

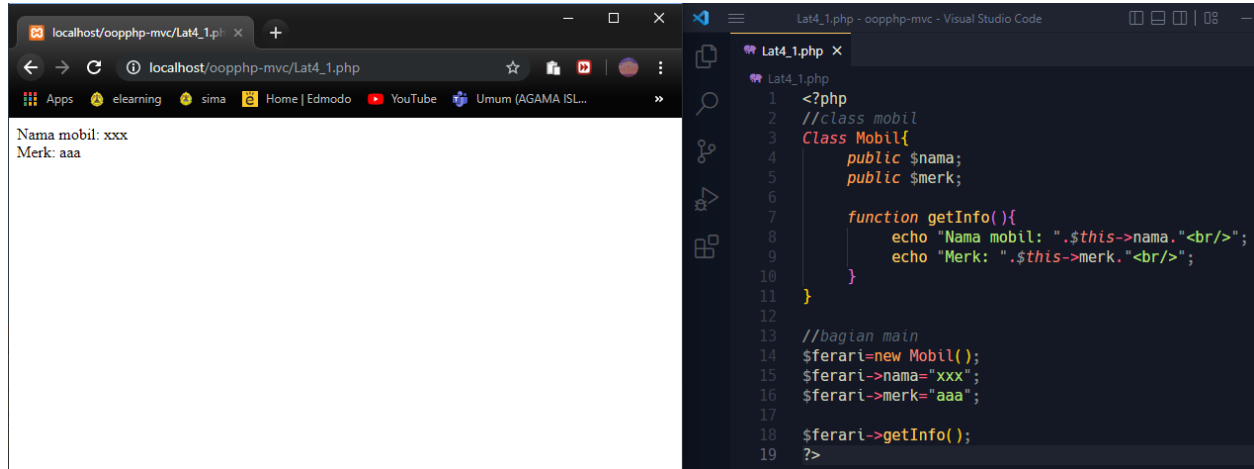
Nama : DEBI SAFA NURDEWANTI

NIM : G.211.21.0078

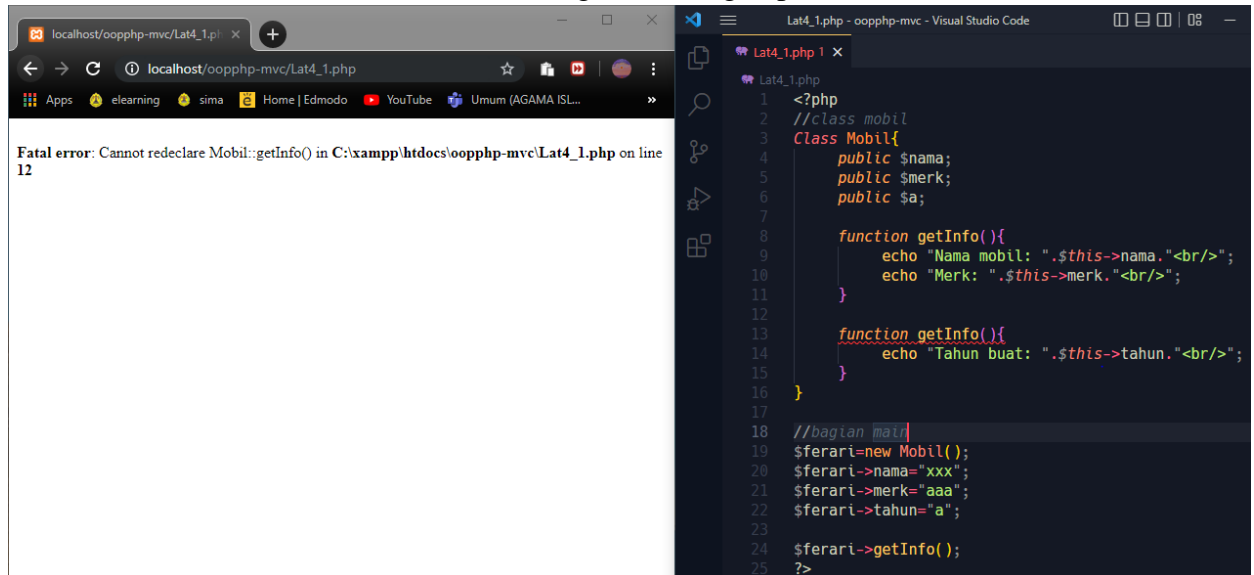
Pemrograman Framework Tugas Pertemuan 3 OOP PHP

