3. *Script* pada file bmkg.go (*service* BMKG)

File bmkg.go memiliki fungsi yang sama dengan *script* yang ada di dalam *file* magma.go, pada Gambar 4 adalah *script* untuk berkomunikasi dengan *service API* BMKG. Di dalam *file* bmkg.go terdapat satu *method* yang dibuat dan digunakan untuk berkomunikasi dengan *service API* BMKG. Yang membedakan dengan *service API* PVMBG adalah bentuk response yang diberikan. Jika pada *service* API PVMBG diberikan *response* berbentuk *JSON*, maka pada *service API* BMKG yang diberikan adalah *response* berbentuk *XML*.

Di dalam *repository* dibuat *script* untuk memproses data yang didapat melalui *service third-party* (PVMBG dan BMKG), oleh karena itu dalam *repository* sebuah *service* akan dipanggil dan datanya akan diproses atau diolah. Penulis membuat dua *repository* atau dua *file*. Yang pertama bernama bmkg.go dan yang kedua adalah magma.go. Kedua file tersebut diletakan di dalam folder *repositores*. Berikut adalah *script* pada file magma.go dan bmkg.go bisa dilihat pada Gambar 5 dan Gambar 6.

```
// BasicVolcanicData basic volcanic
func (m *MagmaRepository) BasicVolcanicData() GeoBasicVolcanicData {
}

// BasicVolcanicData basic volcanic
func (m *MagmaRepository) GetailVolcanicData(gaCode string, gaKoterGapi string) interface{
    } {
}
```

Gambar 4. Method pada file magma.go (*repository*)

```
func (m *BMKGRepository) GetVolcanoWeatherForcest(weathersForcest WeatherForcest,
    GaKoterGapi string) WeatherArea {
}

func (m *BMKGRepository) GetWeatherForcestData() WeatherForcest {
}
```

Gambar 5. Method pada file bmkg.go (*repository*)

Pembuatan *controller* diimplemntasikan berupa *file* magma.go dan diletakan di dalam *folder* bernama *controllers*, di dalam *file* magma.go, Penulis membuat *script* untuk memanggil sebuah *repository*. Gambarannya bisa dilihat pada Gambar 7.

```
// MagmaController mehtod
type MagmaController struct{}

//GetBasicVolcanicData method
func (m *MagmaController) GetBasicVolcanicData() repositories.GeoBasicVolcanicData {
    magmaRepository := repositories.MagmaRepository{}
    return magmaRepository.BasicVolcanicData()
}

//GetDetailVolcanicData method
func (m *MagmaController) GetDetailVolcanicData(gaCode string, gaKoterGapi string) interface{} {
    magmaRepository := repositories.MagmaRepository{}
    return magmaRepository.GetailVolcanicData(gaCode, gaKoterGapi)
}
```

Gambar 6. *Script Magma Controller*

Pada Gambar 7 terdapat beberapa *script* atau langkah-langkah dalam membuat *controller* di dalam sistem *data bridge*, sebagai berikut: 1) Membuat *struct* MagmaController, tujuan untuk menjadi *class/instance* yang menampung *method;* 2) Membuat *method* bernama GetBasicVolcanicData yang menjadi bagian atau *attribute* dari *struct* MagmaController, di

dalam *method* GetBasicVolcanicData, dipanggil *repository* bernama magmaRepository beserta *method* bernama BasicVolcanicData yang dimana jika memanggil *method* tersebut maka akan dikembalikan nilai berupa data gunungapi dalam *format GeoJSON*; 3) Membuat *method* bernama GetDetailVolcanicData, *method* ini memiliki 2 *argument*, *argument* 1 dan 2 bertipe *data string*, di dalam *method* ini dipanggil magmaRepository beserta *method*-nya GetDetailVolcanicData yang mana jika dipanggil *method* tersebut maka akan dikembalikan nilai berupa data detail gunungapi.

