*Jurnal SAINTEKOM (Sains dan Teknologi Komputasi) Vol. 01, No. 01, Juni, 2025*
*Lembaga Penelitian, Pengembangan, dan Pengabdian Kepada Masyarakat UNBIN*
*e-ISSN: XXXXXXX*

Received: ***; Accepted: ***; Published: ***

# Penerapan Metode Test Driven Development untuk Menguji Rest Api pada Script di Visual Basic

**Bilal Abdilah[1*], Arif Harbani[2]**
[1,2] Program Studi Teknik Informatika, Fakultas Informatika dan Komputer, Universitas Binaniaga Indonesia
e-mail: bilalabdilah1604@gmail.com
Coresponding Author

***ABSTRACT***

*This research addresses challenges in software development using Visual Basic .NET (VB.NET) for REST API integration. As the demand for high-quality applications grows, manual testing becomes inefficient and prone to errors. Developers often struggle with identifying and fixing code issues without systematic testing, which can affect software quality. Test-Driven Development (TDD) emphasizes writing tests before coding, ensuring that each part of the code meets specifications. The study adopts the TDD cycle-writing tests, running tests to ensure failure, and writing code to pass them. TDD improves testing efficiency by detecting and fixing errors early, making testing more systematic and enhancing software quality. The research also explores tools and challenges in implementing TDD in VB.NET. Expert testing results show an 85% effectiveness score, while user testing gives a 73.3% score, reflecting positive feedback on its usability.*

***Keywords****: Test-Driven Development (TDD), REST API, Software Development, Automated Testing, Development Efficiency.*

**ABSTRAK**

Penelitian ini mengatasi tantangan dalam pengembangan perangkat lunak menggunakan Visual Basic.NET (VB.NET) untuk integrasi REST API. Seiring meningkatnya permintaan aplikasi berkualitas, pengujian manual menjadi tidak efisien dan rentan kesalahan. Pengembang kesulitan menemukan dan memperbaiki kesalahan tanpa pengujian sistematis, yang mempengaruhi kualitas perangkat lunak. Test-Driven Development (TDD) menekankan penulisan tes sebelum kode, memastikan kode memenuhi spesifikasi. Penelitian ini mengadopsi siklus TDD-menulis tes, menjalankan tes untuk memastikan kegagalan, dan menulis kode untuk memenuhi tes. TDD meningkatkan efisiensi pengujian dengan mendeteksi dan memperbaiki kesalahan lebih awal. Penelitian ini juga mengeksplorasi alat dan tantangan implementasi TDD di VB.NET. Hasil uji ahli menunjukkan efektivitas 85%, sementara uji pengguna mendapatkan skor 73,3%, yang menunjukkan umpan balik positif.

**Kata Kunci**: Test-Driven Development (TDD), REST API, Pengembangan Perangkat Lunak, Pengujian Otomatis, Efisiensi Pengembangan

## A. PENDAHULUAN

### 1. Latar Belakang

Pada Saat ini pengembangan perangkat lunak yang berkualitas sangatlah penting untuk di perhatikan untuk memastikan bahwa perangkat lunak layak dan dapat memenuhi kebutuhan agar mudah dan nyaman saat menggunakannya. Sebuah studi oleh McKinsey menunjukkan bahwa 70% proyek transformasi digital gagal karena kualitas perangkat lunak yang buruk. Salah satu cara untuk memastikan kualitas perangkat lunak adalah dengan melakukan uji coba. Uji coba tersebut dapat dilakukan dengan berbagai metode, salah satunya adalah metode Test Driven Development (TDD).

Metode Test Driven Development (TDD) adalah suatu metode yang mengutamakan pembuatan tes terlebih dahulu sebelum menulis kode implementasi. Dalam TDD, siklus pengembangan diawali dengan menulis tes yang merujuk pada fungsionalitas yang diinginkan dari kode yang akan dibuat. Setelah tes ditulis, kode implementasi yang sesuai dengan tes tersebut kemudian ditulis untuk memenuhi persyaratan yang ditetapkan oleh tes. Menurut (Nur et al., n.d.) Test Driven Development merupakan metode di mana perhatian utama tertuju pada penyusunan uji coba sebelum pelaksanaan implementasi kode aplikasi. Metode ini telah teruji efektivitasnya dalam meningkatkan mutu perangkat lunak dan mengurangi insiden kesalahan pada struktur kode aplikasi serta akan mempercepat waktu saat pengembangan progam ketika mengerjakan proyek dalam skala besar.

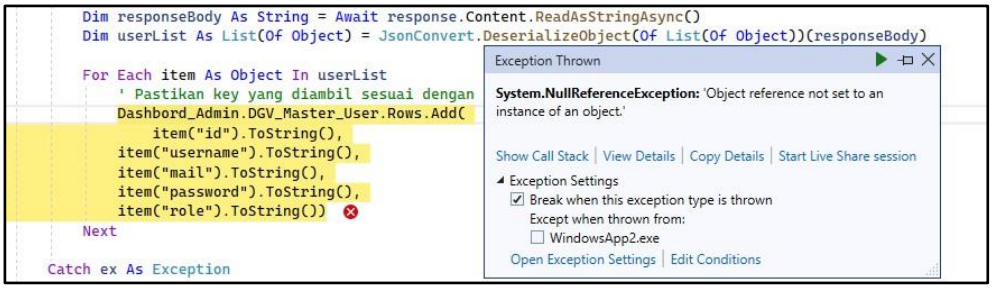
Visual Basic (VB.Net) merupakan bahasa pemograman yang di kembangkan oleh microsoft. Bahasa pemograman ini disebut bahasa yang paling basic dan sangat populer dikalangan mahasiswa dan pengembang perangkat lunak lain nya yang di kenal sebagai alat pengembangan perangkat lunak yang di gunakan untuk windows .Dalam melakukan pengembangan sering sekali mengalami kesulitan saat integrasikan dengan REST API sebagai alat pelantara penghubung aplikasi dengan data .

