LAPORAN PRAKTIKUM PEMROGRAMAN 2

"*Encapsulation*"

Dosen Pengampu: Tri Hadiah Muliawati, S.ST., M.Kom.

Disusun Oleh Kelompok 5

Hanadia Navaz Kamalia	**(3322600007)**
Anita Damayanti	**(3322600013)**
Bayu Kurniawan	**(3322600019)**
Abdul Muffid	**(3322600021)**
Eky Fernanda Setyawan P.	**(3322600025)**

PROGRAM STUDI D4 SAINS DATA TERAPAN

DEPARTEMEN TEKNIK INFORMATIKA DAN KOMPUTER

POLITEKNIK ELEKTRONIKA NEGERI SURABAYA

MEI 2023

Exercise 1

- Buatlah *class* ***Restaurant*** dengan dua atribut, yakni ***name*** dan ***cuisine_type***.
- Lengkapi *class* tersebut dengan *method* ***describe_restaurant()*** yang berfungsi untuk mencetak informasi terkait dua atribut di atas.
- Lengkapi pula *class* tersebut dengan *method* ***open_restaurant()*** yang mencetak pesan ke layar bahwa restoran telah dibuka.
- Buatlah 2 obyek dari *class* ***Restaurant***
- Lakukan pemanggilan *method* ***describe_restaurant()*** untuk semua obyek yang sudah dibuat.

```
class Restaurant:
    def __init__(self, name, cuisine_type):
        self.name = name
        self.cuisine_type = cuisine_type

    def describe_restaurant(self):
        print(f"{self.name} is a {self.cuisine_type} restaurant.")

    def open_restaurant(self):
        print(f"{self.name} is now open!")
```

funtion kosong bisa diisi pass

```
restaurant1 = Restaurant("Primarasa", "Indonesian")
restaurant1.describe_restaurant()
restaurant1.open_restaurant()
print()
restaurant2 = Restaurant("Mie Gacoan", "Indonesian")
restaurant2.describe_restaurant()
restaurant2.open_restaurant()
```

discribe_restorant(restorant1)

```
Primarasa is a Indonesian restaurant.
Primarasa is now open!

Mie Gacoan is a Indonesian restaurant.
Mie Gacoan is now open!
```

Analysis: Kode di atas mendefinisikan *class* ***Restaurant*** yang memiliki atribut ***name*** dan ***cuisine_type***. Objek dari *class* di atas adalah ***restaurant1*** (**Primarasa**) dan ***restaurant2*** (**Mie Gacoan**). *Method* ***describe_restaurant*** berfungsi untuk mencetak informasi tentang restoran, sementara *method* ***open_restaurant*** mencetak informasi bahwa restoran telah dibuka. Pada *output*, informasi tentang nama restoran dan jenis masakannya dicetak dengan menggunakan *method* ***describe_restaurant***, diikuti dengan informasi bahwa restoran telah dibuka menggunakan *method* ***open_restaurant***.

Exercise 2

- Tambahkan atribut menu pada *class* ***Restaurant*** untuk menyimpan daftar menu dan harga menu yang ada di restoran tersebut.
- Tambahkan *method* ***display_menu()*** untuk menampilkan daftar menu yang tersedia pada restoran.
- Lakukan penambahan menu untuk masing-masing obyek restoran yang sudah dibuat.

```
class Restaurant:
    def __init__(self, name, cuisine_type):
        self.name = name
        self.cuisine_type = cuisine_type
        self.menu = {}

    def describe_restaurant(self):
        print(f"{self.name} is a {self.cuisine_type} restaurant.")

    def open_restaurant(self):
        print(f"{self.name} is now open!")

    def display_menu(self):
        print(f"Menu at {self.name}:")
        for item, price in self.menu.items():
            print(f"{item}\t- Rp{price}")
        print()

restaurant1 = Restaurant("Primarasa", "Indonesian")
restaurant2 = Restaurant("Mie Gacoan", "Indonesia")

restaurant1.menu["Ikan bakar"] = 25000
restaurant1.menu["Nasi goreng"] = 15000
restaurant1.menu["Ayam goreng"] = 20000

restaurant2.menu["Gacoan"] = 10000
restaurant2.menu["Es teh"] = 5000
restaurant2.menu["Dimsum"] = 8000
```

```
restaurant1.display_menu()
restaurant2.display_menu()
```

```
Menu at Primarasa:
Ikan bakar      - Rp25000
Nasi goreng     - Rp15000
Ayam goreng     - Rp20000

Menu at Mie Gacoan:
Gacoan  - Rp10000
Es teh  - Rp5000
Dimsum  - Rp8000
```