Dalam membuat sebuah *routing*, dibuat *file* bernama route.go yang disimpan di dalam *folder route*. Pada *file* tersebut diisi *script* yang berfungsi sebagai *entrypoint* dalam sistem. Berikut *script* membuat *routing* bisa dilihat pada Gambar 8.

```
// InitRoute create app route
func InitRoute() {

    e := echo.New()

    // cors setting
e.Use(middleware.CORSWithConfig(middleware.CORSConfig{
    AllowOrigins: []string{"*"},
    AllowHeaders: []string{echo.HeaderOrigin, echo.HeaderContentType, echo.HeaderAccept},
}))

e.Use(middleware.BasicAuth(func(username, password string, c echo.Context) (bool, error) {
    if username == "stmik" && password == "amik" {
        return true, nil
    }
    return false, nil
}))

// Middleware
e.Use(middleware.Logger())
e.Use(middleware.Recover())

e.GET("/geo-volcanic-data", func(c echo.Context) error {
    magmaController := &controllers.MagmaController{}
    return c.JSON(http.StatusOK, magmaController.GetBasicVolcanicData())
})

e.GET("/volcanic-detail-data/:gaCode/:gaKoterGapi", func(c echo.Context) error {
    magmaController := &controllers.MagmaController{}
    gaCode := c.Param("gaCode")
    gaKoterGapi := c.Param("gaKoterGapi")
    return c.JSON(http.StatusOK, magmaController.GetDetailVolcanicData(gaCode, gaKoterGapi))
})

    e.Logger.Fatal(e.Start(":" + viper.GetString("app.port")))
}
```

Gambar 7. *Script* Membuat *Routing Sistem Data Bridge*

Untuk menjalankan atau menghidupkan sistem bisa dengan cara membuka terminal lalu menjalankan sebuah *script* atau perintah didalam terminal tersebut, berikut pada Gambar 9 adalah perintah yang dijalankan pada terminal bertujuan untuk menjalankan sistem.

```
macs-MacBook-Pro-2:Projects mac$ cd Web/data-bridge/
macs-MacBook-Pro-2:data-bridge mac$ go run main.go

   ____    __
  / __/___/ /  ___
 / _// __/ _ \/ _ \
/___/\__//_//_/\___/ v4.1.17
High performance, minimalist Go web framework
https://echo.labstack.com
____________________________________O/_______
                                    O\
⇨ http server started on [::]:8081
```

Gambar 8. Terminal dengan *script* untuk menjalankan sistem

3. Pengujian

Proses pengujian dibantu dengan aplikasi bernama *postman*, dengan tujuan supaya mudah dalam melakukan pengujian dan memastikan hasil yang diharapkan sesuai.

Tabel 4. Daftar *Test Case* Sistem *Data Bridge*

No	*Test Case*	Hasil yang diharapkan	Hasil yang didapatkan	Status
1	Pengujian *API* dengan endpoint `geo-volcaninc-data`	Mendapatkan *response* status *header* 200 dan mendapatkan data daftar gunungapi dengan *format GeoJSON*	Sistem memberikan *response* status *header* 200 dan memberikan data daftar gunungapi dengan *format GeoJSON*	Berhasil
2	Pengujian *API* dengan endpoint `volcaninc-detail-data/{gaCode}/{gaKoterGapi}`	Mendapatkan *response* status *header* 200 dan mendapatkan data detail gunungapi dengan *output encoding JSON*	Sistem memberikan *response* status 200 dan memberikan data detail gunungapi dengan *output encoding JSON*	Berhasil