Rest API (Representational State Transfer API) merupakan rangkaian sebuah API yang menggunakan arsitektur Restful untuk berkomunikasi dengan sistem lain. Menurut (Rangga Gelar Guntara & Azkarin, 2023) API, telah menjadi nilai pertukaran data antar aplikasi, dan menjadi kunci keberhasilan dalam menciptakan ekosistem perangkat lunak yang terintegrasi. Dengan memberikan pemahaman utama dari penelitian ini memberikan arahan untuk meningkatkan fungsionalitas, kinerja dan fleksibilitas dalam melakukan pengembangan pada perangkat lunak visual basic dengan menerapkan metode TDD sebagai alat uji kode yang berkualitas saat ingin melakukan integrasi API.

### 2. Permasalahan

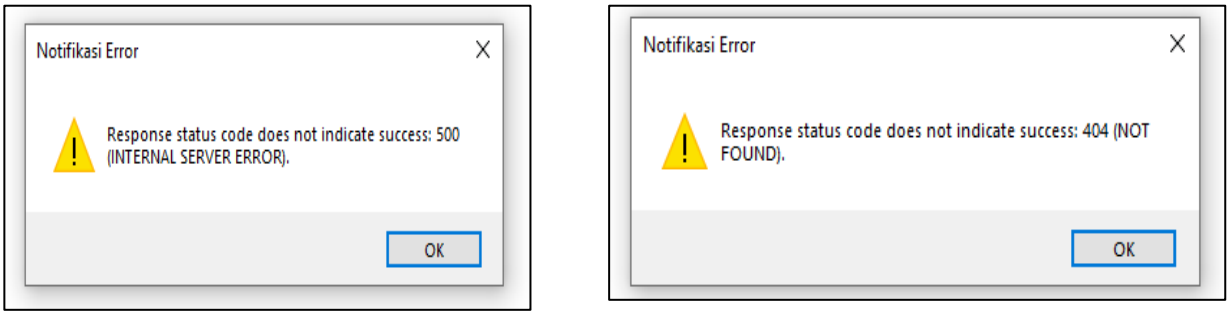
Dari hasil pengumpulan informasi diatas melalui kuesioner pengembang perangkat lunak yang menggunakan Visual Basic mengalami hambatan dalam penerapan REST API seperti kesulitan menulis kode , kesulitan mencari sumber

informasi yang tepat dan menemukan error ketika debugging saat melakukan pengujian secara manual. Sebagai contoh kesalahan yang sering terjadi dalam pengujian:




Gambar 1. Uji Manual Dengan Debugging

Dalam pengujian pada gambar di atas sulit sekali kita mengindefikasi kesalahan dengan melakukannya pengujian secara manual atau langsung debugging.




Gambar 2. Error Saat Debugging

Saat pengujian gambar diatas menunjukan sering nya terjadi kesalahan yang menujukan eror tanpa memberi informasi yang jelas dan sulitnya menemukan kesalahan error pada class code yang terjadinya kesalahan.

### 3. Tujuan

Adapun tujuan dalam penelitian ini adalah sebagai berikut:

a. Mengembangkan pemahaman tentang faktor-faktor yang dapat mempengaruhi saat pengujian REST API
b. Menilai efektivitas TDD itu sendiri sebagai solusi untuk meningkatkan kualitas dan efesiensi saat pengujian REST API di visual basic
c. Mengukur tingkat kelayakan penerapan TDD dalam pengembangan perangkat lunak menggunakan Visual Basic .NET

### 4. Asumsi Dan Keterbatasan

Asumsi ini didasarkan pada penelitian-penelitian sebelumnya yang menunjukkan bahwa metode TDD dapat membantu meningkatkan kualitas kode dan menemukan hambatan masalah yang ada saat proses pengujian. Dalam keterbatasan juga merupakan suatu hal yang membatasi ruang lingkup penelitian.

### 5. Hipotesis

Telah dilakukan penelitian oleh (Rohman & Ahmad, 2023) dengan judul "Pengembangan Aplikasi Mentor untuk Studi Kasus Pembelajaran Keterampilan Digital dengan Metode Test- Driven Development". Dihasilkan bahwa dengan mengunakan metode test driven development dan menyelesaikan pengujian usability dengan rata-rata hasil diatas 90 % dan sangat berdampak besar pada kelayakan aplikasi untuk di gunakan .Berdasarkan penelitian tersebut maka pada penelitian ini dibuat hipotesis yaitu dengan metode Test-Driven-Development diduga dapat meningkatkan kualitas dalam pengujian REST API di perangkat lunak visual.

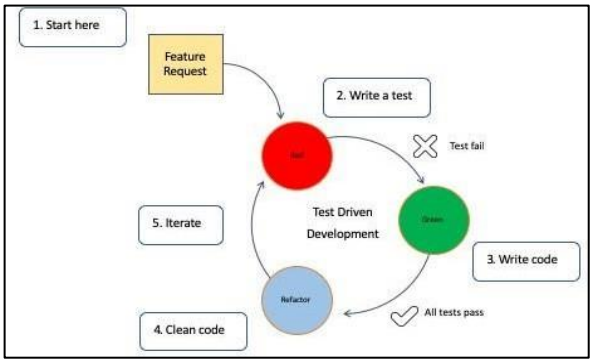
## B. METODE PENELITIAN

### 1. Metode Yang Di Terapkan

#### a. Test Driven Development

Test-Driven Development (TDD) adalah metodologi pengembangan perangkat lunak yang mengutamakan pembuatan tes otomatis sebelum implementasi kode. Proses ini melibatkan waktu pengujian berulang yang dimulai dengan penulisan tes dan dilanjutkan dengan implementasi kode untuk memenuhi persyaratan tes

tersebut.