Analysis: Kode di atas diperbarui dengan menambahkan atribut ***menu*** pada *class* ***Restaurant*** dan *method* ***display_menu*** untuk mencetak daftar menu beserta harganya. Dua objek restoran, yaitu "**Primarasa**" dan "**Mie Gacoan**", telah dibuat dan daftar menu beserta harganya ditambahkan ke dalam atribut ***menu*** dengan menggunakan sintaks **nama_restoran.menu["nama_makanan"] = harga**. *Method* ***display_menu*** digunakan untuk melakukan iterasi pada atribut ***menu*** yang berupa *dictionary* dan mencetak nama makanan serta harga dari setiap objek *class*. Hasil *output*-nya mencetak daftar menu beserta harga dari restoran "**Primarasa**" dan "**Mie Gacoan**".

Exercise 3

- Ubah hak akses atribut menu menjadi *private* (tidak bisa diakses di luar *class*).
- Tambahkan *method* ***add_menu()*** untuk menambahkan menu makanan yang ada pada restoran.
- Tambahkan *method* ***remove_menu()*** untuk menghapus menu makanan yang ada pada restoran.

```
class Restaurant:
    def __init__(self, name, cuisine_type, menu):
        self.name = name
        self.cuisine_type = cuisine_type
        self.__menu = menu

    def describe_restaurant(self):
        print(f"{self.name} is a {self.cuisine_type} restaurant.")

    def open_restaurant(self):
        print(f"{self.name} is now open!")

    def display_menu(self):
        print(f"\nMenu at {self.name}:")
        for item, price in self.__menu.items():
            print(f"{item}\t- Rp{price}")

    def add_menu(self):
        self.display_menu()
        add_item = input("\nMasukkan menu baru\t: ")
        if add_item in self.__menu:
            print(f"Menu {add_item} sudah tersedia.")
        else:
            price = input("Masukkan harga\t\t: ")
            self.__menu[add_item] = price
            print(f"Menu {add_item} telah ditambahkan.")
        self.display_menu()
```

```
    def remove_menu(self):
        self.display_menu()
        del_item = input("\nHapus menu: ")
        if del_item in self.__menu:
            del self.__menu[del_item]
            print(f"Menu {del_item} telah dihapus.")
        else:
            print(f"{del_item} tidak ada di dalam menu.")
        self.display_menu()

menu1 = {"Ikan Bakar": 25000,
         "Nasi Goreng": 15000,
         "Ayam Goreng": 20000}
menu2 = {"Gacoan": 10000,
         "Es Teh": 5000,
         "Dimsum": 8000}

restaurant1 = Restaurant("Primarasa", "Indonesian", menu1)
restaurant2 = Restaurant("Mie Gacoan", "Indonesian", menu2)

def print_restaurant_list(restaurants):
    print("Daftar Restaurant:")
    for i, restaurant in enumerate(restaurants):
        print(f"{i+1}. {restaurant.name}")

restaurants = [restaurant1, restaurant2]

print_restaurant_list(restaurants)
```

```
while True:
    choice = int(input("\nMasukkan nomor restaurant (0 untuk keluar): "))
    if choice > 2:
        print("Anda hanya dapat memilih restoran pada daftar di atas")
        continue
    elif choice == 0:
        break
    restaurant = restaurants[choice-1]
    print(f"\nAnda telah memilih {restaurant.name}.\n")
    restaurant.describe_restaurant()
    print("1. Display menu")
    print("2. Add menu")
    print("3. Remove menu")
    action = int(input("\nChoose an action: "))
    if action == 1:
        restaurant.display_menu()
    elif action == 2:
        restaurant.add_menu()
    elif action == 3:
        restaurant.remove_menu()
```

```
    else:
        print("Invalid input.")
```

```
Daftar Restaurant:
1. Primarasa
2. Mie Gacoan

Masukkan nomor restaurant (0 untuk keluar): 2

Anda telah memilih Mie Gacoan.

Mie Gacoan is a Indonesian restaurant.
1. Display menu
2. Add menu
3. Remove menu

Choose an action: 2

Menu at Mie Gacoan:
Gacoan  - Rp10000
Es Teh  - Rp5000
Dimsum  - Rp8000

Masukkan menu baru      : Pangsit
Masukkan harga          : 8000
Menu Pangsit telah ditambahkan.

Menu at Mie Gacoan:
Gacoan  - Rp10000
Es Teh  - Rp5000
Dimsum  - Rp8000
Pangsit - Rp8000

Masukkan nomor restaurant (0 untuk keluar): 2

Anda telah memilih Mie Gacoan.

Mie Gacoan is a Indonesian restaurant.
1. Display menu
2. Add menu
3. Remove menu

Choose an action: 3

Menu at Mie Gacoan:
Gacoan  - Rp10000
Es Teh  - Rp5000
Dimsum  - Rp8000
Pangsit - Rp8000

Hapus menu: Pangsit
Menu Pangsit telah dihapus.

Menu at Mie Gacoan:
Gacoan  - Rp10000
Es Teh  - Rp5000
Dimsum  - Rp8000

Masukkan nomor restaurant (0 untuk keluar): 0
```

