LAPORAN PRAKTIKUM

Pengujian Perangkat Lunak

Report_Fault_Based_Mutation

Ferdianty Mariany Panjaitan	:	11423008
Ruth Heppi Evelin Sinambela	:	11423011
Andika Parlinggoman Tampubolon	:	11423021
Rino Jeremia Christopher Gultom	:	11423033
Abet Bertand M.P Manurung	:	11423038
Natan Obet Nego Hutahaen	:	11423054

Code Original:

```go
package controller

import (
    "github.com/gin-gonic/gin"
    "cbt-api/config"
    "cbt-api/entity"
    "cbt-api/auth"
    "golang.org/x/crypto/bcrypt"
    "net/http"
    "gorm.io/gorm"
)

func Login(c *gin.Context) {
    var userInput struct {
        Email    string `json:"email"`
        Password string `json:"password"`
    }

    // Parsing input JSON
    if err := c.ShouldBindJSON(&userInput); err != nil {
        c.JSON(http.StatusBadRequest, gin.H{"error": "Invalid input"})
        return
    }

    // Mencari user berdasarkan email
    var user entity.Users
    if err := config.DB.Where("email = ?", userInput.Email).First(&user).Error; err != nil {
        if err == gorm.ErrRecordNotFound {
            c.JSON(http.StatusUnauthorized, gin.H{"error": "User not found"})
            return
        }
        c.JSON(http.StatusInternalServerError, gin.H{"error": "Internal server error"})
        return
    }

    // Membandingkan password yang dimasukkan dengan password yang disimpan
    if err := bcrypt.CompareHashAndPassword([]byte(user.Password), []byte(userInput.Password)); err != nil {
        c.JSON(http.StatusUnauthorized, gin.H{"error": "Incorrect password"})
        return
    }

    // Menghasilkan JWT token jika password benar
    token, err := auth.GenerateJWT(user.Id, user.Email)
    if err != nil {
        c.JSON(http.StatusInternalServerError, gin.H{"error": "Could not generate token"})
        return
    }

    // Mengembalikan token JWT ke client
    c.JSON(http.StatusOK, gin.H{"message": "Login successful", "token": token})
}
```

Mutant Versi 1 : Mengganti pengecekan error pada Password (Ferdianty)

```go
package controller

import (
    "github.com/gin-gonic/gin"
    "cbt-api/config"
    "cbt-api/entity"
    "cbt-api/auth"
    "golang.org/x/crypto/bcrypt"
    "net/http"
    "gorm.io/gorm"
)

func Login(c *gin.Context) {
    var userInput struct {
        Email    string `json:"email"`
        Password string `json:"password"`
    }

    // Parsing input JSON
    if err := c.ShouldBindJSON(&userInput); err != nil {
        c.JSON(http.StatusBadRequest, gin.H{"error": "Invalid input"})
        return
    }

    // Mencari user berdasarkan email
    var user entity.Users
    if err := config.DB.Where("email = ?", userInput.Email).First(&user).Error; err != nil {
        if err == gorm.ErrRecordNotFound {
            c.JSON(http.StatusUnauthorized, gin.H{"error": "User not found"})
            return
        }
        c.JSON(http.StatusInternalServerError, gin.H{"error": "Internal server error"})
        return
    }

    // Membandingkan password yang dimasukkan dengan password yang disimpan
    if err := bcrypt.CompareHashAndPassword([]byte(user.Password), []byte(userInput.Password)); err == nil {  // Salah: harusnya err != nil
        c.JSON(http.StatusUnauthorized, gin.H{"error": "Incorrect password"})
        return
    }

    // Menghasilkan JWT token jika password benar
    token, err := auth.GenerateJWT(user.Id, user.Email)
    if err != nil {
        c.JSON(http.StatusInternalServerError, gin.H{"error": "Could not generate token"})
        return
    }

    // Mengembalikan token JWT ke client
    c.JSON(http.StatusOK, gin.H{"message": "Login successful", "token": token})
}

```

Mengganti pengecekan error pada password (bcrypt.CompareHashAndPassword)

- **Mutant**: Jika pengecekan err == nil digunakan pada bcrypt.CompareHashAndPassword, maka fungsi akan gagal mendeteksi password yang salah.

- **Test Case yang relevan**: Test case yang memverifikasi login dengan password yang salah **dapat** membunuh mutant ini karena mutant ini mengubah cara pengecekan password.