Gambar 3. Alur TDD
("Sumber: Basu Mallick Chiradeep, 2022")

Dalam flowchart atau workflow ini, ada warna merah, hijau, dan biru. Oleh karena itu, namanya adalah Red-Green-Refactoring. Warna-warna ini memiliki arti tersendiri, yaitu menunjukkan status tes dalam satu siklus:

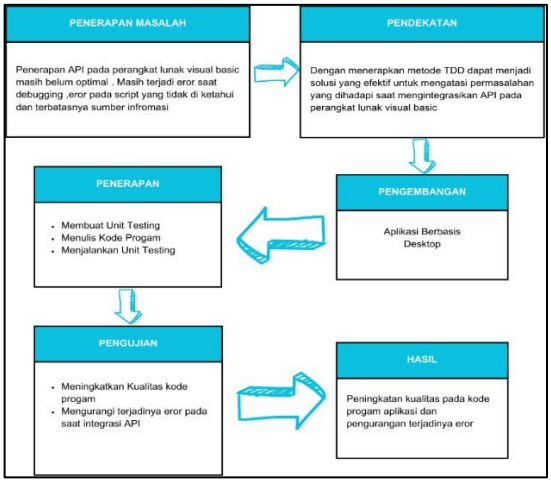
1) Warna merah menunjukkan bahwa kode tidak bekerja.
2) Sementara, hijau menunjukkan bahwa semuanya bekerja dengan baik, tetapi belum optimal.
3) Terakhir, warna biru berarti tester sedang melakukan refactoring dan mengupayakan untuk membuat kode yang lebih baik.

Manfaat dari penerapan API mengunakan Metode Test Driven Development:

1) Target output lebih objectif
2) Memastikan bahwa setiap fitur API memiliki tes yang mencakup semua skenario pengunaaan yang dapat digunakan.
3) Meminimalkan bug dan kesalahan implementasi API dengan adanya pengujian otomatis yang komprehensif.
4) Membuat stuktur kode yang mudah di uji dan yang terpenting tetap dalam konsep pengembangan aplikasi mengunakan API.

**b. Kerangka Berfikir**

Kerangka pemikiran ini dibuat mewakilkan konsep pemecahan masalah penelitian tentang penerapan TDD merupakan solusi yang efektif untuk meningkatkan kualitas. Berikut gambaran flow kerangka pemikiran:




Gambar 4. Kerangka Berfikir

**c. Prosedur Pengembangan**

Dalam prosedur pengembangan ini terdapat model prosudural agile yang berkaitannya dengan metode TDD yang dimana saat melakukan pendekatan pengembangan perangkat lunak yang mengutamakan kolaborasi, fleksibilitas, dan iterasi dalam proses pengembangan. Dalam konteks ini, TDD (Test-Driven Development) berfungsi sebagai salah satu praktik yang mendukung prinsip-prinsip Agile. Berikut adalah penjelasan lebih lanjut mengenai model prosedural Agile dan bagaimana TDD berperan di dalamnya:

1) Prinsip Dasar Agile

Pada tahap ini prinsip Agile berfokus pada pengembangan perangkat lunak yang responsif terhadap perubahan dan kebutuhan pengguna yang dimana terkait dengan prinsip dasarnya seperti kolaborasi dengan tim,iterasi dan inkremental serta flexsibilitas .

2) TDD Dalam Konteks Agile

Dalam konteks ini TDD adalah praktik pengembangan perangkat lunak yang berfokus pada penulisan tes sebelum menulis kode implementasi. TDD memiliki beberapa manfaat yang sejalan dengan prinsip-prinsip Agile seperti Meningkatkan Kualitas Kode, Memfasilitasi Refactoring, Dokumentasi yang Lebih Baik Responsif terhadap Perubahan.

3) Integrasi TDD dalam Proses Agile

Dalam model prosedural Agile, TDD dapat diintegrasikan ke dalam siklus pengembangan sebagai berikut:

Perencanaan Sprint: Pada awal setiap sprint, tim menentukan fitur yang akan dikembangkan dan kasus uji yang relevan. Ini mencakup penentuan skenario pengujian yang akan diimplementasikan menggunakan TDD.

Pengembangan Iteratif: Selama pengembangan, pengembang menulis tes untuk setiap fitur sebelum menulis kode. Setelah tes ditulis, mereka menjalankan tes tersebut dan menulis kode yang diperlukan untuk membuat tes lulus.

Uji dan Tinjau: Setelah pengembangan selesai, tim melakukan tinjauan kode dan pengujian untuk memastikan bahwa semua tes lulus. Ini juga mencakup pengujian integrasi untuk memastikan bahwa fitur baru berfungsi dengan baik dalam konteks sistem yang lebih besar.

Retrospektif: Setelah setiap sprint, tim melakukan retrospektif untuk mengevaluasi proses pengembangan, termasuk efektivitas penggunaan TDD. Ini membantu tim untuk terus meningkatkan praktik mereka dan mengidentifikasi area untuk perbaikan.

Dengan keterhubungan model prosedural Agile dan TDD saling melengkapi dalam menciptakan proses pengembangan perangkat lunak yang efisien dan berkualitas tinggi. Dengan mengintegrasikan TDD ke dalam praktik Agile, tim dapat meningkatkan kolaborasi, fleksibilitas, dan responsivitas terhadap perubahan, sambil memastikan bahwa perangkat lunak yang dihasilkan memenuhi standar kualitas yang tinggi. Pendekatan ini tidak hanya meningkatkan keandalan perangkat lunak tetapi juga meningkatkan kepuasan pengguna melalui pengiriman fitur yang lebih cepat dan lebih baik.