Analysis: Kode di atas mendefinisikan *class* ***Restaurant*** yang memungkinkan *user* untuk mengelola menu dari restoran. Setiap objek restoran memiliki atribut ***name***, ***cuisine_type***, dan ***__menu*** (bersifat *private*). *Method* dalam kelas tersebut memungkinkan *user* untuk melihat menu (***display_menu***), menambahkan menu baru beserta harganya (***add_menu***), dan menghapus menu yang sudah ada (***remove_menu***).

Output : Pada *output*, *user* dapat memilih restoran dari daftar yang ditampilkan. Setelah memilih restoran, *user* dapat memilih aksi yang ingin dilakukan, yaitu ***Display menu***, ***Add menu***, dan ***Remove menu***. *Output* di atas menunjukkan bahwa *user* memilih restoran "**Mie Gacoan**", kemudian menambahkan menu "Pangsit" dengan harga "8000", lalu menghapus menu "Pangsit" tersebut.

Exercise 4

- Buatlah *class* ***User*** dengan atribut ***first_name***, ***last_name***, dan atribut lain yang umumnya ada pada profil *user*.
- Lengkapi *class* tersebut dengan *method* ***describe_user()*** yang berfungsi untuk mencetak informasi terkait *user*.
- Lengkapi pula *class* tersebut dengan *method* ***greet_user()*** yang mencetak sapaan dengan nama *user*.
- Buatlah 2 obyek dari *class* ***User***
- Lakukan pemanggilan *method* ***describe_user()*** dan ***greet_user()*** untuk semua obyek yang sudah dibuat.

```
class User:
    def __init__(self, first_name, last_name, username, gender, job_status):
        self.first_name = first_name
        self.last_name = last_name
        self.username = username
        self.gender = gender
        self.job_status = job_status

    def describe_user(self):
        print(f"User Info:")
        print(f"\tName: {self.first_name} {self.last_name}")
        print(f"\tUsername: {self.username}")
        print(f"\tGender: {self.gender}")
        print(f"\tJob_status: {self.job_status}")

    def greet_user(self):
        print(f"Welcome back, {self.first_name} {self.last_name}!")

user1 = User("Erion", "Keitaro", "erionkei", "Pria", "Pelajar")
user2 = User("Julia", "Jane", "jeje", "Perempuan", "Pelajar")

user1.describe_user()
user1.greet_user()
print()
user2.describe_user()
user2.greet_user()
```

```
User Info:
        Name: Erion Keitaro
        Username: erionkei
        Gender: Pria
        Job_status: Pelajar
Welcome back, Erion Keitaro!

User Info:
        Name: Julia Jane
        Username: jeje
        Gender: Perempuan
        Job_status: Pelajar
Welcome back, Julia Jane!
```

Analysis: Kode tersebut mengimplementasikan sebuah *class* ***User*** yang memiliki atribut ***first_name, last_name***, ***username***, ***gender***, ***job_status*** dan beberapa metode yang digunakan untuk berinteraksi dengan objek pengguna. Atribut ***describe_user*** digunakan untuk menampilkan user informasi. Atribut ***greet_user*** digunakan untuk menampilkan ***first_name*** dan ***last_name*** *user* yang sedang *login*. Pada kode tersebut terdapat 2 variabel *class* ***User***, variabel tersebut adalah variabel ***user1*** dan ***user2***. Variabel-variabel tersebut menggunakan *method* ***describe_user*** dan ***greet_user.***