Mutant Versi 2 : Mengubah status HTTP pada kesalahan “Invalid Input” (Ruth)

```go
package controller

import (
    "github.com/gin-gonic/gin"
    "cbt-api/config"
    "cbt-api/entity"
    "cbt-api/auth"
    "golang.org/x/crypto/bcrypt"
    "net/http"
    "gorm.io/gorm"
)

func Login(c *gin.Context) {
    var userInput struct {
        Email    string `json:"email"`
        Password string `json:"password"`
    }

    // Parsing input JSON
    if err := c.ShouldBindJSON(&userInput); err != nil {
        c.JSON(http.StatusBadRequest, gin.H{"error": "Invalid input"})
        return
    }

    // Mencari user berdasarkan email
    var user entity.Users
    if err := config.DB.Where("email = ?", userInput.Email).First(&user).Error; err != nil {
        if err == gorm.ErrRecordNotFound {
            c.JSON(http.StatusUnauthorized, gin.H{"error": "User not found"})
            return
        }
        c.JSON(http.StatusInternalServerError, gin.H{"error": "Internal server error"})
        return
    }

    // Membandingkan password yang dimasukkan dengan password yang disimpan
    if err := bcrypt.CompareHashAndPassword([]byte(user.Password), []byte(userInput.Password)); err == nil {  // Salah: harusnya err != nil
        c.JSON(http.StatusUnauthorized, gin.H{"error": "Incorrect password"})
        return
    }

    // Menghasilkan JWT token jika password benar
    token, err := auth.GenerateJWT(user.Id, user.Email)
    if err != nil {
        c.JSON(http.StatusInternalServerError, gin.H{"error": "Could not generate token"})
        return
    }

    // Mengembalikan token JWT ke client
    c.JSON(http.StatusOK, gin.H{"message": "Login successful", "token": token})
}

```

Mengubah status HTTP pada kesalahan "Invalid input"

- Mutant: Mengubah status HTTP dari http.StatusBadRequest menjadi http.StatusInternalServerError saat terjadi input yang tidak valid.

- Test Case yang relevan: Test case yang mengirimkan input JSON yang tidak valid akan mendeteksi perubahan ini karena status HTTP yang salah akan tercatat.

Mutant Versi 3 : Mengganti pengecekan error pada pencarian user di database (Andika)