## C. HASIL DAN PEMBAHASAN

### 1. Hasil Penelitian

Dalam melakukan penerapan metode Test Driven Development (TDD) dalam pengujian REST API pada pengembangan perangkat lunak Visual Basic memerlukan persiapan dan sumber daya yang memadai untuk memastikan efektivitas dan efisiensinya. Berikut adalah bagian awal mengenai kebutuhan yang harus dipenuhi:

a. Analisa Kebutuhan: IDE (Intergrated Development Enviroment), Framework Pengujian Xunit, Library alat pengujian API, Sumber Daya Manusia, System Control (GIT).
b. Analisa Metode: Keunggulan Metode TDD, Tantangan Dalam Penerepan Metode TDD, Impelemtasi TDD Untuk Menguji Rest API.
c. Perancangan Kebutuhan Infrastuktur.

### 2. Pembahasan Penelitian

Dalam penelitian ini akan membahas bagaimana metode Test Driven Development (TDD) diterapkan dalam pengujian REST API pada pengembangan perangkat lunak menggunakan Visual Basic. Pendekatan TDD memungkinkan pengembang untuk menulis tes terlebih dahulu sebelum mengembangkan fungsionalitas, sehingga memastikan bahwa setiap komponen dari REST API bekerja sesuai dengan yang diharapkan.

**Tahapan Penerepan Metode Pada Code:**

Pada tahapan ini akan di jelaskan bahwa bagaimana cara penerepannya:

```
<Fact>
| 0 references
Public Async Function apiClient_POST_RED() As Task 'RED
    Try
        ' Arrange
        Dim baseAddress As String = "http://127.0.0.1:5000/users"
        Dim client As New HttpClient()
        client.BaseAddress = New Uri(baseAddress)

        Dim formData As New MultipartFormDataContent()
        formData.Add(New StringContent("TEST1"), "username") ' Kesalahan pada nama field
        formData.Add(New StringContent("mail@mail.com"), "mail")
        formData.Add(New StringContent("123"), "password")
        formData.Add(New StringContent("User"), "role")

        ' Act
        Dim response As HttpResponseMessage = Await client.PostAsync("users", formData)

        ' Assert
        response.EnsureSuccessStatusCode() ' Memastikan status kode sukses
        Dim responseBody As String = Await response.Content.ReadAsStringAsync()
        Console.WriteLine("Response: " & responseBody)

    Catch ex As HttpRequestException
        Console.WriteLine("HTTP Request Error: " & ex.Message)
        Assert.True(False, "HTTP Request failed: " & ex.Message)
    Catch ex As Exception
        Console.WriteLine("Unexpected Error: " & ex.Message)
        Assert.True(False, "Unexpected exception: " & ex.Message)
    End Try
End Function
```

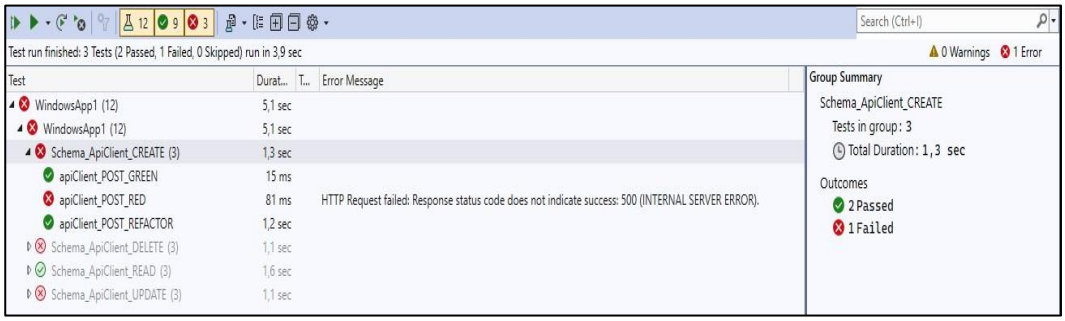
Gambar 5. Codingan Class Create TDD (Red)

```
<Fact>
| 0 references
Public Async Function apiClient_POST_GREEN() As Task  'GREEEN
    Try
        ' Arrange
        Dim baseAddress As String = "http://127.0.0.1:5000/users"
        Dim client As New HttpClient()
        client.BaseAddress = New Uri(baseAddress)

        Dim formData As New MultipartFormDataContent()
        formData.Add(New StringContent("TEST1"), "name") ' Nama field yang benar
        formData.Add(New StringContent("mail@mail.com"), "mail")
        formData.Add(New StringContent("123"), "password")
        formData.Add(New StringContent("User"), "role")

        ' Act
        Dim response As HttpResponseMessage = Await client.PostAsync("users", formData)

        ' Assert
        response.EnsureSuccessStatusCode() ' Memastikan status kode sukses
        Dim responseBody As String = Await response.Content.ReadAsStringAsync()
        Console.WriteLine("Response: " & responseBody)

    Catch ex As HttpRequestException
        Console.WriteLine("HTTP Request Error: " & ex.Message)
        Assert.True(False, "HTTP Request failed: " & ex.Message)
    Catch ex As Exception
        Console.WriteLine("Unexpected Error: " & ex.Message)
        Assert.True(False, "Unexpected exception: " & ex.Message)
    End Try
End Function
```