Exercise 5

- Tambahkan atribut ***login_attempt*** pada *class* ***User*** dengan nilai *default* = 0.
- Tambahkan *method* ***increment_login_attempt()*** untuk menambahkan nilai atribut ***login_attempt*** sebanyak 1.
- Tambahkan *method* ***reset_login_attempt()*** untuk mengembalikan nilai atribut ***login_attempt*** ke 0.
- Uji coba kedua *method* di atas dan tampilkan nilai atribut ***login_attempt*** ke layar.

```
# exercise 5
class User:
    def __init__(self, first_name, last_name, username, gender, job_status):
        self.first_name = first_name
        self.last_name = last_name
        self.username = username
        self.gender = gender
        self.job_status = job_status
        self.login_attempt = 0

    def describe_user(self):
        print(f"User Info:")
        print(f"\tName: {self.first_name} {self.last_name}")
        print(f"\tUsername: {self.username}")
        print(f"\tGender: {self.gender}")
        print(f"\tJob_status: {self.job_status}")

    def greet_user(self):
        print(f"Welcome back, {self.first_name} {self.last_name}!")

    def increment_login_attempt(self):
        self.login_attempt += 1

    def reset_login_attempt(self):
        self.login_attempt = 0

user1 = User("Erion", "Keitaro", "erionkei", "Pria", "Pelajar")
user2 = User("Julia", "Jane", "jeje", "Perempuan", "Pelajar")

user1.increment_login_attempt()
user1.increment_login_attempt()
user1.increment_login_attempt()

user1.reset_login_attempt()

print(f"Login attempt: {user1.login_attempt}")
```

```
Login attempt: 0
```

Analysis: Kode tersebut mengimplementasikan sebuah *class* ***User*** yang memiliki atribut ***first_name, last_name***, ***username***, ***gender, job_status*** dan beberapa metode yang digunakan untuk berinteraksi dengan objek pengguna. Atribut ***describe_user*** digunakan untuk menampilkan user informasi. Atribut ***greet_user*** digunakan untuk menampilkan ***first_name*** dan ***last_name*** *user* yang sedang lagin. Pengguna dapat meningkatkan nilai atribut ***login_attempt*** dengan metode ***increment_login_attempt*** dan mengatur ulang menjadi 0 dengan metode ***reset_login_attempt***. Pada kode tersebut terdapat 2 variabel *class* ***User*** variabel tersebut adalah variabel ***user1*** dan ***user2***. Untuk variabel ***user1*** menggunakan method ***increment_login_attempt*** sebanyak 3x dan menggunakan *method* ***reset_login_attempt***. Terakhir ada perintah *print* yang menampilkan nilai dari atribut ***login_attempt***.

Output : "***Login_attempt:* 0**" menunjukkan bahwa nilai variabel ***login_attempt*** adalah 0. Output tersebut didapatkan karena variabel ***user1*** yang semula nilai atribut ***login_attempt*** = 0, ketika terdapat *method* ***increment_login_attempt*** sebanyak 3x maka nilai atribut ***login_attempt*** berubah dari 0 menjadi 3, dan ketika penggunaan method ***reset_login_attempt*** nilai atribut ***login_attempt*** berubah dari 3 menjadi 0.

Exercise 6

- Lakukan enkapsulasi agar *user* tidak bisa merubah nilai atribut ***login_attempt*** selain menggunakan method ***increment_login_attempt()*** dan ***reset_login_attempt()***.
- Tampilkan nilai atribut ***login_attempt*** ke layar

```
class User:
    def __init__(self, first_name, last_name, username, gender, job_status):
        self.first_name = first_name
        self.last_name = last_name
        self.username = username
        self.gender = gender
        self.job_status = job_status
        self.__login_attempt = 0

    def describe_user(self):
        print(f"User Info:")
        print(f"\tName: {self.first_name} {self.last_name}")
        print(f"\tUsername: {self.username}")
        print(f"\tGender: {self.gender}")
        print(f"\tJob_status: {self.job_status}")

    def greet_user(self):
        print(f"Welcome back, {self.first_name} {self.last_name}!")

    def increment_login_attempt(self):
        self.__login_attempt += 1

    def reset_login_attempt(self):
        self.__login_attempt = 0

    def get_login_attempt(self):
        return self.__login_attempt
user1 = User("Erion", "Keitaro", "erionkei", "Pria", "Pelajar")
user2 = User("Julia", "Jane", "jeje", "Perempuan", "Pelajar")

user1.increment_login_attempt()
user1.increment_login_attempt()
user1.increment_login_attempt()

user1.reset_login_attempt()

print(f"Login attempt: {user1.get_login_attempt()}")
```

```
Login attempt: 0
```

