KERJA PRAKTIK - IF234603

Perancangan dan Implementasi Sistem Modul MyITS Space Management untuk Manajemen Ruangan pada Lingkungan ITS

Direktorat Pengembangan Sistem Informasi
Jl. Kampus Sukolilo, Gedung Pusat Riset Lantai 4, Jl. Teknik Kimia, Keputih, Sukolilo, Surabaya, Jawa Timur, 60117
Periode: 15 September 2023 - 10 Desember 2023

Oleh:

Muhammad Ismail 5025201223
Eldenabih Tavirazin Lutvie 5025201213

Pembimbing Jurusan
Hadziq Fabroyir, S.kom., Ph.D.
Pembimbing Lapangan
Muhammad Fattah Na'im PR, S.Kom.

DEPARTEMEN TEKNIK INFORMATIKA
Fakultas Teknologi Elektro dan Informatika Cerdas
Institut Teknologi Sepuluh Nopember
Surabaya 2023

KERJA PRAKTIK - IF234603

Perancangan dan Implementasi Sistem Modul MyITS Space Management untuk Manajemen Ruangan pada Lingkungan ITS

Direktorat Pengembangan Sistem Informasi
Jl. Kampus Sukolilo, Gedung Pusat Riset Lantai 4, Jl. Teknik Kimia, Keputih, Sukolilo, Surabaya, Jawa Timur, 60117
Periode: 15 September 2023 - 10 Desember 2023

Oleh:

Muhammad Ismail 5025201223
Eldenabih Tavirazin Lutvie 5025201213

Pembimbing Jurusan
Hadziq Fabroyir, S.kom., Ph.D.
Pembimbing Lapangan
Muhammad Fattah Na'im PR, S.Kom.

DEPARTEMEN TEKNIK INFORMATIKA
Fakultas Teknologi Elektro dan Informatika Cerdas
Institut Teknologi Sepuluh Nopember
Surabaya 2023

[Halaman ini sengaja dikosongkan]

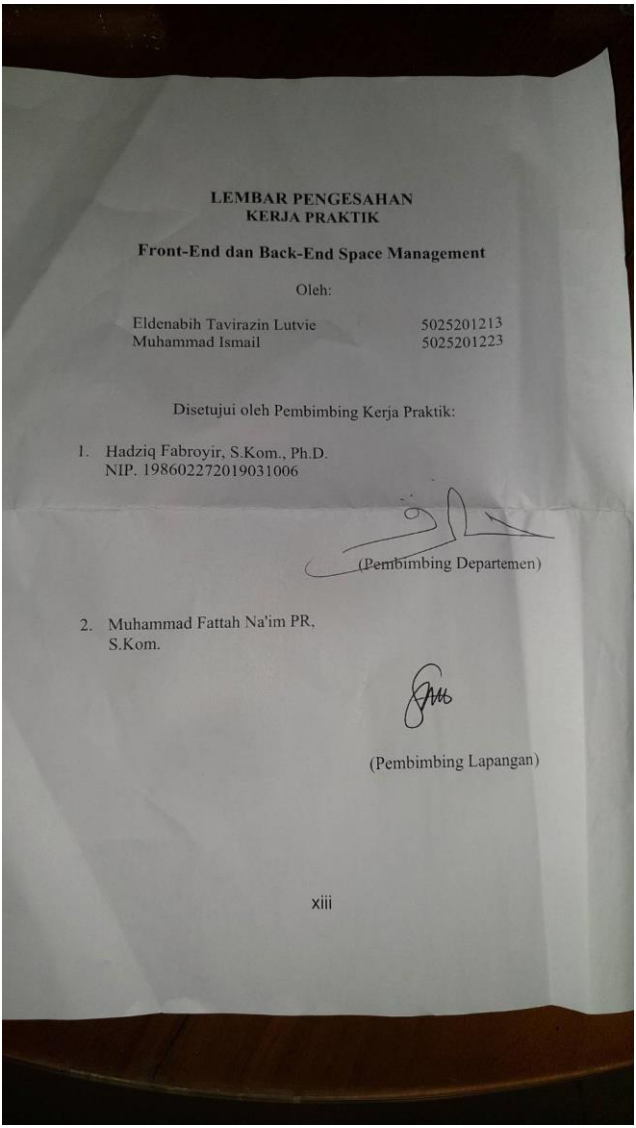
DAFTAR ISI

[Halaman ini sengaja dikosongkan]

[Halaman ini sengaja dikosongkan]

[Halaman ini sengaja dikosongkan]

LEMBAR PENGESAHAN
KERJA PRAKTIK



LEMBAR PENGESAHAN
KERJA PRAKTIK

Front-End dan Back-End Space Management

Oleh:

Eldenabih Tavirazin Lutvie	5025201213
Muhammad Ismail	5025201223

Disetujui oleh Pembimbing Kerja Praktik:

1. Hadziq Fabroyir, S.Kom., Ph.D.
   NIP. 198602272019031006

(Pembimbing Departemen)

2. Muhammad Fattah Na'im PR, S.Kom.

(Pembimbing Lapangan)

xiii

[Halaman ini sengaja dikosongkan]

Perancangan dan Implementasi Sistem Modul MyITS Space Management untuk Manajemen Ruangan pada Lingkungan ITS

Nama Mahasiswa : Muhammad Ismail
NRP : 5025201223
Nama Mahasiswa : Eldenabih Tavirazin Lutvie
NRP : 5025201213
Departemen : Teknik Informatika FTEIC-ITS
Pembimbing Departemen : Hadziq Fabroyir, S.Kom., Ph.D.
Pembimbing Lapangan : Muhammad Fattah Na'im PR, S.Kom.

ABSTRAK

Laporan kerja praktik ini membahas tentang perancangan dan implementasi sistem modul MyITS Space Management untuk manajemen ruangan pada lingkungan ITS. Sistem ini dirancang untuk memudahkan administrator dalam mengelola data aset ruang di ITS. Metode yang digunakan dalam pengembangan sistem meliputi perumusan masalah, studi literatur, analisis dan perancangan sistem, implementasi, pengujian dan evaluasi, serta penarikan kesimpulan.

Perancangan sistem menggunakan arsitektur Domain Driven Design dan pattern CQRS untuk pemisahan operasi baca-tulis. Implementasi sistem menggunakan bahasa pemrograman Golang dengan framework Gin untuk backend API server, dan NextJS sebagai frontend. Database yang digunakan adalah Azure Data Studio.

Hasil evaluasi menunjukkan bahwa sistem mampu memenuhi fungsionalitas yang diharapkan, seperti menampilkan daftar dan detail ruang, melakukan CRUD

ruang dan fasilitas, serta menangani error dengan baik. Dengan demikian, sistem ini dapat membantu administrator dalam mengelola data ruangan di ITS secara efektif. Pengembangan lebih lanjut dapat difokuskan pada penambahan fitur dan performa sistem.

Kata Kunci : API, CRUD, MyITS Space Management

[Halaman ini sengaja dikosongkan]

KATA PENGANTAR

Puji syukur penulis panjatkan kepada Allah SWT atas penyertaan dan karunia-Nya sehingga penulis dapat menyelesaikan salah satu kewajiban penulis sebagai mahasiswa Departemen Teknik Informatika ITS yaitu Kerja Praktik yang berjudul: Perancangan dan implementasi sistem modul MyITS Space Management untuk Manajemen Ruangan pada Lingkungan ITS.

Penulis menyadari bahwa masih banyak kekurangan baik dalam melaksanakan kerja praktik maupun penyusunan buku laporan kerja praktik ini. Namun penulis berharap buku laporan ini dapat menambah wawasan pembaca dan dapat menjadi sumber referensi.

Melalui buku laporan ini penulis juga ingin menyampaikan rasa terima kasih kepada orang-orang yang telah membantu menyusun laporan kerja praktik baik secara langsung maupun tidak langsung antara lain:

1. Kedua orang tua penulis.
2. Bapak Hadziq Fabroyir, S.Kom., Ph.D. selaku dosen pembimbing kerja praktik sekaligus koordinator kerja praktik.
3. Mas Muhammad Fattah Na'im PR, S.Kom.. selaku pembimbing lapangan selama kerja praktik berlangsung.
4. Teman-teman penulis yang senantiasa memberikan semangat ketika penulis melaksanakan KP.

Surabaya, 3 Januari 2024
Muhammad Ismail dan Eldenabih Tavirazin Lutvie

[Halaman ini sengaja dikosongkan]

BAB I
PENDAHULUAN

1.1. Latar Belakang

Saat ini dunia telah berkembang menjadi era digital. Semua layanan dapat disajikan secara online dengan adanya teknologi yang dapat dimanfaatkan untuk mempermudah manusia dalam melakukan layanan tersebut. Bila dibandingkan dengan yang dulu, kita membaca koran atau majalah dengan membeli fisik koran atau majalah. Kini, untuk membaca koran kita dapat membuka website atau aplikasi penyedia koran untuk membacanya. Website atau aplikasi tidak hanya terbatas untuk membaca koran. Dengan website atau aplikasi manusia seakan-akan dapat melakukan segala hal dalam satu tempat. Banyak sekali fungsi dari website atau aplikasi, sebagai contohnya pembayaran pajak, e-banking, pembelajaran, dan lain lain.

Di zaman sekarang, akses terhadap internet semakin mudah dan juga memudahkan berbagai kegiatan di sekitar kita salah satunya adalah peminjaman ruangan. Dengan kemudahan era digital sekarang, kita tidak perlu repot-repot untuk menghubungi seseorang untuk meminta jadwal peminjaman ruangan karena sekarang sudah banyak penyewaan yang menggunakan website sebagai platform untuk menyewakan tempatnya. Di lingkup ITS, terdapat berbagai ruangan kosong yang banyak digunakan untuk peminjaman ruangan. Selain dari itu, ITS juga memiliki website tersendiri untuk melakukan peminjaman ruangan. Akan tetapi, seiring

berjalannya waktu permintaan untuk peminjaman ruangan semakin tinggi sehingga membutuhkan instrumen untuk memanajemen website peminjaman ruangan. Dengan adanya KP ini, kami diberi kesempatan untuk merancang dan mengimplementasikan website dalam memanajemen peminjaman ruangan di ITS.