Gambar 6. Codingan Class Create TDD (GREEN)

```
<Fact>
| 0 references
Public Async Function apiClient_POST_REFACTOR() As Task 'REFACTOR
    Try
        ' Arrange
        Dim client As New HttpClient() With {
        .BaseAddress = New Uri("http://127.0.0.1:5000/users")
    }

        Dim formData As New MultipartFormDataContent From {
        {New StringContent("TEST1"), "name"},
        {New StringContent("mail@mail.com"), "mail"},
        {New StringContent("123"), "password"},
        {New StringContent("User"), "role"}
    }

        ' Act
        Dim response As HttpResponseMessage = Await client.PostAsync("users", formData)

        ' Assert
        response.EnsureSuccessStatusCode() ' Memastikan status kode sukses
        Dim responseBody As String = Await response.Content.ReadAsStringAsync()
        Console.WriteLine("Response: " & responseBody)

    Catch ex As HttpRequestException
        Console.WriteLine("HTTP Request Error: " & ex.Message)
        Assert.True(False, "HTTP Request failed: " & ex.Message)
    Catch ex As Exception
        Console.WriteLine("Unexpected Error: " & ex.Message)
        Assert.True(False, "Unexpected exception: " & ex.Message)
    End Try
End Function
```

Gambar 7. Class Create TDD (REFACTOR)




Gambar 8. Hasil Dari Pengujian Class Create TDD

Hasil dari pengujian ini di dapatkan dari codingan class create yang sebelumnya kita buat dan mendapatkan sesuai dengan apa yang kita skema kan serta bisa di nyatakan layak untuk mengunakan code tersebut untuk masuk ketahap selanjutnya nantinya.

```
    <Fact>
    | 0 references
    Public Async Function apiClient_GET_RED() As Task
        Try
            ' Arrange
            Dim client As New HttpClient() With {
            .BaseAddress = New Uri("http://127.0.0.1:5000/")
        }

            ' Act
            Dim response As HttpResponseMessage = Await client.GetAsync("users")

            ' Assert
            response.EnsureSuccessStatusCode()
            Dim responseBody As String = Await response.Content.ReadAsStringAsync()
            Dim userList As List(Of Object) = JsonConvert.DeserializeObject(Of List(Of Object))(responseBody)

            Assert.NotNull(userList)
            Assert.True(userList.Count > 0)

        Catch ex As Exception
            Assert.True(False, "Test failed: " & ex.Message)
        End Try
    End Function
```

Gambar 9. Codingan Class Read (RED)

```
<Fact>
| 0 references
Public Async Function apiClient_GET_GREEN() As Task
    Try
        Dim stringWriter As New StringWriter()
        Dim originalConsoleOut As TextWriter = Console.Out
        Console.SetOut(stringWriter)


        Dim client As New HttpClient() With {
        .BaseAddress = New Uri("http://127.0.0.1:5000/")
    }

        Dim response As HttpResponseMessage = Await client.GetAsync("users")
        response.EnsureSuccessStatusCode()

        Dim responseBody As String = Await response.Content.ReadAsStringAsync()
        Dim userList As List(Of Object) = JsonConvert.DeserializeObject(Of List(Of Object))(responseBody)

        For Each item As Object In userList
            Console.WriteLine($"ID: {item("id")}, Name: {item("name")}, Mail: {item("mail")}, Password: {item("password")}, Role: {item("role")}")
        Next

    Catch ex As Exception
        Console.WriteLine("Error: " & ex.Message)
    End Try
End Function
```

Gambar 10. Codingan Class Read (GREEN)

```
<Fact>
| 0 references
Public Async Function apiClient_GET_REFACTOR() As Task(Of List(Of Object))
    Dim client As New HttpClient() With {
    .BaseAddress = New Uri("http://127.0.0.1:5000/")
}

    Dim response As HttpResponseMessage = Await client.GetAsync("users")
    response.EnsureSuccessStatusCode()

    Dim responseBody As String = Await response.Content.ReadAsStringAsync()
    Return JsonConvert.DeserializeObject(Of List(Of Object))(responseBody)

    'DIBAWAH INI UNTUK MEMINTA HASIL YG INGIN DI TAMPILKAN SERTI DATAGRIDVIEW
End Function
```

Gambar 11. Codingan Class Read (REFACTOR)


Gambar 12. Hasil Dari Pengujian Class Read TDD

Hasil dari pengujian ini di dapatkan dari codingan class read yang sebelumnya kita buat dan mendapatkan sesuai dengan apa yang kita skema kan serta bisa di nyatakan layak untuk mengunakan code tersebut untuk masuk ketahap selanjutnya nantinya.

```
<Fact>
| 0 references
Public Async Function apiClient_UPDATE_RED() As Task 'RED
    Dim id As String
    id = "785" ' KODE ID Tidak benar atau tidak ada


    Try
        'Arrange
        Dim baseaddress As String = "http://127.0.0.1:5000/"
        Dim client As New HttpClient()
        client.BaseAddress = New Uri(baseaddress)

        Dim formdata As New MultipartFormDataContent()
        formdata.Add(New StringContent("TEST1"), "name")
        formdata.Add(New StringContent("mail@mail.com"), "mail")
        formdata.Add(New StringContent("123"), "password")
        formdata.Add(New StringContent("User"), "role")

        'Act
        Dim response As HttpResponseMessage = Await client.PutAsync($"users/{id}", formdata)

        'Assert
        response.EnsureSuccessStatusCode()
        Dim responseBody As String = Await response.Content.ReadAsStringAsync()
        Console.WriteLine("Response: " & responseBody)

    Catch ex As HttpRequestException
        Console.WriteLine("HTTP Request Error: " & ex.Message)
        Assert.True(False, "HTTP Request failed: " & ex.Message)
    Catch ex As Exception
        Console.WriteLine("Unexpected Error: " & ex.Message)
        Assert.True(False, "Unexpected exception: " & ex.Message)
    End Try
End Function
```