4. Kesimpulan

Penelitian ini dapat disimpulkan sebagai berikut. 1) Dengan Sistem *Data Bridge* yang telah dibangun da dilakukan proses *development* serta pengujian maka *Sistem Data Bridge* mampu untuk memberikan data

berupa informasi lengkap mengenai gunungapi untuk masyarakat, informasi yang diberikan seperti nama gunungapi, letak geografis, status, keadaan cuaca dan lain-lain. 2) Sistem *Data Bridge* dikembangkan dalam bentuk *Open API* sehingga akses data mengenai informasi gunungapi dapat diperoleh dengan mudah karena, pengembang yang lain hanya cukup melakukan komunikasi dengan Sistem *Data Bridge* melalui *API* dan sudah bisa mendapatkan data lengkap mengenai informasi gunungapi. Dengan demikian informasi bisa diperoleh dengan cepat karena sudah tidak butuh lagi mekanisme komunkasi dengan *API* PVMBG & *API* BMKG, semua hal itu sudah di atur dan di *handle* oleh Sistem *Data Bridge.*

Ucapan Terima kasih

Terima kasih kepada semua pihak yang telah membantu dan terlibat pada penelitian ini, khususnya Saudari Hamdan yang telah mlakukan koding dan pengujian terhadap sistem serta Bapak Herry Kuswandarto dari PVMBKG yang telah membantu penyediaan data dan memberikan referensi, materi pendukung serta kesempatan yang telah diberikan untuk melakukan penelitin ini.

Daftar Pustaka

[1] Dinar Astari *et al*, "Pemetaan Zona Bahaya Aliran Piroklastik Gunung Merapi, Jawa Tengah dan Sekitarnya menggunakan Aplikasi Titan2D", *Jurnal Geosains dan Teknologi,* Vol. 5 No. 1, pp. 76-82, 2022

[2] Hasanuddin, H. Asgar, B. Hartono, "Rancang Bangun *Rest* API Aplikasi *Weshare* Sebagai Upaya Mempermudah Pelayanan Donasi Kemanusiaan", *JINTEKS*, vol. 4, no. 1, pp. 8-14, Feb. 2022.

[3] Irian, Yusuf Yudhistira, "Implementasi *Application Programming Interface* (API) Kawal Corona Sebagai Media Informasi Pandemik Covid-19 Berbasis Android", *Jurnal Sistem Informasi dan Teknologi Peradaban (JSITP),* Vol. 2, No. 1, 2021

[4] Annafi' Franz *et al*,"*Web Design and Application Programming Interface (API) Smart Farming Application*", *TEPIAN*, Vol. 2 No. 1, 2021

[5] Rizki Kurnia, *et al*,"Validitas E-Modul Fisika Terintegrasi Bencana Gunung Meletus Berbasis Model *Inquiry Based Learning* untuk Meningkatkan Sikap Kesiapsiagaan Peserta Didik", *Jurnal Penelitian dan Pembelajaran Fisika,* Vol 6, No.1 pp. 73-80, 2020

[6] Sri Ratni Salatun *et al*, "Analisis Tingkat Kerentanan Gunung Api Awu Di Kabupaten Kepulauan Sangihe", *Jurnal Spasial,* Vol 6. No. 3, 2019

[7] Lisa Christie Gosal *et al*, "Analisis Spasial Tingkat Kerentanan Bencana Gunung Api Lokon Di Kota Tomohon", *Jurnal Spasial,* Vol 5. No. 2, 2018

[8] Javatpoint, "Web API", *Retrieved from Javatpoint*: https://www.javatpoint.com/web-api, 2018.

[9] Arni, U. D., "Perbedaan API dengan *Web Service*", *Retrieved from Garuda Cyber Indonesia*: https://garudacyber.co.id/artikel/284-perbedaan-api-dengan-web-service, 2018

[10] Departemen Energi dan Sumber Daya Mineral, "Tipe Gunung Api di Indonesia", *Retrieved from ESDM:* https://magma.esdm.go.id/v1/edukasi/tipe-gunung-api-di-indonesia-a-b-dan-c

[11] "git-scm.com,". *Retrieved from*: https://git-scm.com/book/en/v2/Getting-Started-What-is-Git. [Accessed 15 03 2021].

[12] "github.com/go-resty". *Retrieved from*: https://github.com/go-resty/resty