1.2. Tujuan

Tujuan kerja praktik ini adalah menyelesaikan kewajiban nilai kerja praktik sebesar 2 sks dan membantu DPTSI ITS untuk menyelesaikan permasalahan manajemen peminjaman ruangan dalam bentuk website

1.3. Manfaat

Manfaat yang diperoleh dengan adanya website manajemen peminjaman ruangan ITS antara lain adalah mempermudah pengelola ruangan untuk mengatur workload dalam peminjaman ruangan di lingkup ITS. Dengan begitu, pengelola ruangan dapat menghemat waktunya dalam mengurus ruangan.

1.4. Rumusan Masalah

Rumusan masalah dari kerja praktik ini adalah sebagai berikut:

1. Bagaimana desain dan arsitektur sistem dalam perancangan website MyITS Space Management?

2. Bagaimana bentuk ERD dan proses bisnis dalam MyITS Space Management?

1.5. Lokasi dan Waktu Kerja Praktik

Pengerjaan KP ini dilakukan secara *Hybrid*, menyesuaikan kebutuhan tim.

Adapun kerja praktik dimulai pada tanggal 15 September 20203 hingga 10 Desember 2023.

1.6. Metodologi Kerja Praktik

Metodologi dalam pembuatan buku kerja praktik meliputi :

1.6.1. Perumusan Masalah

Untuk mengetahui kebutuhan dari website, kami mengikuti rapat bersama tim developer. Pada saat rapat kami dijelaskan bagaimana konsep dan proses dalam penyewaan ruangan di ITS. Setelah dijelaskan, pemimpin tim developer merumuskan fitur - fitur apa saja yang akan diterapkan pada website yang akan dibuat.

1.6.2. Studi Literatur

Setelah mendapat gambaran bagaimana sistem tersebut berjalan, kami diberitahu tinjauan apa saja yang akan diimplementasikan untuk membuat website beroperasi. Tinjauan yang dipakai meliputi penerapan Bahasa Pemrograman Go untuk sisi Backend dengan menerapkan sistem desain CQRS dan Domain Data Driven. Selain itu, dari segi Frontend, kami menggunakan NextJS sebagai framework dalam pengembangan website. Selain

dari itu, kami juga dijelaskan aturan-aturan dalam menuliskan dokumentasi agar pengembangan aplikasi dapat mudah dipahami oleh pengembang yang lain.

1.6.3. Analisis dan Perancangan Sistem

Setelah tinjauan yang dipakai telah diberitahu, untuk merancang sistem yang baik perlu adanya sebuah desain arsitektur sistem. Pada website ini tim developer setuju menggunakan arsitektur DDD (Domain Data Driven) dengan pattern CQRS (Command Query Responsibility Segregation)

1.6.4. Implementasi Sistem

Implementasi merupakan realisasi dari tahap perancangan. Pada tahap ini kami Melakukan *deployment* pada aplikasi yang telah dibuat oleh tim developer.

1.6.5. Pengujian dan Evaluasi

Setelah website yang telah direncanakan telah jadi, perlu adanya evaluasi untuk menguji apakah website sesuai dengan harapan client. Jika masih belum sesuai atau perlu menambah fitur, rapat akan dilakukan lagi untuk mem-*floor*-kan fitur - fitur apa saja yang perlu diperbaiki atau ditambah.

1.6.6. Kesimpulan dan Saran

Pengujian yang dilakukan ini telah memenuhi syarat yang diinginkan, dan berjalan dengan baik dan lancar.

1.7. Sistematika Laporan

1.7.1. Bab I Pendahuluan

Bab ini berisi latar belakang, tujuan, manfaat, rumusan masalah, lokasi dan waktu kerja praktik, metodologi, dan sistematika laporan.

1.7.2. Bab II Profil Perusahaan

Bab ini berisi gambaran umum Direktorat Pengembangan Sistem dan Informasi mulai dari profil, lokasi perusahaan.

1.7.3. Bab III Tinjauan Pustaka

Bab ini berisi dasar teori dari teknologi yang digunakan dalam menyelesaikan proyek kerja praktik.

1.7.4. Bab IV Analisis dan Perancangan Infrastruktur Sistem

Bab ini berisi mengenai tahap analisis sistem aplikasi dalam menyelesaikan proyek kerja praktik.

1.7.5. Bab V Implementasi Sistem

Bab ini berisi uraian tahap - tahap yang dilakukan untuk proses implementasi aplikasi.

1.7.6. Bab VI Pengujian dan Evaluasi

Bab ini berisi hasil uji coba dan evaluasi dari aplikasi yang telah dikembangkan selama pelaksanaan kerja praktik.

1.7.7. Bab VII Kesimpulan dan Saran

Bab ini berisi kesimpulan dan saran yang didapat dari proses pelaksanaan kerja praktik.

[Halaman ini sengaja dikosongkan]

BAB II
PROFIL PERUSAHAAN

2.1. Profil Direktorat Pengembangan Sistem dan Informasi

Direktorat Pengembangan Sistem dan Informasi atau juga bisa disingkat DPTSI merupakan instansi milik Institut Teknologi Sepuluh Nopember yang bertanggung jawab tentang semua hal yang berkaitan dengan pengembangan sistem informasi di lingkup ITS. DPTSI bertugas melaksanakan urusan ITS bidang teknologi dan sistem informasi mengikuti arahan dari petinggi-petinggi ITS dan berdiri secara independen.

2.2. Lokasi

Kampus Sukolilo, Gedung Pusat Riset Lantai 4, Jl. Teknik Kimia, Keputih, Sukolilo, Surabaya, Jawa Timur, 60117

[Halaman ini sengaja dikosongkan]

BAB III
TINJAUAN PUSTAKA

3.1. Pemrograman Web

Web atau World Wide Web adalah ruang informasi yang berisi dokumen dan resource web lainnya yang dapat diidentifikasi melalui sebuah URL (Uniform Resource Locators, contohnya www.google.com) dan diakses ketika terkoneksi dengan internet. Halaman penyedia dokumen di dalam web dapat disebut sebagai website yang dapat terkoneksi satu dengan lainnya (hyperlink).

Pemrograman web adalah proses pembuatan halaman tersebut agar bisa diakses oleh semua orang. Dalam pembuatan website, diperlukan sebuah standar pada website agar semua orang dapat membaca informasi dalam keadaan yang berbeda. Standar tersebut adalah HTML (Hypertext Markup Language). Jadi pemrograman web memiliki tugas untuk menciptakan suatu halaman sesuai standar HTML agar semua orang memiliki akses pada informasi di dalam halaman tersebut.

3.2. Javascript

Javascript adalah sebuah bahasa tingkat tinggi yang dinamis. Javascript memiliki banyak sekali fungsionalitas seperti web application, backend, desktop application, internet of things (IoT), dan lain - lain. Pada buku kerja praktik ini javascript digunakan untuk client side scripting language yang tertanam pada HTML sebuah website. Javascript juga

memiliki banyak library yang dapat digunakan contohnya nodejs, axiosjs, bluebirdjs, vuejs, angularjs, reactjs, animatejs, dan lain - lain.

3.3. Azure Data Studio

Merupakan salah satu sistem manajemen relasional basis data SQL yang bersifat open source., cross-platform untuk analisis data [4]. Azure Data Studio digunakan untuk menyimpan data - data yang dapat saling berelasi dengan data yang lain. Untuk melakukan operasi CRUD (Create, Read, Update, Delete) memerlukan query, lalu query tersebut dikirim ke dalam database Azure Data Studio. Pada saat query tersebut sampai pada database, database akan mengolah query tersebut dan diterapkan sesuai perintah query.

3.4. Go Language

Go, atau Golang, merupakan bahasa pemrograman yang dikembangkan oleh Google dengan fokus pada keterbacaan kode, produktivitas pengembangan, dan performa tinggi. Diperkenalkan secara resmi pada tahun 2009, Go didesain dengan sintaksis yang sederhana, membuatnya mudah dipahami oleh pemula maupun pengembang berpengalaman [2]. Kelebihan lainnya meliputi waktu kompilasi yang cepat, manajemen memori otomatis, serta dukungan bawaan untuk pemrograman konkuren dengan goroutines dan channels, memungkinkan pengembangan aplikasi yang efisien dan scalable.

Selain itu, Go menonjol dengan dukungan cross-platform, memungkinkan aplikasi yang dikembangkan di Go dapat dijalankan di berbagai sistem operasi tanpa banyak modifikasi. Pustaka standar yang kaya fitur, mencakup pengolahan string, HTTP server, enkripsi, dan banyak lagi, juga menjadi daya tarik bagi pengembang. Go umumnya digunakan dalam pengembangan server backend, perangkat lunak infrastruktur, alat pengembangan, dan aplikasi berkinerja tinggi. Banyak perusahaan terkemuka telah mengadopsi Go, mengakui keunggulannya dalam menghadirkan aplikasi yang efisien, handal, dan mudah dipelihara.

3.5. NextJS

Next JS adalah framework full-stack yang memungkinkan pembangunan tampilan situs web dan menangani proses rendering serta pengelolaan basis data [1]. Framework ini memungkinkan pengembang untuk membuat aplikasi web dengan fitur-fitur terbaru dari React dan integrasi dengan tooling JavaScript berbasis Rust yang powerful untuk build yang lebih cepat. NextJS memiliki beberapa fitur utama seperti built-in optimizations, data fetching, node.js & edge runtimes, advanced routing & nested layouts, dan dynamic HTML streaming. NextJS juga memiliki kemampuan server-side rendering dan static site generation yang memungkinkan perubahan konten yang cepat, sehingga cocok digunakan untuk platform publishing, aplikasi berita, website dokumentasi, dan platform hiburan. Referensi paper

mengenai NextJS tidak ditemukan dalam hasil pencarian.