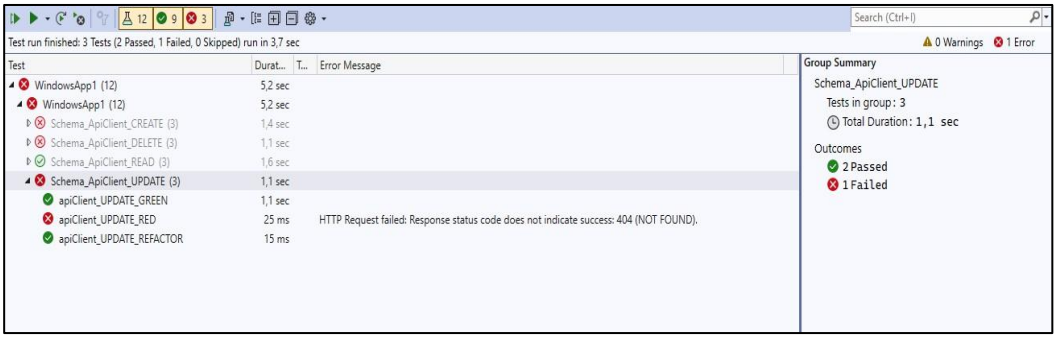
Gambar 13. Codingan Class Update (RED)

```
<Fact>
| 0 references
Public Async Function apiClient_UPDATE_GREEN() As Task 'GREEN
    Dim id As String

    id = "5" ' KODE ID BENAR

    Try
        'Arrange
        Dim baseaddress As String = "http://127.0.0.1:5000/"
        Dim client As New HttpClient()
        client.BaseAddress = New Uri(baseaddress)

        Dim formdata As New MultipartFormDataContent()
        formdata.Add(New StringContent("TESTGREEND"), "name")
        formdata.Add(New StringContent("mail@mail.com"), "mail")
        formdata.Add(New StringContent("123"), "password")
        formdata.Add(New StringContent("User"), "role")

        'Act
        Dim response As HttpResponseMessage = Await client.PutAsync($"users/{id}", formdata)

        'Assert
        response.EnsureSuccessStatusCode()
        Dim responseBody As String = Await response.Content.ReadAsStringAsync()
        Console.WriteLine("Response: " & responseBody)

    Catch ex As HttpRequestException
        Console.WriteLine("HTTP Request Error: " & ex.Message)
        Assert.True(False, "HTTP Request failed: " & ex.Message)
    Catch ex As Exception
        Console.WriteLine("Unexpected Error: " & ex.Message)
        Assert.True(False, "Unexpected exception: " & ex.Message)
    End Try
End Function
```

Gambar 14. Codingan Class Update (GREEN)

```
<Fact>
| 0 references
Public Async Function apiClient_UPDATE_REFACTOR() As Task
    Dim id As String = "5" ' KODE ID BENAR

    Try
        ' Arrange
        Dim client As New HttpClient() With {
        .BaseAddress = New Uri("http://127.0.0.1:5000/")
    }

        Dim formData As New MultipartFormDataContent From {
        {New StringContent("TEST_REFACTOR"), "name"},
        {New StringContent("mail@mail.com"), "mail"},
        {New StringContent("123"), "password"},
        {New StringContent("User"), "role"}
    }

        ' Act
        Dim response As HttpResponseMessage = Await client.PutAsync($"users/{id}", formData)

        ' Assert
        response.EnsureSuccessStatusCode()
        Dim responseBody As String = Await response.Content.ReadAsStringAsync()
        Console.WriteLine("Response: " & responseBody)

    Catch ex As HttpRequestException
        Console.WriteLine("HTTP Request Error: " & ex.Message)
        Assert.True(False, "HTTP Request failed: " & ex.Message)
    Catch ex As Exception
        Console.WriteLine("Unexpected Error: " & ex.Message)
        Assert.True(False, "Unexpected exception: " & ex.Message)
    End Try
End Function
```

Gambar 15. Codingan Class Update (REFACTOR)




Gambar 16. Hasil Dari Pengujian Class Update TDD

Hasil dari pengujian ini di dapatkan dari codingan class update yang sebelumnya kita buat dan mendapatkan sesuai dengan apa yang kita skema kan serta bisa di nyatakan layak untuk mengunakan code tersebut untuk masuk ketahap selanjutnya nantinya.

```
<Fact>
0 references
Public Async Function apiClient_DELETE_RED() As Task 'RED
    Dim id As String
    id = "1" ' KODE ID Tidak benar atau tidak ada

    Try
        'Arrange
        Dim baseaddress As String = "http://127.0.0.1:5000/"
        Dim client As New HttpClient()
        client.BaseAddress = New Uri(baseaddress)

        'Act
        Dim response As HttpResponseMessage = Await client.DeleteAsync($"users/{id}")

        'Assert
        response.EnsureSuccessStatusCode()
        Dim responseBody As String = Await response.Content.ReadAsStringAsync()
        Console.WriteLine("Response: " & responseBody)

    Catch ex As HttpRequestException
        Console.WriteLine("HTTP Request Error: " & ex.Message)
        Assert.True(False, "HTTP Request failed: " & ex.Message)
    Catch ex As Exception
        Console.WriteLine("Unexpected Error: " & ex.Message)
        Assert.True(False, "Unexpected exception: " & ex.Message)
    End Try
End Function
```

Gambar 17. Codingan Class Delete (RED)

```
<Fact>
0 references
Public Async Function apiClient_DELETE_GREEN() As Task 'GREEN
    Dim id As String
    id = "10" ' KODE ID TERSEDIA
    Try
        'Arrange
        Dim baseaddress As String = "http://127.0.0.1:5000/"
        Dim client As New HttpClient()
        client.BaseAddress = New Uri(baseaddress)

        'Act
        Dim response As HttpResponseMessage = Await client.DeleteAsync($"users/{id}")

        'Assert
        response.EnsureSuccessStatusCode()
        Dim responseBody As String = Await response.Content.ReadAsStringAsync()
        Console.WriteLine("Response: " & responseBody)

    Catch ex As HttpRequestException
        Console.WriteLine("HTTP Request Error: " & ex.Message)
        Assert.True(False, "HTTP Request failed: " & ex.Message)
    Catch ex As Exception
        Console.WriteLine("Unexpected Error: " & ex.Message)
        Assert.True(False, "Unexpected exception: " & ex.Message)
    End Try
End Function
```

