

NAMA : Aziz Wibowo

NIM/KLS : 1900018194/D

DOSEN : rdiansyah, M.Cs

UAS : Pengantar Rekayasa Perangkat Lunak

1. Berdasarkan prinsip SOLID, jelaskan apa pengertian dan keuntungan dari penerapan prinsip berikut pada pengembangan perangkat lunak berikut

- Prinsip single responsibility

Jawab: di dalam satu class , class tersebut seharusnya hanya bertanggung jawab atas satu bagian dari fungsionalitas yang disediakan oleh perangkat lunak, dan tanggung jawab itu dienkapsulasi oleh class.

- Prinsip open/closed

Jawab: Prinsip Open/Closed menyatakan “*software entities (classes, modules, functions, etc.) should be open for extension, but closed for modification*”. Prinsip ini memaksa kita untuk dapat merancang sebuah entitas, misalnya class, yang dapat dimodifikasi perilakunya (misalnya, apa yang dikerjakan oleh sebuah method) tanpa harus mempermak *source code*-nya yang sudah ditetapkan sebelumnya. Maksudnya, sebuah entitas sebaiknya bersifat terbuka untuk diperluas, sehingga sebuah perilaku (method) dapat diubah dengan cara menambahkan kode program baru. Sedangkan untuk kode sumber yang sudah ada, sedapat mungkin tertutup atau setidaknya jarang sekali untuk perlu memodifikasi *source code* perilaku yang sudah ada.

2. Berdasarkan prinsip single responsibility pengembangan perangkat lunak, lakukanlah modifikasi program pendataan anggota baru yang telah disediakan di eLearning. Ketentuannya [Bobot 30 poin]: Data inputan berupa array yang berisi - nama: string - email: string - date of birth: string Keluaran atau outputnya adalah data anggota baru dalam bentuk json, disertai dengan usianya saat ini. Input: \$data = ['nama' => 'Zakiah', 'email' => 'zakiah@uad.ac.id', 'dob' => '23.7.1990']; Output: {"nama":"Zakiah","email":"zakiah@uad.ac.id","dob":"23.7.1990"} Usia: 30 tahun 10 bulan 15 hari

Jawab: <?php

```
use Illuminate\Support\Facades\Route;  
use PhpParser\Builder\Class_;
```

```
// use Illuminate\Http\Resources\Json;
```

```
$data =
```

```
[  
    [  
        'name' => 'aziz',  
        'email' => 'bowo.aziz12@gmail.com',  
        'Dob' => '29.9.2001'  
    ],  
    [  
        'name' => 'Luthfi',  
        'email' => 'luthfi123@gmail.com',  
        'Dob' => '1.8.2007'  
    ],  
]
```

```
[
    'name' => 'Septia',
    'email' => 'septia12@gmail.com',
    'Dob' => '11.6.2013'
],
[
    'name' => 'Hafidz',
    'email' => 'hafidz2@gmail.com',
    'Dob' => '14.9.2016'
]
];
```

```
class json
{
    public static function from($data)
    {
        return json_encode($data);
    }
}
```

```
class UserRequest
{
    protected static $rules =
    [
        'name' => 'string',
        'email' => 'string',
        'Dob' => 'string'
    ];
    public static function validate($data)
    {
        foreach(static::$rules as $property => $type)
        {
            if (gettype($data[$property]) != $type)
            {
                throw new \Exception("User property { $property } Must Be
ofType { $type }");
            }
        }
    }
}
```

```

    }
  }
}

```

```

class user

```

```

{
    public $name, $email, $Dob;

    public function __construct($data)
    {
        $this->name = $data['name'];
        $this->email = $data['email'];
        $this->Dob = $data['Dob'];
    }

    public function validate($data)
    {
        if(!isset($data['name']))
        {
            throw new \Exception("Bad request, User requires a name");
        }
        if(!isset($data['email']))
        {
            throw new \Exception("Bad request, User email required");
        }
        if(!isset($data['Dob']))
        {
            throw new \Exception("Bad request, User Date of Birthday
required");
        }
    }
}

```

```

class Age

```

```

{
    public static function now($data)