3.6. Command Query Responsibility Segregation

CQRS (Command Query Responsibility Segregation) adalah sebuah pola arsitektur yang memisahkan antara operasi menulis (command) dan membaca (query) dalam sebuah aplikasi [5]. Prinsip ini memungkinkan pengembang untuk menggunakan model yang berbeda untuk membaca dan menulis data, sehingga memungkinkan optimalisasi terhadap masing-masing model. CQRS berguna dalam mengelola kompleksitas aplikasi dengan memisahkan proses penulisan dan pembacaan data, sehingga memungkinkan pengembangan fitur-fitur yang lebih fleksibel dan skalabilitas yang lebih baik.

3.7. HTML

Bahasa standar internasional yang digunakan untuk membuat halaman web. HTML adalah singkatan dari Hyper Text Markup Language, yang berarti tata cara penulisan yang digunakan dalam dokumen web, atau dapat juga diartikan sebagai bahasa yang digunakan untuk merancang sebuah halaman web [3]. HTML menggambarkan struktur dan isi semantik dari sebuah dokumen. HTML biasanya digabungkan dengan css dan javascript. css untuk memperindah tampilan

[Halaman ini sengaja dikosongkan]

BAB IV
ANALISIS DAN PERANCANGAN INFRASTRUKTUR SISTEM

4.1. Analisis Sistem

Pada bab ini akan dijelaskan mengenai tahapan dalam membangun infrastruktur aplikasi MyITS Space Management yaitu analisis dari infrastruktur sistem yang akan dibangun. Hal tersebut dijelaskan ke dalam dua bagian, definisi umum aplikasi dan perancangan infrastruktur sistem.

4.1.1. Definisi Umum Aplikasi

Aplikasi MyITS Space Management adalah sistem manajemen ruangan yang dirancang untuk memudahkan Administrator dalam mengelola data aset ruang di lingkungan ITS. Aplikasi ini memberikan kemampuan untuk melihat, menambahkan, mengubah, dan menghapus informasi terkait ruang dan fasilitasnya.

4.1.2. Definisi Fungsional Aplikasi

Fungsi utama aplikasi MyITS Space Management melibatkan:

- Daftar Ruang: Menampilkan halaman menu Daftar Ruang untuk melihat dan mengelola data aset ruang.
- Tambah Ruang: Memberikan kemampuan kepada Administrator untuk menambahkan data aset ruang baru ke dalam sistem.

- Detail Area: Menyediakan informasi rinci terkait ruang dan fasilitas yang dapat diakses oleh Administrator.
- Ubah Data Ruang: Memungkinkan Administrator untuk mengubah informasi terkait ruang, sehingga data yang tidak sesuai atau mengalami perubahan dapat diperbarui.
- Ubah Data Fasilitas: Memberikan fleksibilitas kepada Administrator untuk mengubah data fasilitas di dalam ruang, sehingga data yang tidak sesuai dapat diubah dan diperbarui.

4.2. Perancangan Infrastruktur Sistem

Perancangan infrastruktur sistem mencakup desain sistem secara menyeluruh, termasuk tata letak, interaksi antar komponen, dan pengaturan umum aplikasi.

4.2.1. Desain Sistem

Desain sistem MyITS Space Management mencakup komponen-komponen berikut:

- Antarmuka Pengguna (UI): Antarmuka pengguna didesain intuitif untuk memastikan Administrator dapat dengan mudah mengakses dan mengelola informasi ruang.
- Database: Digunakan untuk menyimpan dan mengelola data aset ruang, termasuk data fasilitas.
- Backend Server: Menangani logika bisnis, memproses permintaan dari Antarmuka

Pengguna, dan berkomunikasi dengan database.

- Fungsionalitas CRUD: Menerapkan fungsi Create, Read, Update, dan Delete untuk pengelolaan data ruang dan fasilitas.

Dengan perancangan ini, aplikasi dapat memberikan pengalaman pengguna yang efisien dan efektif sesuai dengan *user story* yang telah ditetapkan.

[Halaman ini sengaja dikosongkan]

BAB V
IMPLEMENTASI SISTEM

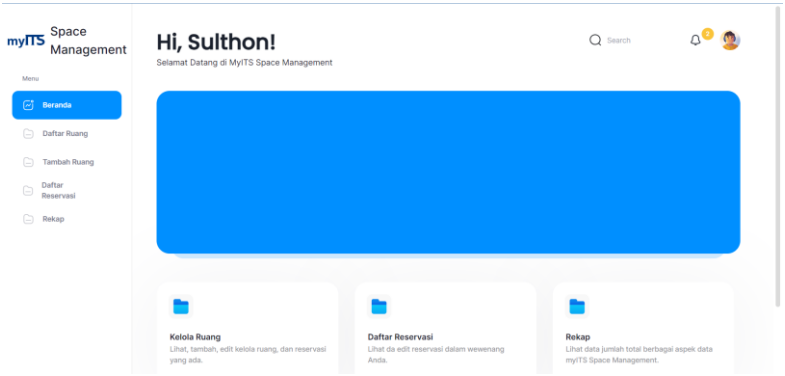
Bab ini membahas tentang implementasi dari sistem MyITS Space Management yang kami buat. Implementasi ini akan dibagi ke dalam beberapa bagian, yaitu bagian implementasi antarmuka pengguna dan implementasi sistem.

5.1. Implementasi Antarmuka Pengguna

Pada bagian ini akan ditampilkan antarmuka halaman MyITS Space Management.

5.1.1. Halaman Beranda

Pada halaman ini, implementasi antarmuka pengguna dirancang untuk menampilkan halaman utama sistem.



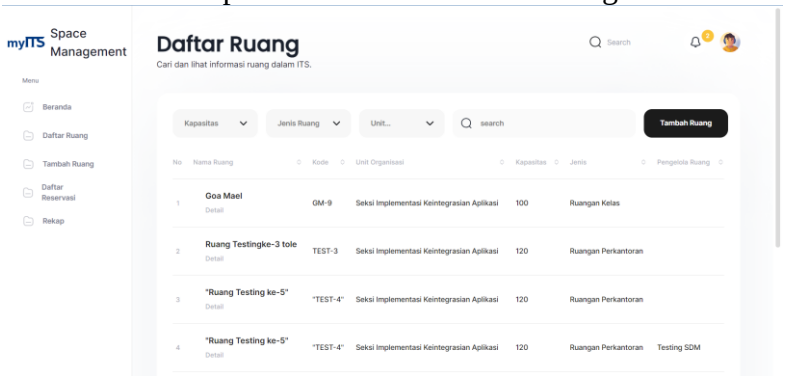
Gambar 5.1: Halaman Beranda

5.1.2. Halaman Daftar Ruang

Pada halaman ini, implementasi antarmuka pengguna dirancang untuk menampilkan daftar

ruang dengan detail yang relevan. Beberapa komponen antarmuka termasuk:

- Tabel Daftar Ruang: Menampilkan informasi ruang seperti nama ruang, jenis, kapasitas, dan opsi pengelolaan.
- Tombol Tambah Ruang: Memberikan akses cepat untuk menambahkan ruang baru.



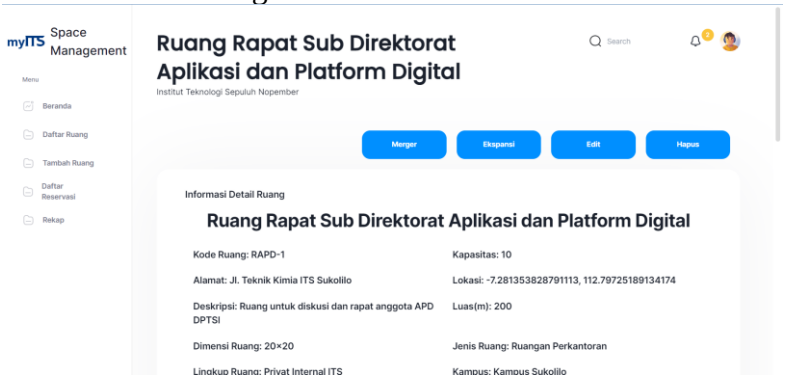
Gambar 5.2: Halaman Daftar Ruang

5.1.3. Halaman Detail Ruang

Halaman ini memperlihatkan detail lengkap terkait suatu ruang, termasuk fasilitas yang terkait. Antarmuka mencakup:

- Informasi Detail Ruang: Menampilkan semua informasi terkait ruang, seperti nama, kode, kapasitas, dan deskripsi.
- Daftar Fasilitas: Tabel yang menampilkan fasilitas-fasilitas yang terkait dengan ruang tersebut.

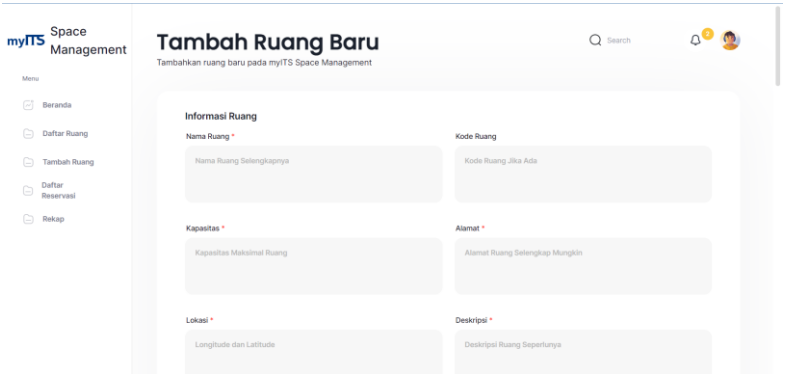
- Tombol dan Form Edit Ruang: Memberikan akses cepat untuk melakukan perubahan pada ruang.



Gambar 5.3: Halaman Detail Ruang

5.1.4. Halaman Tambah Ruang

Pada halaman ini, terdapat formulir untuk menambahkan ruang baru. Form ini mencakup semua atribut yang diperlukan dan memberikan feedback yang jelas jika terdapat kesalahan validasi.



Gambar 5.4: Halaman Tambah Ruang

5.2. Implementasi Sistem

Implementasi sistem backend melibatkan berbagai komponen dan logika bisnis untuk menangani permintaan dari antarmuka pengguna dan berinteraksi dengan database. Untuk DBMS yang digunakan pada kerja praktik ini adalah Azure Data Studio.

5.2.1 Implementasi API Daftar Ruang

Implementasi API daftar ruang berfokus pada pengambilan dan pengolahan data pada tabel ruang di dalam database.