Analysis: Kode di atas mengimplementasikan kelas ***User*** dengan atribut ***__login_attempt*** dan beberapa metode terkait. Atribut ***__login_attempt*** melacak jumlah percobaan *login* pengguna dan dienkapsulasi menggunakan *double underscore* untuk menghindari akses langsung dari luar kelas. Metode ***increment_login_attempt*** digunakan untuk meningkatkan jumlah percobaan login, ***reset_login_attempt*** digunakan untuk mengatur ulang jumlah percobaan menjadi 0, dan ***get_login_attempt*** digunakan untuk mengakses nilai __login_attempt. Metode ***describe_user*** mencetak informasi pengguna dan **greet_user** memberikan sambutan kepada pengguna.

Output : "***Login attempt: 0***" menunjukkan bahwa setelah objek ***user1*** dibuat, atribut ***__login_attempt*** memiliki nilai awal 0. Ini menandakan bahwa pengguna belum melakukan percobaan *login* sama sekali. *Output* ini didapatkan setelah melakukan pemanggilan metode ***get_login_attempt()*** pada objek ***user1*** dan mencetak nilai atribut ***__login_attempt***. Hal ini mengindikasikan bahwa sebelum pengguna melakukan percobaan *login* pertama kali, jumlah percobaan *login* adalah 0.

Exercise 7

- Tambahkan atribut ***max_login_attempt*** pada *class* ***User*** yang bernilai sama untuk semua obyek dari *class* ***User*** yakni 3.
- Lakukan enkapsulasi pada atribut ***max_login_attempt*** sehingga atribut tersebut tidak bisa diakses secara langsung dari luar *class*.
- Modifikasi *method* ***increment_login_attempt()*** sehingga muncul notifikasi apabila *user* melakukan *login* melebihi 3 kali.
- Tambahkan *method* untuk mengubah nilai atribut ***max_login_attempt*** sehingga perubahan nilai tersebut berpengaruh pada semua obyek yang dibuat dari *class* ***User***.

```python
class User:
    __max_login_attempt = 3

    def __init__(self, first_name, last_name, username, gender, job_status):
        self.first_name = first_name
        self.last_name = last_name
        self.username = username
        self.gender = gender
        self.job_status = job_status
        self.__login_attempt = 0

    def describe_user(self):
        print(f"User Info:")
        print(f"\tName: {self.first_name} {self.last_name}")
        print(f"\tUsername: {self.username}")
        print(f"\tGender: {self.gender}")
        print(f"\tJob_status: {self.job_status}")

    def greet_user(self):
        print(f"Welcome back, {self.first_name} {self.last_name}!")

    def increment_login_attempt(self):
        if self.__login_attempt >= User.__max_login_attempt:
            print("Maaf, Anda telah melebihi batas maksimum login attempt.")
        else:
            self.__login_attempt += 1

    def reset_login_attempt(self):
        self.__login_attempt = 0

    def get_login_attempt(self):
        return self.__login_attempt

    @classmethod
    def set_max_login_attempt(cls, max_login_attempt):
        cls.__max_login_attempt = max_login_attempt
        print(f"Batas maksimum login attempt berhasil diubah menjadi {max_login_attempt}")

# contoh penggunaan kelas User
user1 = User("Erion", "Keitaro", "erionkei", "Pria", "Pelajar")

user1.describe_user()
user1.greet_user()

print(f"Jumlah login attempt saat ini: {user1.get_login_attempt()}")
user1.increment_login_attempt()
print(f"Jumlah login attempt saat ini: {user1.get_login_attempt()}")
user1.increment_login_attempt()
print(f"Jumlah login attempt saat ini: {user1.get_login_attempt()}")
user1.increment_login_attempt()
print(f"Jumlah login attempt saat ini: {user1.get_login_attempt()}")
user1.increment_login_attempt()

user1.reset_login_attempt()
print(f"Jumlah login attempt setelah direset: {user1.get_login_attempt()}")

User.set_max_login_attempt(5)
```

```
User Info:
        Name: Erion Keitaro
        Username: erionkei
        Gender: Pria
        Job_status: Pelajar
Welcome back, Erion Keitaro!
Jumlah login attempt saat ini: 0
Jumlah login attempt saat ini: 1
Jumlah login attempt saat ini: 2
Jumlah login attempt saat ini: 3
Maaf, Anda telah melebihi batas maksimum login attempt.
Jumlah login attempt setelah direset: 0
Batas maksimum login attempt berhasil diubah menjadi 5
```