1. Latihan 1

a. Hasil:



b. Hasil setelah ditambahkan method overload getInfo dengan parameter \$a:



Error karena pada php tidak bisa menambahkan perintah overload method.

c. Kesimpulan:

- Membuat class di php

```
<?
Class nama_class{
}
?>
```
- Menuliskan property

```
Modifier $nama_property;
```

- Menuliskan method


```
Modifier function nama_method(){
  Isi_method;
}
```
- Menginisiasi object

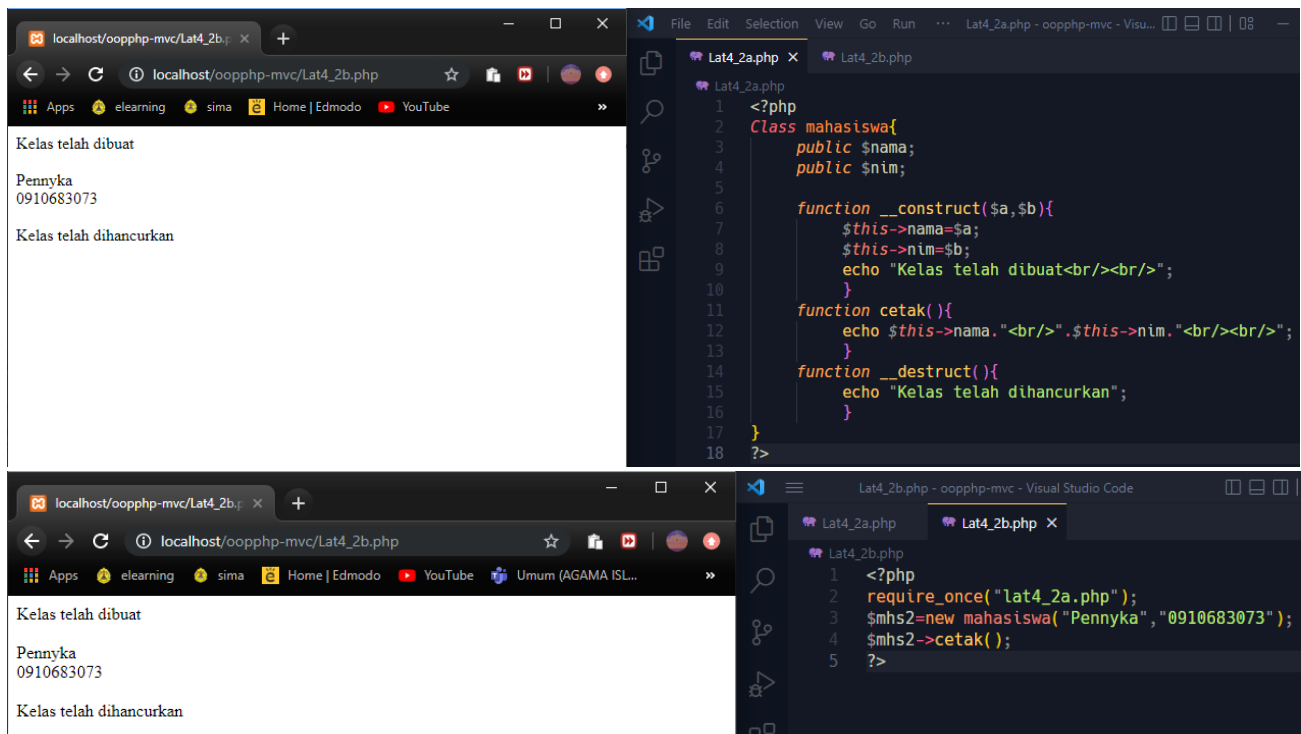

```
$nama_object=new nama_class;
```
- Mengisi atau mendefinisikan property


```
$nama_object->properties="xxx";
```
- Memanggil atau menjalankan method pada suatu class