5.2.1.1 GET Daftar Ruang By Page

Implementasi GET daftar ruang berdasarkan halaman melibatkan query menggunakan framework GORM dan juga GIN untuk controller dan routenya yang mana digunakan untuk mendapatkan list ruang berdasarkan halaman tertentu. Berikut merupakan implementasinya.

```
func (q *DaftarRuangQuery) GetDaftarRuangByPage(ctx
context.Context, currentPage int) ([]queries.Ruang, error) {
   query := q.db.
      Table(RuangTableName).
      Select("ruang.id_ruang as id_ruang,lr.lingkup_ruang, jr.jenis_ruang,
pm.jenis_pemanfaatan, kp.kampus as lokasi_kampus,ruang.nama as
nama, kode, kapasitas, jr.jenis_ruang as jenis_ruang, uo.nama as
unit_organisasi, sdm.nama as pengelola_ruang").
      Joins("LEFT JOIN tmp_lingkup_ruang as lr ON lr.id_lingkup_ruang =
ruang.id_lingkup_ruang").
```

```
        Joins("LEFT JOIN tmp_jenis_ruang as jr ON jr.id_jenis_ruang =
ruang.id_jenis_ruang").
        Joins("LEFT JOIN tmp_pemanfaatan as pm ON pm.id_pemanfaatan
= ruang.id_pemanfaatan").
        Joins("LEFT JOIN tmp_unit_organisasi as uo ON uo.id_unit_org =
ruang.id_unit_org").
        Joins("LEFT JOIN tmp_kampus as kp ON kp.id_kampus =
ruang.id_kampus").
        Joins("LEFT JOIN pengelola_ruang as pr ON pr.id_ruang =
ruang.id_ruang").
        Joins("LEFT JOIN sdm ON sdm.id_sdm = pr.id_sdm").
        Order("(SELECT NULL)").
        Offset(5 * (currentPage - 1)).
        Limit(5)
    var datas []queries.Ruang
    if err := query.Find(&datas).Error; err != nil {
        if err == gorm.ErrRecordNotFound {
            return []queries.Ruang{}, queries.ErrTidakAdaDaftarRuang
        }
        return []queries.Ruang{}, err
    }
    output := make([]queries.Ruang, 0)

    for _, data := range datas {
        data.IdRuang = utils.ConvertId(data.IdRuang)
        // Dokumen
        query_dokumen := q.db.Table("dokumen").
            Select(" dokumen.id_dokumen as id_dokumen, nama_dokumen,
ukuran, mime, public_link").
            Joins("LEFT JOIN dokumen_ruang ON
dokumen_ruang.id_dokumen = dokumen.id_dokumen").
            Where("dokumen_ruang.id_ruang = ?", data.IdRuang)
        var datas_dokumen []queries.Dokumen
```

```
    if err := query_dokumen.Find(&datas_dokumen).Error; err != nil {
        if err == gorm.ErrRecordNotFound {
            return []queries.Ruang{}, queries.ErrTidakAdaDaftarRuang
        }
        return []queries.Ruang{}, err
    }
    outputDokumen := make([]queries.Dokumen, 0)
    for _, data := range datas_dokumen {
        data.IdDokumen = utils.ConvertId(data.IdDokumen)
        outputDokumen = append(outputDokumen, queries.Dokumen{
            IdDokumen:   data.IdDokumen,
            NamaDokumen: data.NamaDokumen,
            Ukuran:      data.Ukuran,
            Mime:        data.Mime,
            PublicLink:  data.PublicLink,
        })
    }
    output = append(output, queries.Ruang{
        IdRuang:          data.IdRuang,
        LingkupRuang:     data.LingkupRuang,
        JenisRuang:       data.JenisRuang,
        LokasiKampus:     data.LokasiKampus,
        JenisPemanfaatan: data.JenisPemanfaatan,
        Nama:             data.Nama,
        Kode:             data.Kode,
        Kapasitas:        data.Kapasitas,
        UnitOrganisasi:   data.UnitOrganisasi,
        Jenis:            data.Jenis,
        PengelolaRuang:   data.PengelolaRuang,
        Dokumen:          outputDokumen,
    })
}
return output, nil
```

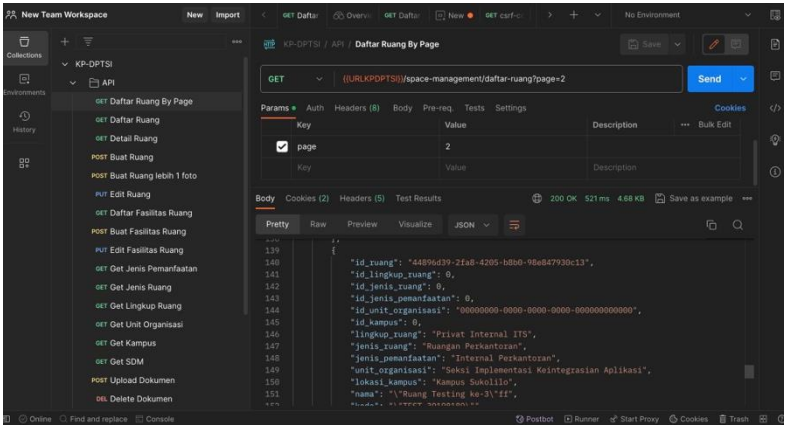
```
}
```

Code 5.1 Query GET Daftar Ruang By Page

```
func (c *DaftarRuangController) GetDaftarRuangByPage(ctx
*gin.Context) {
    newCurrentPage, err := strconv.Atoi(ctx.Query("page"))
    if err != nil {
        common.AbortAndResponseErrorWithJSON(ctx, err)
        return
    }
    DaftarRuang, err := c.daftarRuangQuery.GetDaftarRuangByPage(ctx,
newCurrentPage)
    if err != nil {
        common.AbortAndResponseErrorWithJSON(ctx, err)
        return
    }
    scheme := "http"
    if ctx.Request.TLS != nil {
        scheme = "https"
    }

    Next := newCurrentPage + 1
    var total int
    if newCurrentPage <= 1 {
        total = len(DaftarRuang)
    } else {
        total = len(DaftarRuang) + (5 * (newCurrentPage - 1))
    }
    ctx.JSON(http.StatusOK, gin.H{
        "code":    http.StatusOK,
        "message": "success get ",
        "links": gin.H{
```

```
            "next": fmt.Sprintf("%s://%s/space-management/daftar-
ruang/?page=%d", scheme, ctx.Request.Host, Next),
        },
        "meta": gin.H{
            "total": total,
            "range": len(DaftarRuang),
            "page":  newCurrentPage,
        },
        "data": DaftarRuang,
    })
}
```

Code 5.2 Controller GET Daftar Ruang By Page



Gambar 5.5: Postman GET Daftar Ruang By Page

5.2.1.2 POST Daftar Ruang

Implementasi POST daftar ruang melibatkan query menggunakan framework GORM dan juga GIN untuk controller dan

routenya yang mana digunakan untuk membuat ruangan baru. Berikut merupakan implementasinya.

```
func (q *DaftarRuangQuery) CreateNewRuangs(ctx context.Context,
ruang queries.CreateRuang, fasilitasRuang
queries.CreateFasilitasRuang, sdm queries.CreateSdm) (string, error) {
    newUUIDRuang := uuid.New()
    newUUIDFasilitasBarang := uuid.New()
    if err := q.db.Transaction(func(tx *gorm.DB) error {
        ruang.IDRuang = newUUIDRuang
        createRuang := tx.Table(RuangTableName).Create(&ruang)
        if createRuang.Error != nil {
            return createRuang.Error
        }
        fasilitasRuang.IDBarang = newUUIDFasilitasBarang
        createFasilitasRuang :=
tx.Table(FasilitasRuangTableName).Create(&queries.CreateFasilitasRua
ng{
            IDBarang:   fasilitasRuang.IDBarang,
            IDRuang:    newUUIDRuang,
            NamaBarang: fasilitasRuang.NamaBarang,
            Jumlah:     fasilitasRuang.Jumlah,
            Keterangan: fasilitasRuang.Keterangan,
        })
        if createFasilitasRuang.Error != nil {
            return createFasilitasRuang.Error
        }

        // Create SDM
        newUUIDSDM := uuid.New()
        newUUIDUser := uuid.New()
        createSDM := tx.Table("sdm").Create(&queries.CreateSdm{
            IdSDM:     newUUIDSDM,
```

```
            IdUnitOrg: ruang.IDUnitOrg,
            IdUser:    newUUIDUser,
            NamaSDM:   sdm.NamaSDM,
            NIP:       sdm.NIP,
        })
        if createSDM.Error != nil {
            return createSDM.Error
        }
        createPengelolaRuang :=
tx.Table("pengelola_ruang").Create(&queries.CreatePengelolaRuang{
            IdSDM:   newUUIDSDM,
            IdRuang: newUUIDRuang,
        })
        if createPengelolaRuang.Error != nil {
            return createPengelolaRuang.Error
        }
        return nil
    }); err != nil {
        return "Error", err
    }
    return newUUIDRuang.String(), nil
}
```