```

```

    {
      $Dob = new DateTime($data['Dob']);
      $today = new DateTime(date('d.m.y'));
      return
      [
        'year' => $today -> diff($Dob) -> y,
        'month' => $today -> diff($Dob) -> m,
        'day' => $today -> diff($Dob) -> d
      ];
    }
  }
}

Route::get('/', function(){
  $data = request()->query();

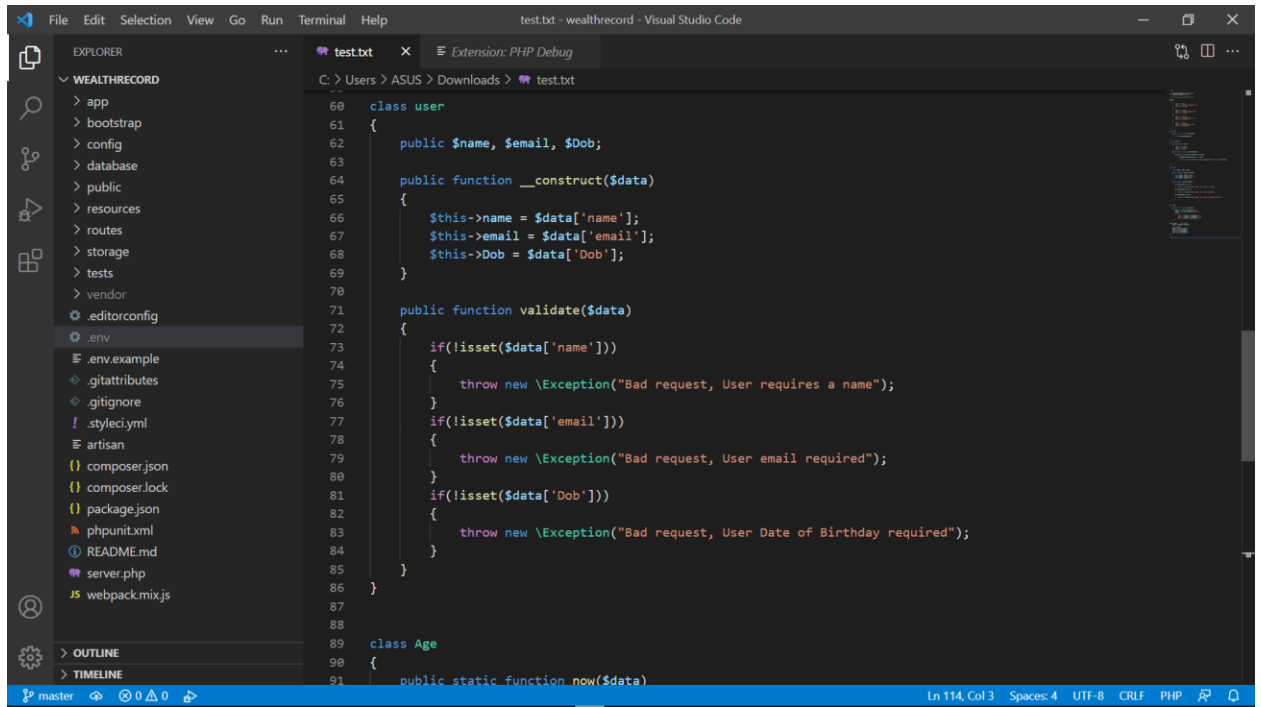
  return $data;
  $user = new User($data);
  $user->validate($data);
  print Json::from($data);
  print (Age::now($data));
})