Analysis: Kode tersebut mengimplementasikan kelas ***User*** dengan beberapa atribut terkait pengguna dan metode-metode yang digunakan untuk mengelola percobaan *login* pengguna. Atribut ***__max_login_attempt*** adalah atribut kelas yang menentukan batas maksimum jumlah percobaan *login* yang diizinkan. Metode ***increment_login_attempt*** diperbarui sehingga memeriksa apakah jumlah percobaan *login* telah mencapai batas maksimum sebelum meningkatkan nilainya. Jika telah melebihi batas maksimum, pesan kesalahan akan dicetak. Metode ***set_max_login_attempt*** digunakan untuk mengubah nilai atribut

Output :

- *Output* tersebut memberikan hasil dari berbagai pemanggilan metode pada objek ***user1*** dan pemanggilan metode kelas ***User***.
- *Output* pertama adalah hasil dari pemanggilan metode ***describe_user()*** pada objek ***user1***. *Output* ini menampilkan informasi pengguna seperti **nama**, ***username***, ***gender***, dan ***job_status***.
- *Output* kedua adalah hasil dari pemanggilan metode ***greet_user()*** pada objek ***user1***. *Output* ini memberikan sambutan kepada pengguna dengan menggunakan nama lengkap pengguna.
- *Output* ketiga adalah hasil dari pemanggilan metode ***get_login_attempt()*** pada objek ***user1*** setelah objek tersebut dibuat. *Output* ini menunjukkan jumlah percobaan *login* saat ini, yang pada saat itu adalah 0.
- *Output* keempat adalah hasil dari pemanggilan metode ***increment_login_attempt()*** pada objek ***user1***. Setiap pemanggilan metode ini akan meningkatkan nilai ***__login_attempt*** sebanyak satu. Oleh karena itu, *output* ini menunjukkan peningkatan jumlah percobaan *login* dari 0 menjadi 1.
- *Output* kelima adalah hasil dari pemanggilan metode ***increment_login_attempt()*** pada objek ***user1*** sekali lagi. Dalam *output* ini, jumlah percobaan *login* meningkat menjadi 2.
- *Output* keenam adalah hasil dari pemanggilan metode ***increment_login_attempt()*** pada objek ***user1*** untuk ketiga kalinya. Jumlah percobaan *login* meningkat menjadi 3.
- *Output* ketujuh adalah hasil dari pemanggilan metode ***increment_login_attempt()*** pada objek ***user1*** setelah mencapai batas maksimum ***__max_login_attempt*** yang telah diubah menjadi 5 menggunakan metode kelas ***set_max_login_attempt()***. Dalam *output* ini, karena jumlah percobaan *login* telah melebihi batas maksimum, maka pesan kesalahan dicetak.
- *Output* kedelapan adalah hasil dari pemanggilan metode ***reset_login_attempt()*** pada objek ***user1***. Metode ini mengatur ulang jumlah percobaan *login* menjadi 0. Oleh karena itu, *output* ini menunjukkan bahwa setelah direset, jumlah percobaan *login* kembali menjadi 0.
- *Output* terakhir adalah hasil dari pemanggilan metode kelas ***set_max_login_attempt()*** dengan nilai 5. Metode ini mengubah nilai batas maksimum ***__max_login_attempt***. *Output* ini mencetak pesan yang menyatakan perubahan berhasil.

Output tersebut memberikan informasi tentang interaksi dengan objek ***user1*** dan perubahan nilai atribut serta pesan kesalahan yang dapat terjadi saat pengguna melakukan percobaan *login*.