Code 5.3: Query POST Daftar Ruang

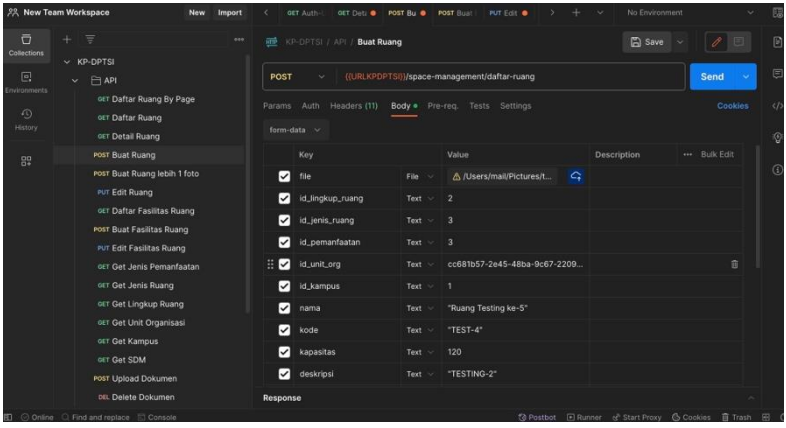
```
func (c *DaftarRuangController) CreateNewRuangs(ctx *gin.Context) {
    var requestBodyRuang queries.CreateRuang
    var requestBodyFasilitasRuang queries.CreateFasilitasRuang
    var requestBodySDM queries.CreateSdm

    config := storageapi.Config{
        ClientID:        os.Getenv("OIDC_CLIENT_ID"),
        ClientSecret:    os.Getenv("OIDC_CLIENT_SECRET"),
        OidcProviderURL: os.Getenv("OIDC_PROVIDER"),
```

```
        StorageApiURL:   os.Getenv("STORAGE_API_URL"),
    }

    storageApi, err := storageapi.NewStorageApi(ctx, config)
    if err != nil {
        ctx.AbortWithStatusJSON(http.StatusBadRequest, gin.H{
            "code":    http.StatusBadRequest,
            "message": "Gagal pada Inisialisasi StorageAPI",
        })
        return
    }
    form, err := ctx.MultipartForm()
    if err != nil {
        ctx.AbortWithStatusJSON(http.StatusBadRequest, gin.H{
            "code":    http.StatusBadRequest,
            "message": "Gagal pada Inisialissi MultiPartForm",
        })
        return
    }
    files := form.File["files[]"]

    var uploadResponses []storageapi.UploadResponse

    // Save each file to the defined path and generate a unique identifier
for each file
    for _, file := range files {
        uploadResponse, err := storageApi.Upload(ctx, file)

        if err != nil {
            ctx.AbortWithStatusJSON(http.StatusBadRequest, gin.H{
                "code":    http.StatusBadRequest,
                "message": "Gagal pada Upload File",
            })
```

```
            return
        }
        uploadResponses = append(uploadResponses, uploadResponse)
    }
    //Bind JSON
    if err := ctx.ShouldBind(&requestBodyRuang); err != nil {
        ctx.AbortWithStatusJSON(http.StatusBadRequest, gin.H{
            "code":    http.StatusBadRequest,
            "message": "Gagal pada Bind Body Ruang",
        })
        return
    }
    if err := ctx.ShouldBind(&requestBodyFasilitasRuang); err != nil {
        ctx.AbortWithStatusJSON(http.StatusBadRequest, gin.H{
            "code":    http.StatusBadRequest,
            "message": "Gagal pada Bind Body Fasilitas Ruang",
        })
        return
    }
    if err := ctx.ShouldBind(&requestBodySDM); err != nil {
        ctx.AbortWithStatusJSON(http.StatusBadRequest, gin.H{
            "code":    http.StatusBadRequest,
            "message": "Gagal pada Bind Body SDM",
        })
        return
    }
    // Panggil fungsi CreateNewRuang dari RuangQuery
    idRuang, err := c.daftarRuangQuery.CreateNewRuangs(ctx,
requestBodyRuang, requestBodyFasilitasRuang, requestBodySDM);
    if err != nil {
        ctx.AbortWithStatusJSON(http.StatusBadRequest, gin.H{
            "code":    http.StatusBadRequest,
            "message": "Gagal pada Create Ruangan",
```

```
        })
        return
    }
    if err := c.dokumenQuery.UploadDokumen(ctx, idRuang,
uploadResponses); err != nil {
        ctx.AbortWithStatusJSON(http.StatusBadRequest, gin.H{
            "code":    http.StatusBadRequest,
            "message": "Gagal pada Create Ruangan",
        })
        return
    }
    // Return a success message and the file metadata
    ctx.JSON(http.StatusCreated, gin.H{
        "code":    http.StatusCreated,
        "message": "File uploaded successfully",
        "files":   "Ruang baru telah berhasil dibuat",
    })
}
```

Code 5.4 Controller POST Daftar Ruang



Gambar 5.6 Postman POST Daftar Ruang

5.2.1.3 PUT Daftar Ruang

Implementasi PUT daftar ruang melibatkan query menggunakan framework GORM dan juga GIN untuk controller dan routenya yang mana digunakan untuk mengubah value dari ruangan yang telah ada sebelumnya berdasarkan id ruangan tertentu. Berikut merupakan implementasinya.

```
func (q *DaftarRuangQuery) UpdateRuang(ctx context.Context, ruang
queries.UpdateRuang, idRuang string, pengelolaRuang
queries.UpdatePengelolaRuang) error {
    if err := q.db.Transaction(func(tx *gorm.DB) error {
        updateRuang := tx.Table(RuangTableName).Where("id_ruang = ?",
idRuang).Updates(&ruang)
        if updateRuang.Error != nil {
            return updateRuang.Error
        }
        var datas []queries.CreatePengelolaRuang
        query := tx.Table("pengelola_ruang").Where("id_ruang = ?",
idRuang).Find(&datas)
        if query != nil {
            updatePengelolaRuang :=
tx.Table("pengelola_ruang").Where("id_ruang = ?",
idRuang).Updates(&pengelolaRuang)
            if updatePengelolaRuang.Error != nil {
                return updatePengelolaRuang.Error
            }
        } else {
            return ctx.Err()
        }
        return nil
    }); err != nil {
```

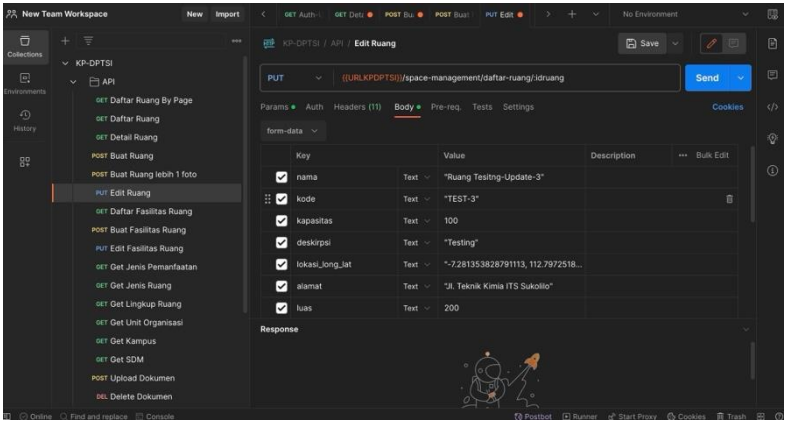
```
        return err
    }
    return nil
}
```

Code 5.5 Query PUT Daftar Ruang

```
func (c *DaftarRuangController) UpdateRuang(ctx *gin.Context) {
    var requestBodyRuang queries.UpdateRuang
    var requestBodyPengelolaRuang queries.UpdatePengelolaRuang
    idRuang := ctx.Param("idruang")
    //Bind JSON
    if err := ctx.ShouldBind(&requestBodyRuang); err != nil {
        ctx.AbortWithStatusJSON(http.StatusBadRequest, gin.H{
            "code":    http.StatusBadRequest,
            "message": "Gagal pada Bind Body Ruang",
        })
        return
    }
    if err := ctx.ShouldBind(&requestBodyPengelolaRuang); err != nil {
        ctx.AbortWithStatusJSON(http.StatusBadRequest, gin.H{
            "code":    http.StatusBadRequest,
            "message": "Gagal pada Bind Pengelola Ruang",
        })
        return
    }
    if err := c.daftarRuangQuery.UpdateRuang(ctx, requestBodyRuang,
idRuang, requestBodyPengelolaRuang); err != nil {
        ctx.AbortWithStatusJSON(http.StatusBadRequest, gin.H{
            "code":    http.StatusBadRequest,
            "message": "Gagal pada Update Ruang",
        })
        return
    }
```

```
    ctx.JSON(http.StatusAccepted,
        gin.H{
            "code":    http.StatusAccepted,
            "message": fmt.Sprintf("Ruang dengan id %s berhasil dirubah",
idRuang),
        })
}
```

Code 5.6 Controller PUT Daftar Ruang



Gambar 5.7 Postman PUT Daftar Ruang

5.2.2 **Implementasi API Detail Ruang**

Implementasi API detail ruang berfokus pada pengambilan data pada beberapa tabel di antaranya adalah ruang, sdm, dan pengelola ruang di dalam database.

5.2.2.1 GET Detail Ruang

Implementasi GET detail ruang melibatkan query menggunakan framework GORM dan juga GIN untuk controller dan