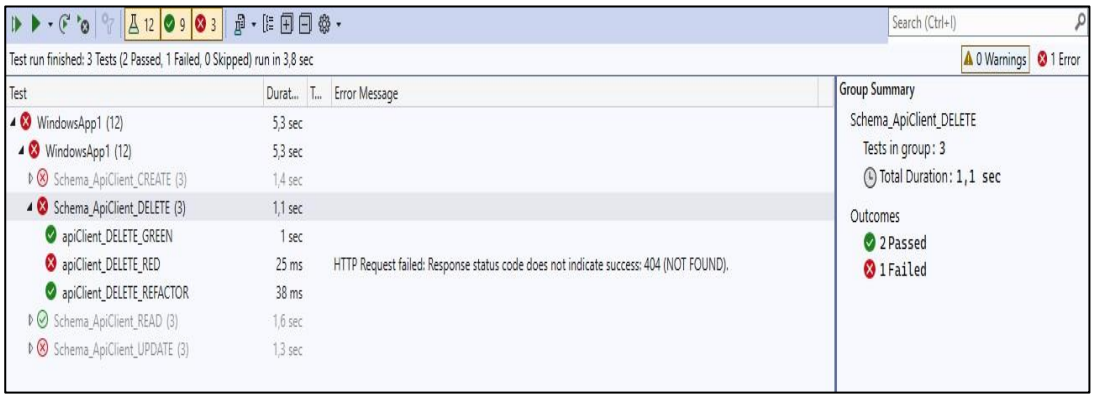
Gambar 18. Codingan Class Delete (GREEN)

```
<Fact>
0 references
Public Async Function apiClient_DELETE_REFACTOR() As Task

    Dim id As String = "10" ' KODE ID TERSEDIA

    Try
        ' Arrange
        Dim client As New HttpClient() With {
        .BaseAddress = New Uri("http://127.0.0.1:5000/")
    }

        ' Act
        Dim response As HttpResponseMessage = Await client.DeleteAsync($"users/{id}")

        ' Assert
        response.EnsureSuccessStatusCode()
        Dim responseBody As String = Await response.Content.ReadAsStringAsync()
        Console.WriteLine("Response: " & responseBody)

    Catch ex As HttpRequestException
        Console.WriteLine("HTTP Request Error: " & ex.Message)
        Assert.True(False, "HTTP Request failed: " & ex.Message)
    Catch ex As Exception
        Console.WriteLine("Unexpected Error: " & ex.Message)
        Assert.True(False, "Unexpected exception: " & ex.Message)
    End Try
End Function
```

Gambar 19. Codingan Class Delete (Green)

```
<Fact>
0 references
Public Async Function apiClient_DELETE_REFACTOR() As Task

    Dim id As String = "10" ' KODE ID TERSEDIA

    Try
        ' Arrange
        Dim client As New HttpClient() With {
        .BaseAddress = New Uri("http://127.0.0.1:5000/")
    }

        ' Act
        Dim response As HttpResponseMessage = Await client.DeleteAsync($"users/{id}")

        ' Assert
        response.EnsureSuccessStatusCode()
        Dim responseBody As String = Await response.Content.ReadAsStringAsync()
        Console.WriteLine("Response: " & responseBody)

    Catch ex As HttpRequestException
        Console.WriteLine("HTTP Request Error: " & ex.Message)
        Assert.True(False, "HTTP Request failed: " & ex.Message)
    Catch ex As Exception
        Console.WriteLine("Unexpected Error: " & ex.Message)
        Assert.True(False, "Unexpected exception: " & ex.Message)
    End Try
End Function
```

Gambar 20. Codingan Class Delete (REFACTOR)




Gambar 21. Hasil Dari Pengujian Class Delete TDD

Hasil dari pengujian ini di dapatkan dari codingan class delete yang sebelumnya kita buat dan mendapatkan sesuai dengan apa yang kita skema kan serta bisa di nyatakan layak untuk mengunakan code tersebut untuk masuk ketahap selanjutnya nantinya.

Dengan semua pengujian yang dilakukan, pengembang dapat langsung menerapkannya pada aplikasi yang sedang dibuat. Pengembang hanya perlu membuat kelas kode dan mengikuti skrip yang telah diuji di awal. Hal ini sangat berguna dalam mengurangi risiko terjadinya bug, membuat kode lebih efisien, dan memudahkan pemahaman serta pemeliharaan kode.

## D. KESIMPULAN

Berdasarkan hasil penelitian dan pengujian dalam penerapan metode TDD dan pengujian REST API, dapat disimpulkan beberapa hal berikut:

1. Penerapan TDD terbukti efektif dalam meningkatkan kualitas pengujian REST API. Dengan menulis tes terlebih dahulu, setiap fungsionalitas API diuji secara menyeluruh sebelum diimplementasikan.
2. Metode TDD mendorong pengembang untuk menulis kode yang bersih dan terstruktur. Proses refactoring yang dilakukan setelah tes berhasil memastikan bahwa kode tetap efisien dan mudah dipahami.
3. Setelah dilakukan uji ahli dan uji pengguna, uji ahli menunjukkan hasil 85 % yang berarti rest api dengan TDD sangat mudah dan efetif dan uji pengguna menunjukkan hasil sebesar 73.3% yang berarti pendekatan ini dinilai baik saat digunakan.