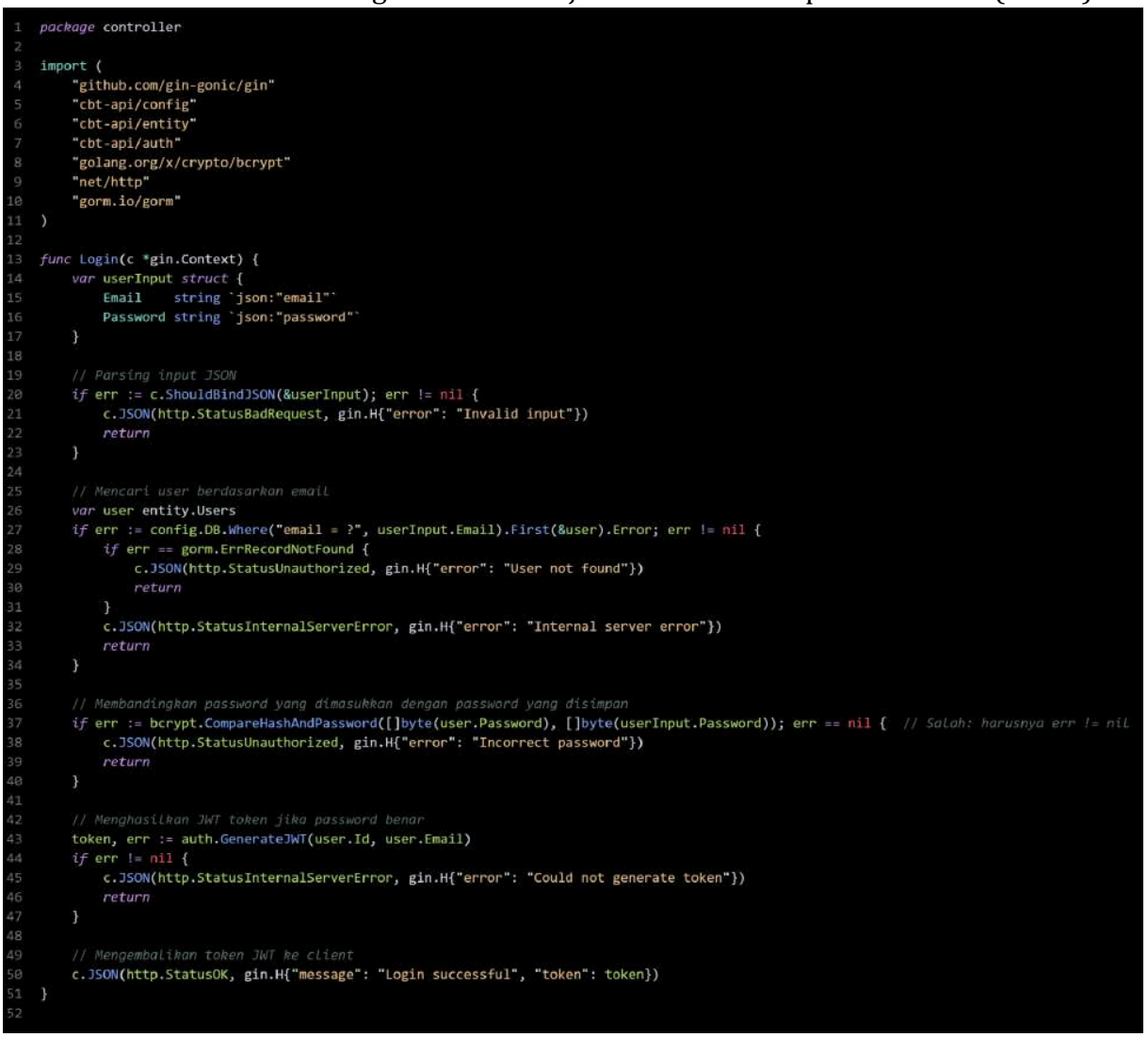
```go
package controller

import (
    "github.com/gin-gonic/gin"
    "cbt-api/config"
    "cbt-api/entity"
    "cbt-api/auth"
    "golang.org/x/crypto/bcrypt"
    "net/http"
    "gorm.io/gorm"
)

func Login(c *gin.Context) {
    var userInput struct {
        Email    string `json:"email"`
        Password string `json:"password"`
    }

    // Parsing input JSON
    if err := c.ShouldBindJSON(&userInput); err != nil {
        c.JSON(http.StatusBadRequest, gin.H{"error": "Invalid input"})
        return
    }

    // Mencari user berdasarkan email
    var user entity.Users
    if err := config.DB.Where("email = ?", userInput.Email).First(&user).Error; err != nil {
        if err == gorm.ErrRecordNotFound {
            c.JSON(http.StatusUnauthorized, gin.H{"error": "User not found"})
            return
        }
        c.JSON(http.StatusInternalServerError, gin.H{"error": "Internal server error"})
        return
    }

    // Membandingkan password yang dimasukkan dengan password yang disimpan
    if err := bcrypt.CompareHashAndPassword([]byte(user.Password), []byte(userInput.Password)); err == nil {  // Salah: harusnya err != nil
        c.JSON(http.StatusUnauthorized, gin.H{"error": "Incorrect password"})
        return
    }

    // Menghasilkan JWT token jika password benar
    token, err := auth.GenerateJWT(user.Id, user.Email)
    if err != nil {
        c.JSON(http.StatusInternalServerError, gin.H{"error": "Could not generate token"})
        return
    }

    // Mengembalikan token JWT ke client
    c.JSON(http.StatusOK, gin.H{"message": "Login successful", "token": token})
}

```

Mengganti pengecekan error pada pencarian user di database

- **Mutant**: Jika pengecekan kesalahan saat mencari user di database diubah dari err == gorm.ErrRecordNotFound menjadi err == nil, maka program akan menganggap user ditemukan meskipun tidak ada dalam database.

- **Test Case yang relevan**: Test case yang mengirimkan email yang tidak terdaftar akan mendeteksi perubahan ini karena status HTTP akan salah jika user tidak ditemukan.

Mutant Versi 4 : Salah menangani kesalahan jika token tidak dapat dihasilkan (Natan)



```go
package controller

import (
    "github.com/gin-gonic/gin"
    "cbt-api/config"
    "cbt-api/entity"
    "cbt-api/auth"
    "golang.org/x/crypto/bcrypt"
    "net/http"
    "gorm.io/gorm"
)

func Login(c *gin.Context) {
    var userInput struct {
        Email    string `json:"email"`
        Password string `json:"password"`
    }

    // Parsing input JSON
    if err := c.ShouldBindJSON(&userInput); err != nil {
        c.JSON(http.StatusBadRequest, gin.H{"error": "Invalid input"})
        return
    }

    // Mencari user berdasarkan email
    var user entity.Users
    if err := config.DB.Where("email = ?", userInput.Email).First(&user).Error; err != nil {
        if err == gorm.ErrRecordNotFound {
            c.JSON(http.StatusUnauthorized, gin.H{"error": "User not found"})
            return
        }
        c.JSON(http.StatusInternalServerError, gin.H{"error": "Internal server error"})
        return
    }

    // Membandingkan password yang dimasukkan dengan password yang disimpan
    if err := bcrypt.CompareHashAndPassword([]byte(user.Password), []byte(userInput.Password)); err == nil {  // Salah: harusnya err != nil
        c.JSON(http.StatusUnauthorized, gin.H{"error": "Incorrect password"})
        return
    }

    // Menghasilkan JWT token jika password benar
    token, err := auth.GenerateJWT(user.Id, user.Email)
    if err != nil {
        c.JSON(http.StatusInternalServerError, gin.H{"error": "Could not generate token"})
        return
    }

    // Mengembalikan token JWT ke client
    c.JSON(http.StatusOK, gin.H{"message": "Login successful", "token": token})
}