routenya yang mana digunakan untuk mendapatkan data dari hasil Join beberapa tabel yang digunakan untuk memperlihatkan detail ruangan dengan id tertentu. Berikut merupakan implementasinya.

```
func (q *DetailRuangQuery) GetDetailRuangById(ctx context.Context,
idruang string) (queries.Ruang, error) {
    query := q.db.
        Table(RuangTableName).
        Select("ruang.id_ruang, ruang.id_lingkup_ruang,
ruang.id_jenis_ruang, ruang.id_pemanfaatan as id_jenis_pemanfaatan,
ruang.id_kampus, ruang.id_unit_org as
id_unit_organisasi,lr.lingkup_ruang, jr.jenis_ruang,
pm.jenis_pemanfaatan, kp.kampus as lokasi_kampus,ruang.nama as
nama, kode, kapasitas, jr.jenis_ruang as jenis_ruang, uo.nama as
unit_organisasi, ruang.lokasi_long_lat as lokasi, luas,
ruang.dimensi_ruang as dimensi, deskripsi, alamat").
        Where("ruang.id_ruang = ?", idruang).
        Joins("LEFT JOIN tmp_lingkup_ruang as lr ON lr.id_lingkup_ruang =
ruang.id_lingkup_ruang").
        Joins("LEFT JOIN tmp_jenis_ruang as jr ON jr.id_jenis_ruang =
ruang.id_jenis_ruang").
        Joins("LEFT JOIN tmp_pemanfaatan as pm ON pm.id_pemanfaatan
= ruang.id_pemanfaatan").
        Joins("LEFT JOIN tmp_unit_organisasi as uo ON uo.id_unit_org =
ruang.id_unit_org").
        Joins("LEFT JOIN tmp_kampus as kp ON kp.id_kampus =
ruang.id_kampus")
        // Joins("LEFT JOIN pengelola_ruang as pr ON pr.id_ruang =
ruang.id_ruang").
        // Joins("LEFT JOIN sdm ON sdm.id_sdm = pr.id_sdm")

    var data queries.Ruang
```

```
    if err := query.Find(&data).Error; err != nil {
        if err == gorm.ErrRecordNotFound {
            return queries.Ruang{}, queries.ErrTidakAdaDetailRuang
        }
        return queries.Ruang{}, err

    }
    data.IdRuang = utils.ConvertId(data.IdRuang)
    // Get Dokumen
    query_dokumen := q.db.Table("dokumen").
        Select("dokumen.id_dokumen,nama_dokumen, ukuran, mime,
public_link").
        Joins("LEFT JOIN dokumen_ruang ON
dokumen_ruang.id_dokumen = dokumen.id_dokumen").
        Where("dokumen_ruang.id_ruang = ?", data.IdRuang)
    var datas_dokumen []queries.Dokumen
    if err := query_dokumen.Find(&datas_dokumen).Error; err != nil {
        if err == gorm.ErrRecordNotFound {
            return queries.Ruang{}, queries.ErrTidakAdaDaftarRuang
        }
        return queries.Ruang{}, err
    }
    outputDokumen := make([]queries.Dokumen, 0)
    for _, data := range datas_dokumen {
        data.IdDokumen = utils.ConvertId(data.IdDokumen)
        outputDokumen = append(outputDokumen, queries.Dokumen{
            IdDokumen:   data.IdDokumen,
            NamaDokumen: data.NamaDokumen,
            Ukuran:     data.Ukuran,
            Mime:       data.Mime,
            PublicLink:  data.PublicLink,
        })
    }
```

```
    // Get PengelolaRuang
    query_pr := q.db.Table("pengelola_ruang").
        Select("pengelola_ruang.id_sdm, sdm.nama as nama,
sdm.id_unit_org, sdm.id_user, sdm.npp_nip").
        Where("pengelola_ruang.id_ruang = ?", data.IdRuang).
        Joins("LEFT JOIN sdm ON sdm.id_sdm =
pengelola_ruang.id_sdm")
    var datas_pr []queries.Sdm
    if err := query_pr.Find(&datas_pr).Error; err != nil {
        if err == gorm.ErrRecordNotFound {
            return queries.Ruang{}, queries.ErrTidakAdaDaftarRuang
        }
        return queries.Ruang{}, err
    }
    outputPR := make([]queries.Sdm, 0)
    for _, data := range datas_pr {
        data.IdSDM = utils.ConvertId(data.IdSDM)
        data.IdUnitOrg = utils.ConvertId(data.IdUnitOrg)
        data.IdUser = utils.ConvertId(data.IdUser)
        outputPR = append(outputPR, queries.Sdm{
            IdSDM:     data.IdSDM,
            IdUnitOrg: data.IdUnitOrg,
            IdUser:    data.IdUser,
            NamaSDM:   data.NamaSDM,
            NIP:       data.NIP,
        })
    }
    output := queries.Ruang{
        IdRuang:            data.IdRuang,
        IdLingkupRuang:     data.IdLingkupRuang,
        IdJenisRuang:       data.IdJenisRuang,
        IdJenisPemanfaatan: data.IdJenisPemanfaatan,
        IdUnitOrganisasi:   data.IdUnitOrganisasi,
```

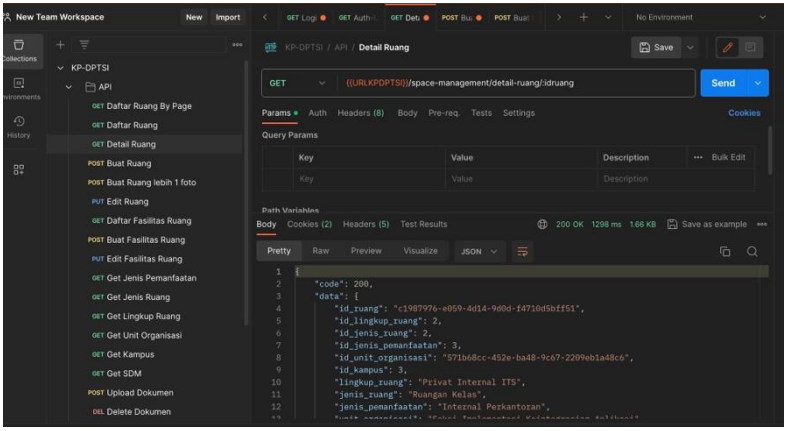
```
        IdKampus:           data.IdKampus,
        LingkupRuang:       data.LingkupRuang,
        JenisRuang:         data.JenisRuang,
        LokasiKampus:       data.LokasiKampus,
        JenisPemanfaatan:   data.JenisPemanfaatan,
        Nama:               data.Nama,
        Kode:               data.Kode,
        Kapasitas:          data.Kapasitas,
        UnitOrganisasi:     data.UnitOrganisasi,
        Jenis:              data.Jenis,
        Lokasi:             data.Lokasi,
        Luas:               data.Luas,
        Dimensi:            data.Dimensi,
        Deskripsi:          data.Deskripsi,
        Alamat:             data.Alamat,
        Dokumen:            outputDokumen,
        PengelolaRuang:     outputPR,
    }
    return output, nil
}
```

Code 5.7 Query GET Detail Ruang

```
func (dr *DetailRuangController) GetDetailRuangById(ctx *gin.Context) {
    idRuang := ctx.Param("idruang")
    DetailRuang, err := dr.detailRuangQuery.GetDetailRuangById(ctx,
idRuang)
    if err != nil {
        common.AbortAndResponseErrorWithJSON(ctx, err)
        return
    }
    ctx.JSON(http.StatusOK, gin.H{
        "code":    http.StatusOK,
        "message": "success",
```

```
        "data": DetailRuang,
    })
}
```

Code 5.8 Controller GET Detail Ruang



Gambar 5.8 Postman GET Detail Ruang

5.2.3 **Implementasi API Fasilitas Ruang**

Implementasi API fasilitas ruang berfokus pada pengambilan dan pengolahan data pada tabel fasilitas ruang di dalam database.

5.2.3.1 GET Fasilitas Ruang

Implementasi GET fasilitas ruang berdasarkan halaman melibatkan query menggunakan framework GORM dan juga GIN untuk controller dan routenya yang mana digunakan untuk mendapatkan list fasilitas ruang berdasarkan halaman tertentu. Berikut merupakan implementasinya.

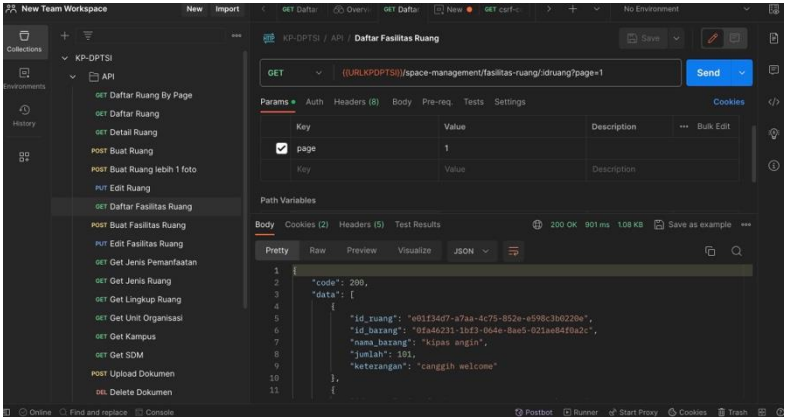
```
func (q *FasilitasRuangQuery) GetFasilitasRuangByIdRuang(ctx
context.Context, idRuang string, currentPage int)
([]queries.FasilitasRuang, error) {
    query :=
q.db.Table(FasilitasRuangTableName).Select("*").Where("id_ruang = ?",
idRuang).
    Order("(SELECT NULL)").
    Offset(5 * (currentPage - 1)).
    Limit(5)

    var datas []queries.FasilitasRuang
    if err := query.Find(&datas).Error; err != nil {
        if err == gorm.ErrRecordNotFound {
            return []queries.FasilitasRuang{},
queries.ErrTidakAdaFasilitasRuang
        }
        return []queries.FasilitasRuang{}, err

    }
    output := make([]queries.FasilitasRuang, 0)
    for _, data := range datas {
        data.IdRuang = utils.ConvertId(data.IdRuang)
        output = append(output, queries.FasilitasRuang{
            IdRuang:    data.IdRuang,
            IdFasilitasRuang: data.IdFasilitasRuang,
            NamaBarang: data.NamaBarang,
            Jumlah:     data.Jumlah,
            Keterangan: data.Keterangan,
        })
    }
    return output, nil
}
```

Code 5.9 Query GET Fasilitas Ruang

```
func (c *FasilitasRuangController) GetFasilitasRuangByIdRuang(ctx
*gin.Context) {
    newCurrentPage, err := strconv.Atoi(ctx.Query("page"))
    if err != nil {
        common.AbortAndResponseErrorWithJSON(ctx, err)
        return
    }
    idRuang := ctx.Param("idruang")
    FasilitasRuang, err :=
c.fasilitasRuangQuery.GetFasilitasRuangByIdRuang(ctx, idRuang,
newCurrentPage)
    if err != nil {
        common.AbortAndResponseErrorWithJSON(ctx, err)
        return
    }
    scheme := "http"
    if ctx.Request.TLS != nil {
        scheme = "https"
    }
    Next := newCurrentPage + 1

    var total int
    if newCurrentPage <= 1 {
        total = len(FasilitasRuang)
    } else{
        total = len(FasilitasRuang) + (5 * (newCurrentPage - 1) )
    }

    ctx.JSON(http.StatusOK, gin.H{
        "code":    http.StatusOK,
        "message": "success",
        "links": gin.H{
```

```
            "next": fmt.Sprintf("%s://%s/space-management/fasilitas-
ruang/%s?page=%d", scheme, ctx.Request.Host, idRuang, Next),
        },
        "meta": gin.H{
            "total": total ,
            "range": len(FasilitasRuang),
            "page":  newCurrentPage,
        },
        "data": FasilitasRuang,
    })
}
```

Code 5.10 Controller GET Fasilitas Ruang



Gambar 5.9 Postman GET Fasilitas Ruang

5.2.3.2 POST Fasilitas Ruang

Implementasi POST fasilitas ruang melibatkan query menggunakan framework GORM dan juga GIN untuk controller dan routenya yang mana digunakan untuk membuat

fasilitas ruangan baru. Berikut merupakan implementasinya.