Adapun saran untuk penelitian selanjutnya adalah:

1. Untuk pengembangan lebih lanjut sistem ini dapat di kembangkan mengunakan metode yang berbeda ataupun mengkombinasikan metode test driven development dengan metode yang lain.
2. Diharapkan melakukan pelatihan agar penerapan metode yang lebih efektif
3. Diharapkan saat pengembangan perangkat lunak dengan TDD harus disertai dokumentasi yang baik untuk memudahkan pemeliharaan dan pengembangan.

## E. DAFTAR PUSTAKA

[1] Ahmad Dahlan Jamalludin, U., Yuliansyah, H., Winiati, S., Riadi, I., Ahmad Dahlan, U., & Prof Dr Supomo, J. (2018). Implementasi Test Driven Development Pada Pengembangan Aplikasi Android Untuk Mahasiswa. In *Jurnal Ilmu Teknik Elektro Komputer dan Informatika (JITEKI)* (Vol. 4, Issue 1). http://simeru.uad.ac.id

[2] BasuMallick Chiradeep. (2022, September 29). *What Is TDD (Test Driven Development)? Process, Importance, and Limitations*. https://www.spiceworks.com/tech/devops/articles/what-is-tdd/

[3] FAIRUZ PANE, A. F. (2023). *EVALUASI USER INTERFACE PADA APLIKASI ANCOL UNTUK MENINGKATKAN USER EXPERIENCE MENGGUNAKAN USABILITY EVALUATION METHODS.*

[4] Federick, J., & Pakereng, M. A. I. (2020). IJCIT (Indonesian Journal on Computer and Information Technology) Test-Driven Development pada Pengembangan Aplikasi Android untuk Memantau COVID-19. In *IJCIT (Indonesian Journal on Computer and Information Technology)* (Vol. 6, Issue 1). https://creativecommons.org/licenses/by-sa/4.0/

[5] Huda Nurul. (2023, May 18). *Mengenal Apa itu API & Manfaatnya guna Pengembangan Aplikasi*. https://www.dewaweb.com/blog/apa-itu-api/

[6] lambdatest.com. (2020). *Detailed Guide On Test Coverage*. https://Www.Lambdatest.Com/Learning-Hub/Test-Coverage.

[7] Microsoft. (2023, May 25). *Apa itu Visual Studio?* Https://Learn.Microsoft.Com/Id- Id/Visualstudio/Get-Started/Visual-Studio-Ide?View=vs-2022.

[8] Nur, A., Thohari, A., & Amalia, A. E. (n.d.). *IMPLEMENTASI TEST DRIVEN DEVELOPMENT DALAM PENGEMBANGAN APLIKASI BERBASIS WEB*. http://jurnal.umk.ac.id/index.php/sitech

[9] Parlika, R., Wijaya, D. C., Khariono, H., KSiregar, I., & SPP Arianto, C. (n.d.). *Parlika, Wijaya, Khariono, Siregar, Arianto Implementasi API Region Visual Basic 6 Untuk Membentuk Huruf Hijaiyah IMPLEMENTASI API REGION VISUAL BASIC 6 UNTUK MEMBENTUK HURUF HIJAIYAH.*

[10] Rafiadly, M., Fauzi, R., & Musnansyah, A. (2023). Perancangan Aplikasi Naviku untuk Memberikan Informasi Navigasi Kepada Tunanetra Menggunakan Metode Test Driven Development. *Journal of Information System Research*, *4*(4), 1455–1463. https://doi.org/10.47065/josh.v4i4.3948

[11] Rangga Gelar Guntara, & Azkarin, V. (2023). Implementasi dan Pengujian REST API Sistem Reservasi Ruang Rapat dengan Metode Black Box Testing. *Jurnal Minfo Polgan*, *12*(1). https://doi.org/10.33395/jmp.v12i1.12691

[12] Rizkyana, M. A., Yunanto, A., Herdian, A., & Ainul, Y. R. (n.d.). *Implementasi Unit Testing Menggunakan Metode Test-First Development* (Vol. 7, Issue 1).

[13] Rohman, A., & Ahmad, A. (2023). *Pengembangan Aplikasi Mentor untuk Studi Kasus Pembelajaran Keterampilan Digital dengan Metode Test-Driven Development* (Vol. 10, Issue 1).

[14] Siregar, L. (2020). Review Pengujian Keamanan Perangkat Lunak dalam Software Development Life Cycle (SDLC). *Jurnal ASEECT*, *1*(3).

[15] Sunardi, S., Riadi, I., & Raharja, P. A. (2019). Analisis Application Programming Interface Pada Mobile E-Voting Menggunakan Metode Test-Driven Development. *Techno (Jurnal Fakultas Teknik, Universitas Muhammadiyah Purwokerto)*, *20*(2). https://doi.org/10.30595/techno.v20i2.4266

[16] Veni Manik, Hetty Primasari, C., Yohanes Priadi Wibisono, & Aloysius Bagas Pradipta Irianto. (2021). Evaluasi Usability pada Aplikasi Mobile ACC.ONE menggunakan System Usability Scale (SUS) dan Usability Testing. *Jurnal Sains Dan Informatika*, *7*(1), 1–10. https://doi.org/10.34128/jsi.v7i1.286

[17] Winaryati, E., Munsarif, M., Mardiana, & Suwahono. (2021). *Cercular Model of RD&D (Model RD&D Pendidikan dan Sosial)* (S. Nahidloh, A. Rochmah, & Danillstr, Eds.). PENERBIT KBM INDONESIA.

[18] Yutia, S. N., & Satrinia, D. (n.d.). Automated Functional Testing pada API menggunakan Keyword Driven Framework. *OPEN ACCESS J. OF ICT*, *3*(1), 65–078.