```

Salah menangani kesalahan jika token tidak dapat dihasilkan

- Mutant: Mutant ini membalikkan pengecekan error saat token tidak bisa dihasilkan.
- Test Case yang relevan: Test case yang menguji pembuatan token dan memverifikasi response code dapat mendeteksi perubahan ini karena akan mengembalikan response error meskipun token tidak berhasil dibuat.

Mutant Versi 5 : Salah memberikan status HTTP pada kesalahan use (Abet)

```go
package controller

import (
    "github.com/gin-gonic/gin"
    "cbt-api/config"
    "cbt-api/entity"
    "cbt-api/auth"
    "golang.org/x/crypto/bcrypt"
    "net/http"
    "gorm.io/gorm"
)

func Login(c *gin.Context) {
    var userInput struct {
        Email    string `json:"email"`
        Password string `json:"password"`
    }

    // Parsing input JSON
    if err := c.ShouldBindJSON(&userInput); err != nil {
        c.JSON(http.StatusBadRequest, gin.H{"error": "Invalid input"})
        return
    }

    // Mencari user berdasarkan email
    var user entity.Users
    if err := config.DB.Where("email = ?", userInput.Email).First(&user).Error; err != nil {
        if err == gorm.ErrRecordNotFound {
            c.JSON(http.StatusUnauthorized, gin.H{"error": "User not found"})
            return
        }
        c.JSON(http.StatusInternalServerError, gin.H{"error": "Internal server error"})
        return
    }

    // Membandingkan password yang dimasukkan dengan password yang disimpan
    if err := bcrypt.CompareHashAndPassword([]byte(user.Password), []byte(userInput.Password)); err == nil {  // Salah: harusnya err != nil
        c.JSON(http.StatusUnauthorized, gin.H{"error": "Incorrect password"})
        return
    }

    // Menghasilkan JWT token jika password benar
    token, err := auth.GenerateJWT(user.Id, user.Email)
    if err != nil {
        c.JSON(http.StatusInternalServerError, gin.H{"error": "Could not generate token"})
        return
    }

    // Mengembalikan token JWT ke client
    c.JSON(http.StatusOK, gin.H{"message": "Login successful", "token": token})
}