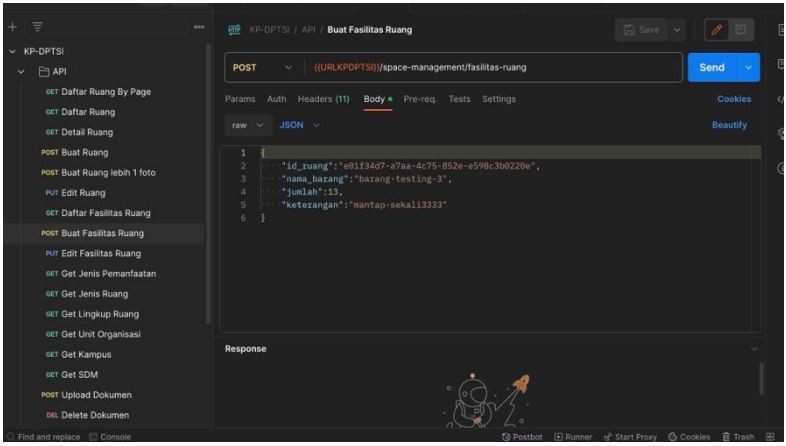
```
func (q *FasilitasRuangQuery) CreateFasilitasRuang(ctx context.Context,
fasilitasRuang queries.CreateFasilitasRuang) error {
    newUUID := uuid.New()
    fasilitasRuang.IDBarang = newUUID

    result :=
q.db.Table(FasilitasRuangTableName).Create(&fasilitasRuang)
    if result.Error != nil {
        return result.Error
    }
    return nil
}
```

Code 5.11 Query POST Fasilitas Ruang

```
func (c *FasilitasRuangController) CreateFasilitasRuang(ctx
*gin.Context) {
    var requestBody queries.CreateFasilitasRuang
    if err := ctx.ShouldBindJSON(&requestBody); err != nil {
        common.AbortAndResponseErrorWithJSON(ctx, err)
        return
    }

    // Panggil fungsi CreateNewRuang dari RuangQuery
    if err := c.fasilitasRuangQuery.CreateFasilitasRuang(ctx,
requestBody); err != nil {
        common.AbortAndResponseErrorWithJSON(ctx, err)
        return
    }

    ctx.JSON(http.StatusCreated,
```

```
        gin.H{
            "code":    http.StatusOK,
            "message": "Fasilitas Ruang baru telah berhasil dibuat",
        })
}
```

Code 5.12 Controller POST Fasilitas Ruang



Gambar 5.10 Postman POST Fasilitas Ruang

5.2.3.3 PUT Fasilitas Ruang

Implementasi PUT fasilitas ruang melibatkan query menggunakan framework GORM dan juga GIN untuk controller dan routenya yang mana digunakan untuk mengubah value dari fasilitas ruangan yang telah ada sebelumnya berdasarkan id ruangan tertentu. Berikut merupakan implementasinya.

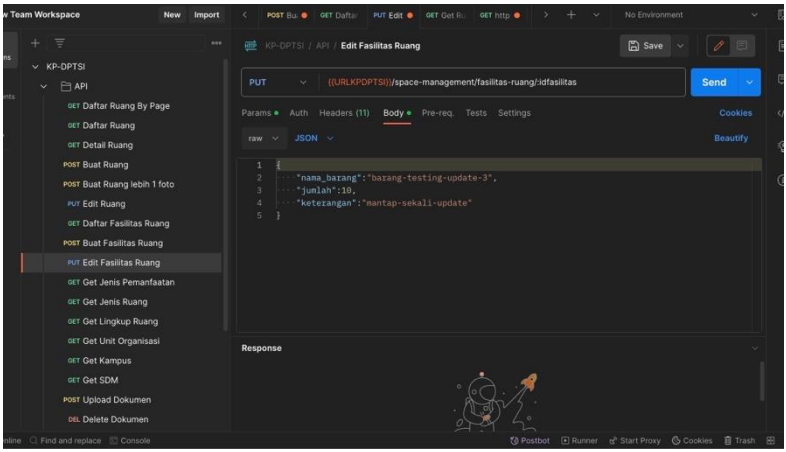
```
func (q *FasilitasRuangQuery) UpdateFasilitasRuang(ctx
context.Context, fasilitasRuang queries.UpdateFasilitasRuang,
idFasilitas string) error {
```

```
    result := q.db.Table(FasilitasRuangTableName).Where("id_barang =
?", idFasilitas).Updates(&fasilitasRuang)
    if result.Error != nil {
        return result.Error
    }
    return nil
}
```

Code 5.13 Query PUT Fasilitas Ruang

```
func (c *FasilitasRuangController) UpdateFasilitasRuang(ctx
*gin.Context) {
    var requestBody queries.UpdateFasilitasRuang
    idFasilitas := ctx.Param("idfasilitas")
    if err := ctx.ShouldBindJSON(&requestBody); err != nil {
        common.AbortAndResponseErrorWithJSON(ctx, err)
        return
    }

    if err := c.fasilitasRuangQuery.UpdateFasilitasRuang(ctx,
requestBody, idFasilitas); err != nil {
        common.AbortAndResponseErrorWithJSON(ctx, err)
        return
    }

    ctx.JSON(http.StatusCreated,
        gin.H{
            "code":    http.StatusOK,
            "message": fmt.Sprintf("Fasilitas dengan id %s berhasil dirubah",
idFasilitas),
        })
}
```

Code 5.14 Controller PUT Fasilitas Ruang



Gambar 5.11 Postman PUT Fasilitas Ruang

5.2.4 **Implementasi API File**

Implementasi API file berfokus pada pengolahan data pada tabel dokumen di dalam database.

5.2.4.1 POST File

Implementasi POST file melibatkan query menggunakan framework GORM dan juga GIN untuk controller dan routenya yang mana digunakan untuk membuat dokumen baru. Berikut merupakan implementasinya.

```
func (q *DokumenQuery) UploadDokumen(ctx context.Context, idRuang
string, uploadResponses []storageapi.UploadResponse) error {
    if err := q.db.Transaction(func(tx *gorm.DB) error {
        dokumens := make([]queries.CreateDokumen, 0)
        dokumenRuang := make([]queries.CreateDokumenRuang, 0)
```

```
        for _, uploadResponse := range uploadResponses {
            newUUIDDokumen := uuid.New()
            dokumens = append(dokumens, queries.CreateDokumen{
                IdDokumen:   newUUIDDokumen,
                NamaDokumen: uploadResponse.Info.FileName,
                NamaFile:    uploadResponse.Info.FileName,
                Mime:        uploadResponse.Info.FileMimetype,
                Ekstensi:    uploadResponse.Info.FileExt,
                Keterangan:  uploadResponse.Message,
                Ukuran:      uploadResponse.Info.FileSize,
                FileId:      uploadResponse.FileID,
                PublicLink:  uploadResponse.Info.PublicLink,
            })
            dokumenRuang = append(dokumenRuang,
queries.CreateDokumenRuang{
                IdDokumen: newUUIDDokumen,
                IdRuang:   uuid.MustParse(idRuang),
            })
            createDokumen := tx.Table("dokumen").Create(&dokumens)
            if createDokumen.Error != nil {
                return createDokumen.Error
            }
            createDokumenRuang :=
tx.Table("dokumen_ruang").Create(&dokumenRuang)
            if createDokumenRuang.Error != nil {
                return createDokumenRuang.Error
            }
        }
        return nil
    }); err != nil {
        return err
    }
```

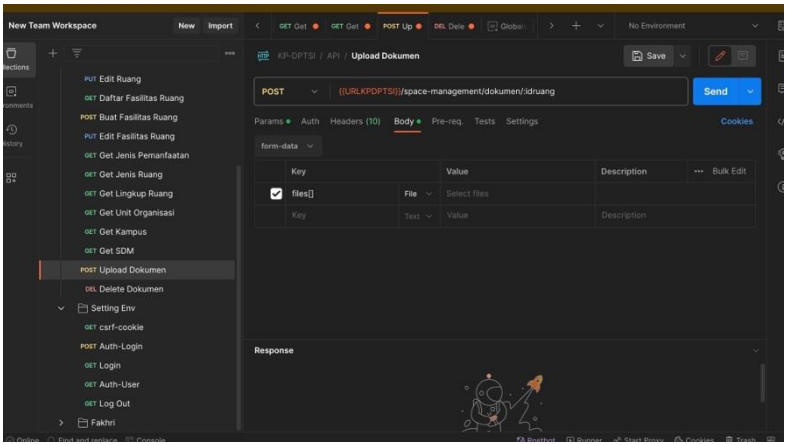
```
    return nil
}
```

Code 5.15 Query POST File

```
func (c *DokumenController) UploadDokumenController(ctx *gin.Context)
{
    idRuang := ctx.Param("idruang")
    config := storageapi.Config{
        ClientID:        os.Getenv("OIDC_CLIENT_ID"),
        ClientSecret:    os.Getenv("OIDC_CLIENT_SECRET"),
        OidcProviderURL: os.Getenv("OIDC_PROVIDER"),
        StorageApiURL:   os.Getenv("STORAGE_API_URL"),
    }

    storageApi, err := storageapi.NewStorageApi(ctx, config)
    if err != nil {
        ctx.AbortWithStatusJSON(http.StatusBadRequest, gin.H{
            "code":    http.StatusBadRequest,
            "message": "Gagal pada Inisialisasi StorageAPI",
        })
        return
    }
    form, err := ctx.MultipartForm()
    if err != nil {
        ctx.AbortWithStatusJSON(http.StatusBadRequest, gin.H{
            "code":    http.StatusBadRequest,
            "message": "Gagal pada Inisialissi MultiPartForm",
        })
        return
    }
    files := form.File["files[]"]

    var uploadResponses []storageapi.UploadResponse
```

```
    // Save each file to the defined path and generate a unique identifier
for each file
    for _, file := range files {
        uploadResponse, err := storageApi.Upload(ctx, file)

        if err != nil {
            ctx.AbortWithStatusJSON(http.StatusBadRequest, gin.H{
                "code":    http.StatusBadRequest,
                "message": "Gagal pada Upload File",
            })
            return
        }
        uploadResponses = append(uploadResponses, uploadResponse)
    }
    if err := c.dokumenQuery.UploadDokumen(ctx, idRuang,
uploadResponses); err != nil {
        ctx.AbortWithStatusJSON(http.StatusBadRequest, gin.H{
            "code":    http.StatusBadRequest,
            "message": "Gagal pada Upload File",
        })
        return
    }
    // Return a success message and the file metadata
    ctx.JSON(http.StatusCreated, gin.H{
        "code":    http.StatusCreated,
        "message": "File uploaded successfully",
        "files": uploadResponses  ,
    })
}
```

Code 5.16 Controller POST File



Gambar 5.12 Postman POST File

5.2.4.2 DELETE File

Implementasi DELETE file melibatkan query menggunakan framework GORM dan juga GIN untuk controller dan routenya yang mana digunakan untuk menghapus dokumen berdasarkan id tertentu. Berikut merupakan implementasinya.