```

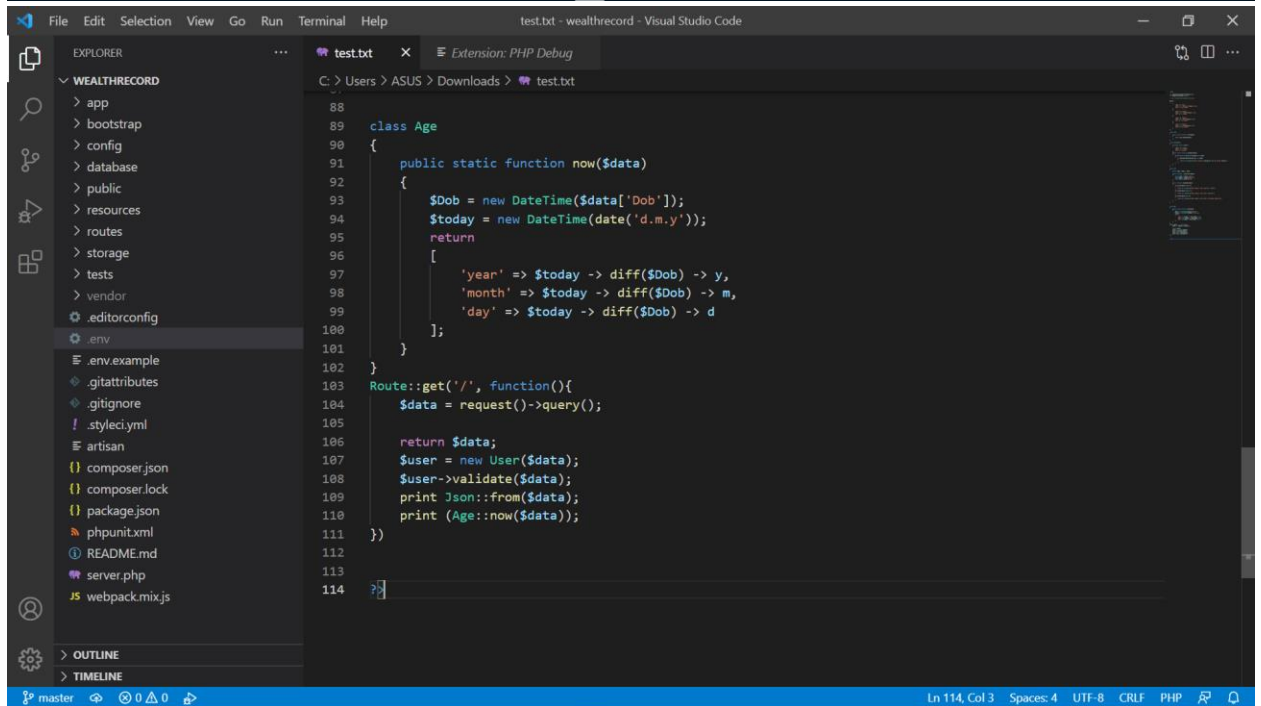
?>

```
1 1?php
2
3 use Illuminate\Support\Facades\Route;
4 use PhpParser\Builder\Class_;
5
6 // use Illuminate\Http\Resources\Json;
7
8 $data =
9 [
10     [
11         'name' => 'aziz',
12         'email' => 'bowo.azir12@gmail.com',
13         'Dob' => '29.9.2001'
14     ],
15     [
16         'name' => 'Luthfi',
17         'email' => 'luthfi123@gmail.com',
18         'Dob' => '1.8.2007'
19     ],
20     [
21         'name' => 'Septia',
22         'email' => 'septia12@gmail.com',
23         'Dob' => '11.6.2013'
24     ],
25     [
26         'name' => 'Hafidz',
27         'email' => 'hafidz2@gmail.com',
28         'Dob' => '14.9.2016'
29     ]
30 ];
31
32 class json
```

```
31
32 class json
33 {
34     public static function from($data)
35     {
36         return json_encode($data);
37     }
38 }
39
40 class UserRequest
41 {
42     protected static $rules =
43     [
44         'name' => 'string',
45         'email' => 'string',
46         'Dob' => 'string'
47     ];
48     public static function validate($data)
49     {
50         foreach(static::$rules as $property => $type)
51         {
52             if (gettype($data[$property]) != $type)
53             {
54                 throw new \Exception("User property {$property} Must Be ofType {$type}");
55             }
56         }
57     }
58 }
59
60 class user
61 {
62     public $name, $email, $Dob;
```



```
60 class user
61 {
62     public $name, $email, $Dob;
63
64     public function __construct($data)
65     {
66         $this->name = $data['name'];
67         $this->email = $data['email'];
68         $this->Dob = $data['Dob'];
69     }
70
71     public function validate($data)
72     {
73         if(!isset($data['name']))
74         {
75             throw new \Exception("Bad request, User requires a name");
76         }
77         if(!isset($data['email']))
78         {
79             throw new \Exception("Bad request, User email required");
80         }
81         if(!isset($data['Dob']))
82         {
83             throw new \Exception("Bad request, User Date of Birthday required");
84         }
85     }
86 }
87
88
89 class Age
90 {
91     public static function now($data)
```



```
100 }
101
102 }
103
104 Route::get('/', function(){
105     $data = request()->query();
106
107     return $data;
108     $user = new User($data);
109     $user->validate($data);
110     print Json::from($data);
111     print (Age::now($data));
112 })
113
114 }
```

3. Perhatikan potongan class yang terdapat di tautan ini
<https://github.com/ardiansyah-sweng/prpl2021/blob/main/payable.php>.