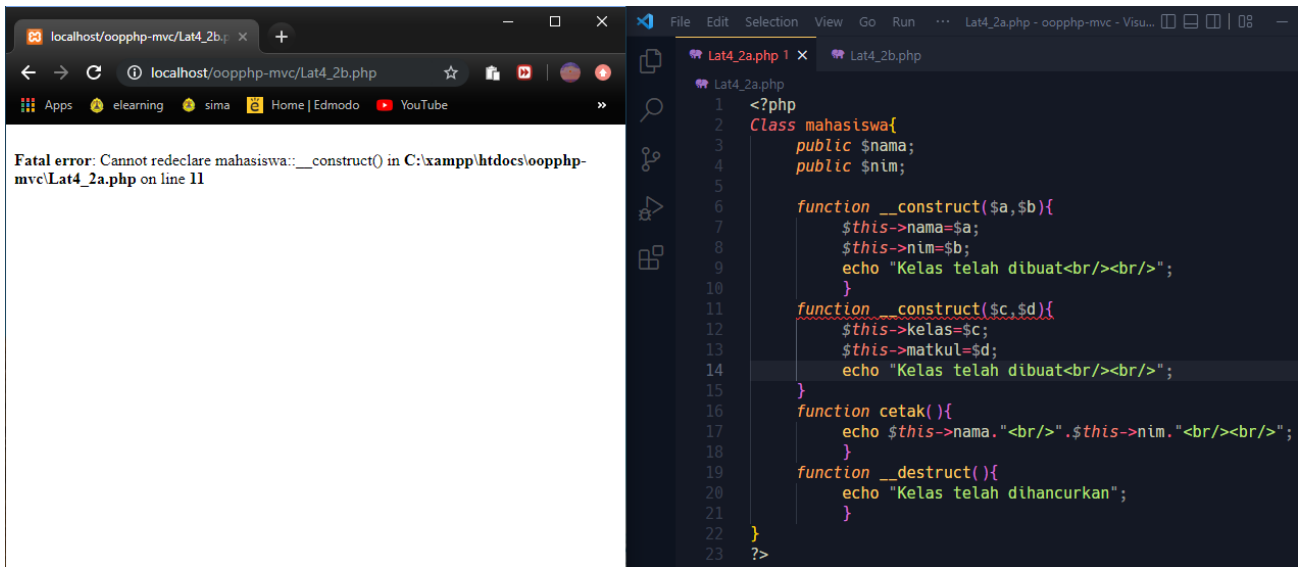

```
$nama_object->nama_method();
```

2. Latihan 2

Sebelum ditambahkan konstruktor:



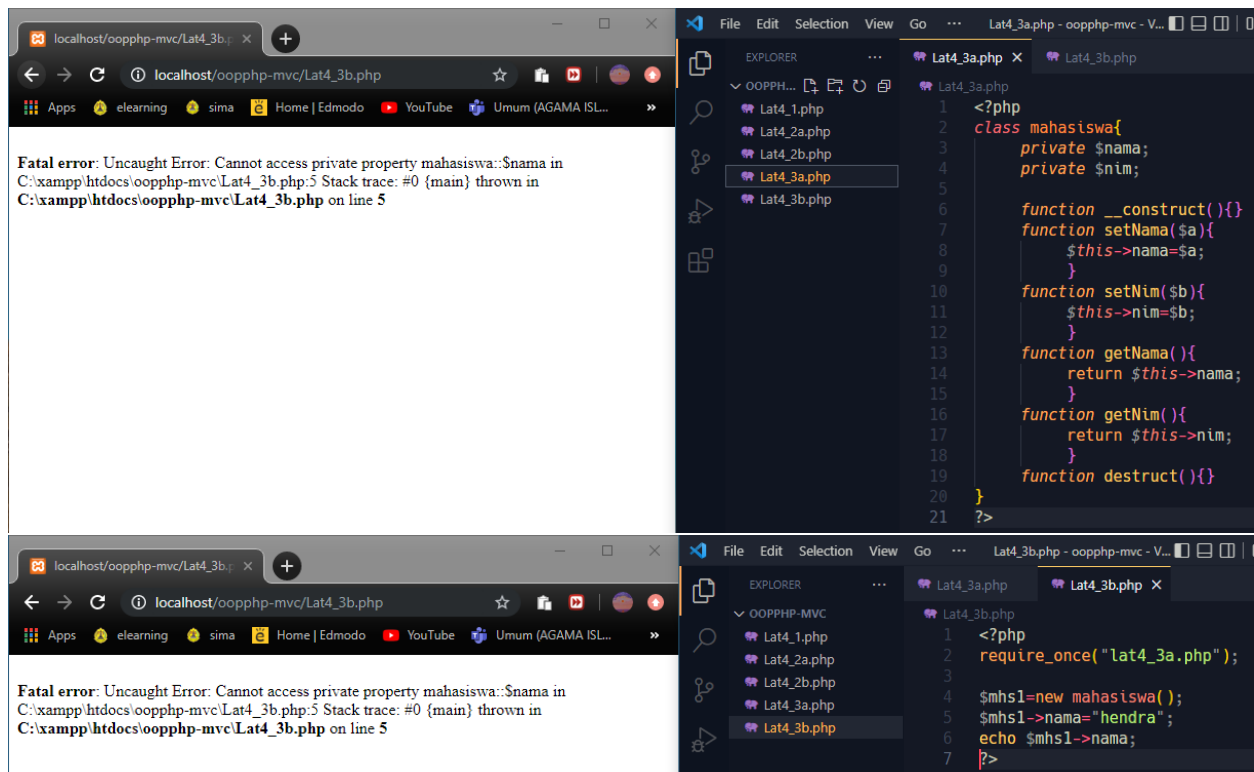
Sesudah ditambahkan konstruktor:



- Setelah ditambahkan konstruktor hasilnya menjadi error karena constructor dalam PHP OOP tidak bisa di override.

3. Latihan 3

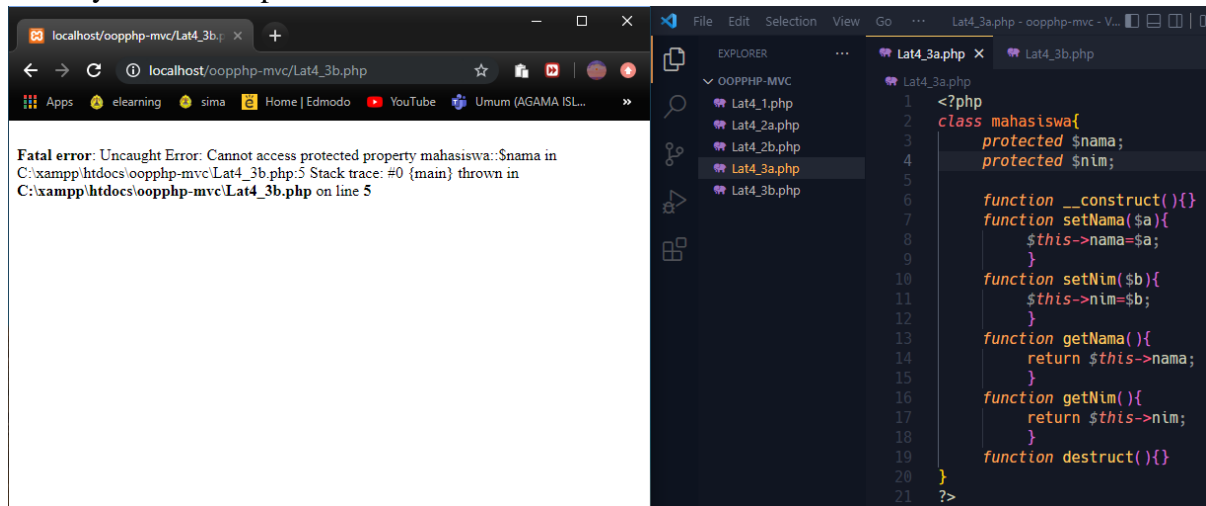
a. Hasil awal:



- Hasilnya menjadi error karena property yang bermodifier private hanya bisa digunakan pada class mahasiswa itu sendiri.
- b. Hasil setelah diubah menjadi protected dan public:

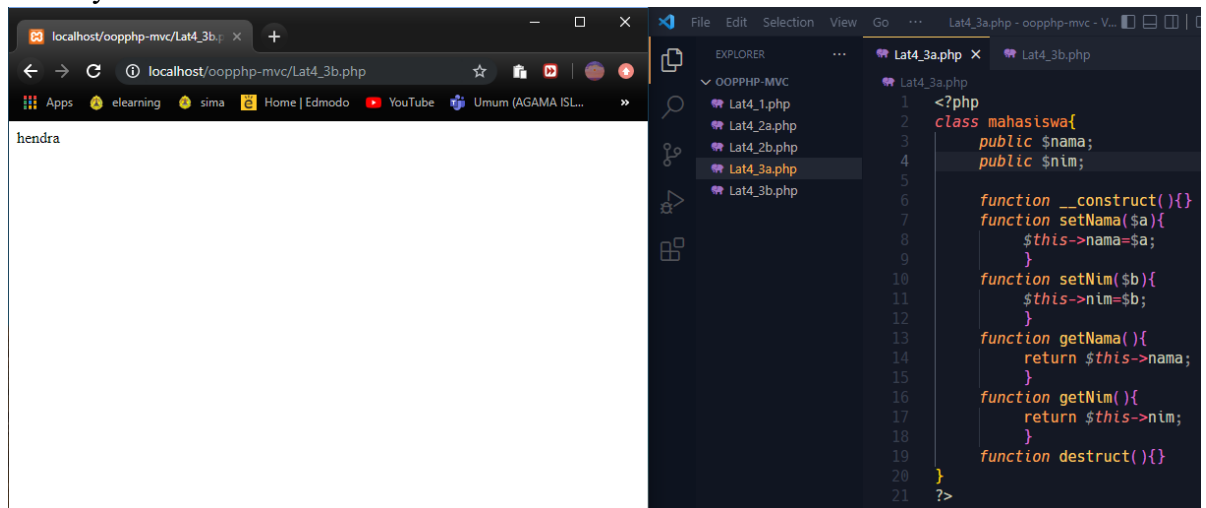
➤ Protected

Hasilnya masih tetap error



➤ Public

Hasilnya muncul tulisan “hendra”



c. Hasil dari mengubah Lat4_3b dengan modifier private:

The screenshot shows a web browser on the left and a code editor on the right. The browser displays the output of a PHP script: "Nama: Hendra" and "NIM: G.211.19.0001". The code editor shows two files: Lat4_3a.php and Lat4_3b.php. Lat4_3a.php contains a PHP class named 'mahasiswa' with private properties '\$nama' and '\$nim', and methods for setting and getting these values. Lat4_3b.php uses 'require_once' to include Lat4_3a.php and then creates an instance of the 'mahasiswa' class, setting its name and NIM, and finally echoes the values back.

```
Lat4_3a.php
1 <?php
2 class mahasiswa{
3     private $nama;
4     private $nim;
5
6     function __construct(){}
7     function setName($a){
8         $this->nama=$a;
9     }
10    function setNim($b){
11        $this->nim=$b;
12    }
13    function getName(){
14        return $this->nama;
15    }
16    function getNim(){
17        return $this->nim;
18    }
19    function destruct(){}
20 }
21 ?>
```

```
Lat4_3b.php
1 <?php
2 require_once("lat4_3a.php");
3
4 $mhs1=new mahasiswa();
5 /* sebelum diubah
6 // $mhs1->nama="hendra";
7 // echo $mhs1->nama;
8 */
9 //setelah diubah
10 $mhs1->setName("Nama: Hendra<br/>");
11 echo $mhs1->getName();
12
13 $mhs1->setNim("NIM: G.211.19.0001");
14 echo $mhs1->getNim();
15 ?>
```

```
/* sebelum diubah
// $mhs1->nama="hendra";
// echo $mhs1->nama;
*/

//setelah diubah
$mhs1->setName("Nama: Hendra<br/>");
echo $mhs1->getName();

$mhs1->setNim("NIM: G.211.19.0001");
echo $mhs1->getNim();
```