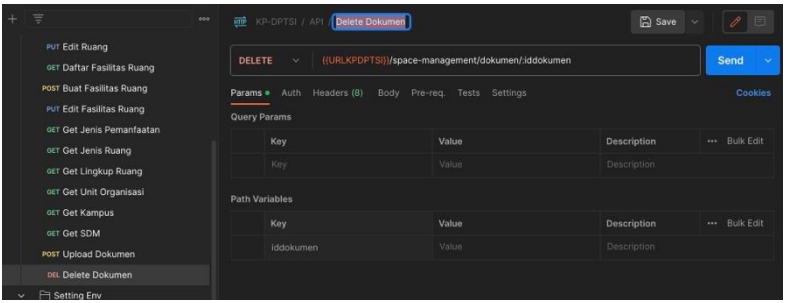
```
func (q *DokumenQuery) DeleteDokumen(ctx context.Context,
idDokumen string) error {
    if err := q.db.Transaction(func(tx *gorm.DB) error {
        deleteDokumenRuang :=
tx.Table("dokumen_ruang").Where("id_dokumen = ?",
idDokumen).Delete(&queries.CreateDokumenRuang{})
        if deleteDokumenRuang.Error != nil {
            return deleteDokumenRuang.Error
        }
        deleteDokumen := tx.Table("dokumen").Where("id_dokumen = ?",
idDokumen).Delete(&queries.CreateDokumen{})
        if deleteDokumen.Error != nil {
```

```
            return deleteDokumen.Error
        }
        return nil
    }); err != nil {
        return err
    }
    return nil
}
```

Code 5.17 Query DELETE File

```
func (c *DokumenController) DeleteDokumenController(ctx *gin.Context)
{
    idDokumen := ctx.Param("iddokumen")
    if err := c.dokumenQuery.DeleteDokumen(ctx, idDokumen); err != nil {
        ctx.AbortWithStatusJSON(http.StatusBadRequest, gin.H{
            "code":    http.StatusBadRequest,
            "message": "Gagal pada Update Ruang",
        })
        return
    }
    ctx.JSON(http.StatusAccepted,
        gin.H{
            "code":    http.StatusAccepted,
            "message": fmt.Sprintf("Ruang dengan id %s berhasil dirubah",
idDokumen),
        })
}
```

Code 5.18 Controller DELETE File



Gambar 5.13 Postman DELETE Fasilitas Ruang

[Halaman ini sengaja dikosongkan]

BAB VI
PENGUJIAN DAN EVALUASI

Bab ini menjelaskan tahap uji coba terhadap Aplikasi MyITS Space Management. Pengujian dilakukan untuk memastikan fungsionalitas dan kesesuaian hasil implementasi arsitektur dengan analisis dan perancangan arsitektur.

6.1. Tujuan Pengujian

Tujuan dari pengujian sistem ini adalah untuk memastikan bahwa fungsi-fungsi utama dari Modul MyITS Space Management berjalan sesuai dengan spesifikasi yang telah ditetapkan. Pengujian dilakukan untuk memverifikasi keamanan, kinerja, dan keandalan sistem. Adapun tujuan pengujian meliputi:

1. Verifikasi Fungsionalitas:
    - Memastikan daftar ruang dapat ditampilkan dengan benar.
    - Memverifikasi kemampuan menambah, mengubah, dan menghapus ruang.
    - Meyakinkan bahwa detail ruang dan fasilitasnya dapat diakses dengan benar.
2. Pengujian Kesalahan:
    - Memverifikasi bahwa pesan kesalahan memberikan informasi yang jelas dan bermanfaat.

6.2. Kriteria Pengujian

Kriteria pengujian mengacu pada standar dan spesifikasi yang telah ditetapkan sebelumnya. Kriteria utama yang diuji mencakup:

1. Fungsionalitas:
    - Semua fungsi CRUD (Create, Read, Update, Delete) berjalan dengan benar.
    - Antarmuka pengguna memberikan pengalaman pengguna yang baik.
2. Kesalahan:
    - Sistem dapat menangani kesalahan dengan baik dan memberikan pesan kesalahan yang informatif.

6.3. Skenario Pengujian

Skenario pengujian dilakukan dengan melakukan peran sebagai user yang akan menjalankan fitur-fitur. Langkah-langkah untuk setiap kebutuhan yaitu sebagai berikut.

1. Pengujian Fungsionalitas:
    - Menambahkan ruang baru ke dalam sistem.
    - Mengubah informasi ruang yang sudah ada.
    - Melihat detail ruang dan fasilitasnya.
2. Pengujian Kesalahan:
    - Menjalankan skenario dengan input yang salah dan memeriksa respons sistem terhadap kesalahan validasi.

6.4. Evaluasi Pengujian

Hasil pengujian dilakukan terhadap pengamatan mengenai perilaku sistem aplikasi MyITS Space Management terhadap kasus skenario uji coba. Tabel 6.1 di bawah ini menjelaskan hasil uji coba terhadap aplikasi yang telah dibuat.

Tabel 6.1. Hasil Evaluasi Pengujian

Kriteria Pengujian	**Hasil Pengujian**
Menambahkan ruang baru ke dalam sistem.	Terpenuhi
Mengubah informasi ruang yang sudah ada.	Terpenuhi
Melihat detail ruang dan fasilitasnya.	Terpenuhi
Menjalankan skenario dengan input yang salah dan memeriksa respons sistem terhadap kesalahan validasi.	Terpenuhi

[Halaman ini sengaja dikosongkan]

BAB VII
KESIMPULAN DAN SARAN

7.1. Kesimpulan

Kesimpulan yang didapat setelah melakukan perancangan arsitektur sistem aplikasi MyITS Space Management pada kegiatan kerja praktek di Direktorat Pengembangan Sistem dan Informasi adalah sebagai berikut :

a. Modul MyITS Space Management mampu memberikan fungsionalitas optimal sesuai dengan kebutuhan pengguna, seperti menambah, mengubah, dan menghapus data ruang dengan mudah.
b. Antarmuka pengguna dirancang secara intuitif, memudahkan pengguna dalam berinteraksi dengan sistem dan mengakses informasi mengenai ruang dan fasilitasnya.

7.2. Saran

Saran untuk perancangan arsitektur sistem aplikasi MyITS Space Management adalah sebagai berikut :

a. Menambahkan fitur-fitur tambahan yang dapat memperkaya pengalaman pengguna, seperti pencarian yang lebih canggih, integrasi dengan sistem lain, atau pembaruan berkala data ruangan.
b. Pertimbangkan pengembangan modul atau fitur lain yang dapat melengkapi fungsionalitas dan memberikan solusi

terintegrasi untuk kebutuhan manajemen ruangan yang lebih komprehensif.

c. Lakukan reorganisasi dan refaktorisasi source code untuk memastikan bahwa setiap modul terdokumentasi dengan baik dan siap untuk adaptasi atau ekspansi di masa depan sehingga dapat mempermudah pengembangan dan pemeliharaan jangka panjang.

[Halaman ini sengaja dikosongkan]

DAFTAR PUSTAKA

[1] Tsaqif, A. D., & Utomo, I. C. (2023). Aplikasi Penjualan Tiket Acara Berbasis Web Menggunakan Library React Js. Jurnal Komputer Dan Teknik Informatika, 1(1), 83–100. https://doi.org/10.54082/kontak.12.

[2] Maulana, M. R. (2017). Pengembangan Aplikasi Android Untuk Studi Bahasa Carakan Madura. *Journal Information Engineering and Educational Technology*.

[3] Mozilla.(n.a). Structuring the web with HTML.Media Komputindo, 2014. https://developer.mozilla.org/en-US/docs/Learn/HTML. Diakses tanggal 22 November 2023.

[4] Azure Data Studio. (2024). https://azure.microsoft.com/en-us/products/data-studio. Diakses tanggal 4 Januari 2024

[5] Difa Al Fansha, M. Y. (2021). Load Test pada Microservice yang menerapkan CQRS dan Event Sourcing. *Jurnal Buana Informatika*.

[Halaman ini sengaja dikosongkan]

BIODATA PENULIS I

Nama : Muhammad Ismail
Tempat, Tanggal Lahir : Surabaya, 09 Februari 2002
Jenis Kelamin : Laki-laki
Telepon : +6281358945230
Email : muhammadismail2418@gmail.com

AKADEMIS
Kuliah : Departemen Teknik Informatika – FTEIC , ITS
Angkatan : 2020
Semester : 7 (Tujuh)

BIODATA PENULIS II

Nama : Eldenabih Tavirazin Lutvie
Tempat, Tanggal Lahir : Surabaya, 23 November 2002
Jenis Kelamin : Laki-laki
Telepon : +6281249395866
Email : tavirazin@gmail.com

AKADEMIS
Kuliah : Departemen Teknik Informatika – FTEIC , ITS
Angkatan : 2020
Semester : 7 (Tujuh)