Pertanyaannya :

- a. Apa yang terjadi jika kita menambahkan class baru yang meng-implements interface? Bagaimana cara memanggil tiap class tersebut?

Jawab:

- kalo kita menambahkan class-class baru di dlm interface maka yang terjadi.

yang terjadi adalah: akan ada objek baru di dalam interface tersebut!

- bagaimana cara memanggil tiap class-class tersebut?

cara nya ialah : buatlah file induk yang akan di gunakan sebagai class utama.

simpan dengan suaraharimau.java

```
import javax.swing.JOptionPane;
```

```
public class suaraharimau {
```

```
public static void main (String[] args){
```

```
String suaraHewan("");
```

```
System.out.println("in Demo class.metode.data");
```

```
System.out.println(".....");
```

```
harimau pussy=new harimau();
```

```
suaraHewan= pussy.mengaum();
```



```
JOptionPane.showMessageDialog(null,"suara harimau" + "" +  
suaraHewan);
```

```
System.exit(0);} 
```

```
}
```

setelah itu buat lah harimau .java

```
class harimau {
```

```
public String mengaum(){
```

```
String Suara = new String(" mengaum ");
```

```
return Suara;
```

```
}
```

```
}
```

- b. Studi kasus: Kalkulator Bangun Ruang Ada 5 perhitungan bangun ruang yang harus disediakan yaitu: Luas
Buatlah sebuah program yang mampu menangani kebutuhan tersebut dengan menerapkan Open/Closed principle.

Jawab: <?php

```
interface Kalkulator
```

```
{  
    public function cal();  
}
```

```
class L_PersegiP implements Kalkulator
```

```
{  
    public $panjang = 10;  
    public $lebar = 5;  
    public function cal()  
    {  
        return [  
            $this->panjang * $this->lebar  
        ];  
    }  
}
```

```
class V_Bola implements Kalkulator
```

```
{  
    public $jari;  
    public $phi;  
    public function cal()  
    {  
        return [  
            (4/3) * $this->phi * $this->jari * $this->jari  
        ];  
    }  
}
```

```
class V_Kerucut implements Kalkulator
```

```

{
    public $tinggi;
    public $jari;
    public $phi;
    public function cal()
    {
        return [
            (1/3) * $this->phi * $this->jari * $this->jari * $this->jari * $this-
>tinggi
        ];
    }
}

```

```

class V_Kubus implements Kalkulator
{
    public $rusuk;
    public function cal()
    {
        return [
            $this->rusuk * $this->rusuk * $this->rusuk
        ];
    }
}

```

```

class K_Lingkaran implements Kalkulator
{
    public $jari;
    public $phi;
    public function cal()
    {
        return [
            2 * $this->phi * $this->jari
        ];
    }
}

```

```

class BangunRuangFactory

```

```

{
    public function initializeBangunRuang($type)
    {
        if ($type === 'lpersegip') {
            return new L_PersegiP();
        }
        if ($type == 'vbola') {
            return new V_Bola();
        }
        if ($type === 'vkerucut') {
            return new V_Kerucut();
        }
        if ($type === 'vkubus') {
            return new V_Kubus();
        }
        if ($type === 'klingkaran') {
            return new K_Lingkaran();
        }

        throw new Exception("Try Again Slur!!!");
    }
}

```

```

$pilihan = ['rusuk'=>12, 'tinggi'=>0, 'panjang'=>10, 'lebar'=>10,
'jari'=>21, 'phi'=>3.14];

```

```

$pilihan = 'lpersegip';
$bangunRuangFactory = new BangunRuangFactory();
$bangunRuang = $bangunRuangFactory->initializeBangunRuang($pilihan);
print_r($bangunRuang->cal());

```

```

?>

```

4. Lakukanlah deployment proyek perangkat lunak berbasis Laravel sesuai di eLearning dengan ketentuan sebagai berikut

Jawab:

<http://bjjsl.herokuapp.com/public/>