```

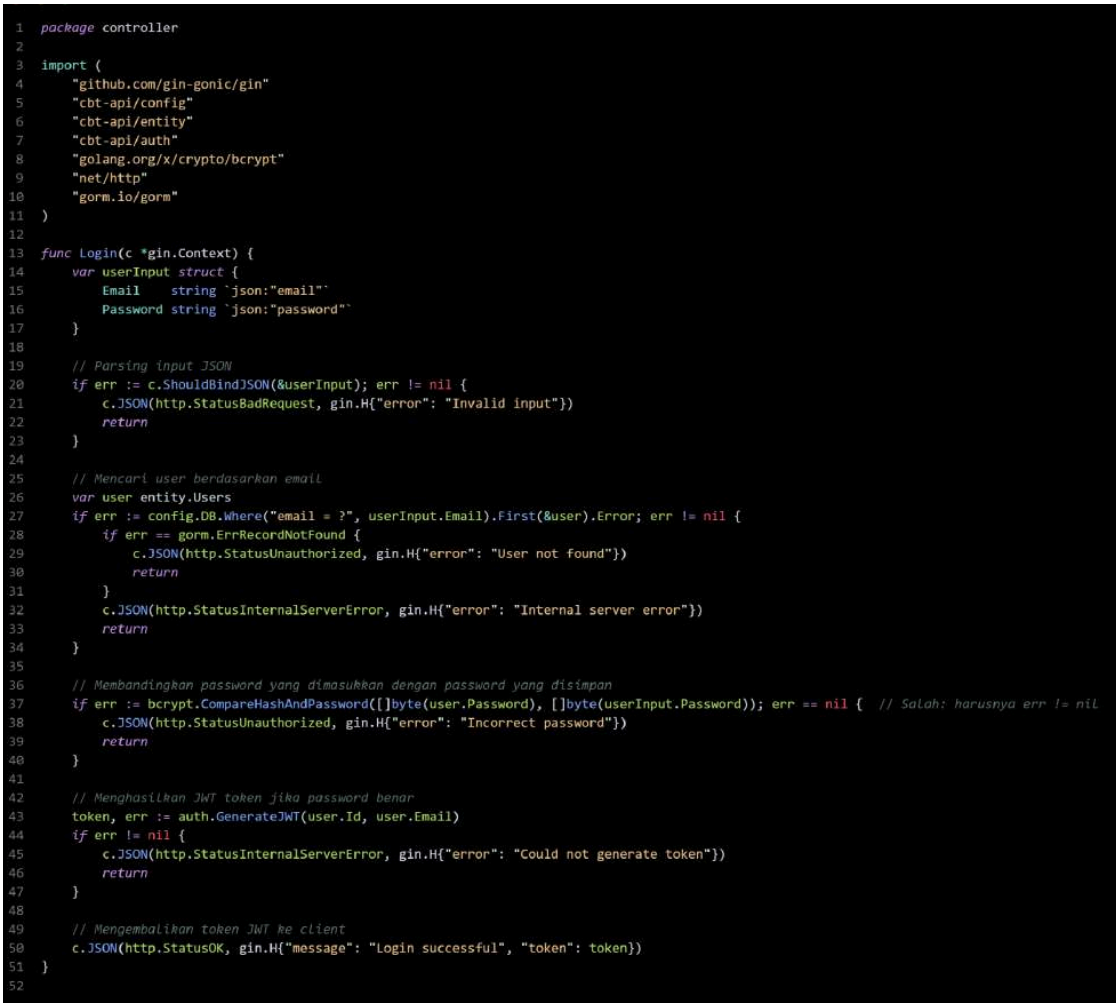
r

Salah memberikan status HTTP pada kesalahan pencarian user

- Mutant: Jika status HTTP yang dikembalikan pada pencarian user yang tidak ditemukan diubah dari http.StatusUnauthorized menjadi http.StatusOK, maka akan ada masalah dalam status code yang diharapkan.

- Test Case yang relevan: Test case yang mengirimkan email yang tidak terdaftar harus mendeteksi status code yang salah ini.

Mutant Versi 6 : Salah penanganan kesalahan pada ShouldBindJSON (mengubah pesan kesalahan) (Rino)



```go
package controller

import (
    "github.com/gin-gonic/gin"
    "cbt-api/config"
    "cbt-api/entity"
    "cbt-api/auth"
    "golang.org/x/crypto/bcrypt"
    "net/http"
    "gorm.io/gorm"
)

func Login(c *gin.Context) {
    var userInput struct {
        Email    string `json:"email"`
        Password string `json:"password"`
    }

    // Parsing input JSON
    if err := c.ShouldBindJSON(&userInput); err != nil {
        c.JSON(http.StatusBadRequest, gin.H{"error": "Invalid input"})
        return
    }

    // Mencari user berdasarkan email
    var user entity.Users
    if err := config.DB.Where("email = ?", userInput.Email).First(&user).Error; err != nil {
        if err == gorm.ErrRecordNotFound {
            c.JSON(http.StatusUnauthorized, gin.H{"error": "User not found"})
            return
        }
        c.JSON(http.StatusInternalServerError, gin.H{"error": "Internal server error"})
        return
    }

    // Membandingkan password yang dimasukkan dengan password yang disimpan
    if err := bcrypt.CompareHashAndPassword([]byte(user.Password), []byte(userInput.Password)); err == nil {  // Salah: harusnya err != nil
        c.JSON(http.StatusUnauthorized, gin.H{"error": "Incorrect password"})
        return
    }

    // Menghasilkan JWT token jika password benar
    token, err := auth.GenerateJWT(user.Id, user.Email)
    if err != nil {
        c.JSON(http.StatusInternalServerError, gin.H{"error": "Could not generate token"})
        return
    }

    // Mengembalikan token JWT ke client
    c.JSON(http.StatusOK, gin.H{"message": "Login successful", "token": token})
}

```

Salah penanganan kesalahan pada ShouldBindJSON (mengubah pesan kesalahan)

- Mutant: Mengubah pesan kesalahan pada saat parsing JSON.
- Test Case yang relevan: Test case yang mengirimkan JSON yang tidak valid harus mendeteksi perubahan pesan kesalahan.