d. Kesimpulan:

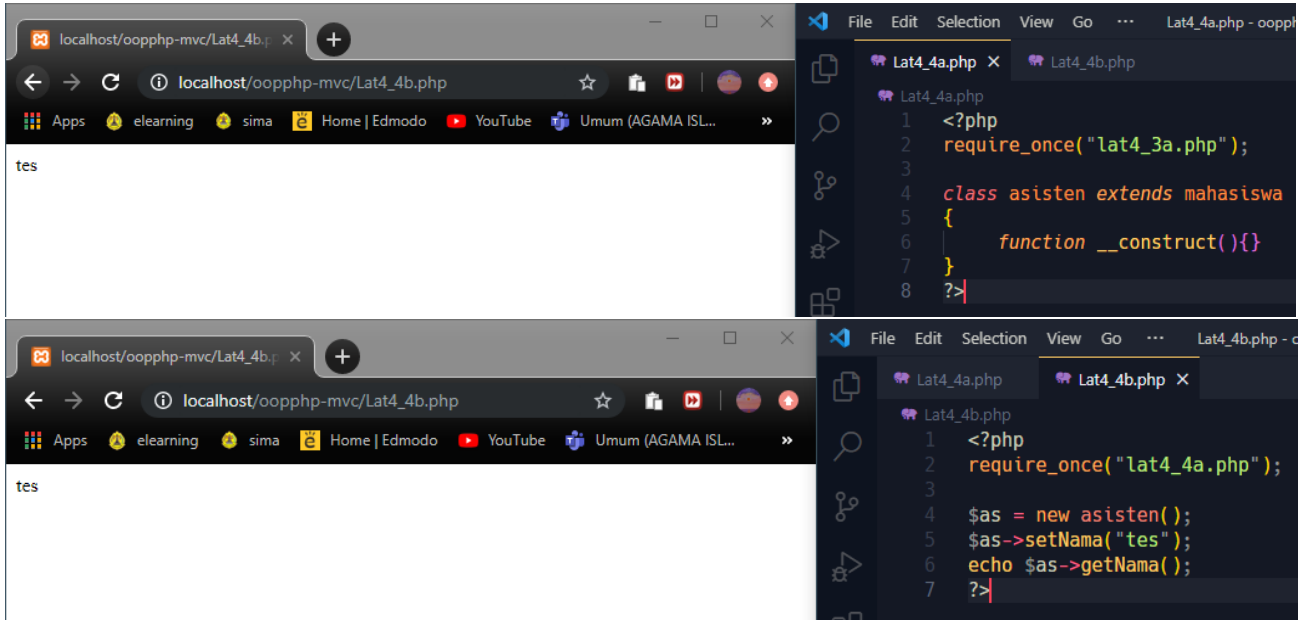
➤ Macam-macam modifier:

- 1) Public → Untuk mendefinisikan data atau metode yang akan terlihat dari luar oleh siapapun dan dimanapun.
- 2) Private → Untuk mendefinisikan data atau metode agar hanya terlihat pada class/object itu sendiri.

- 3) Protected → Untuk mendefinisikan data atau metode untuk tidak terlihat dari luar (sama dengan private), tetapi akan dapat diakses oleh sub-class dari class tersebut.
- Properties pada modifier protected dan private bisa dipanggil dengan mengimplementasikan setter dan getter.

4. Latihan 4

Hasil dari script php latihan 4:

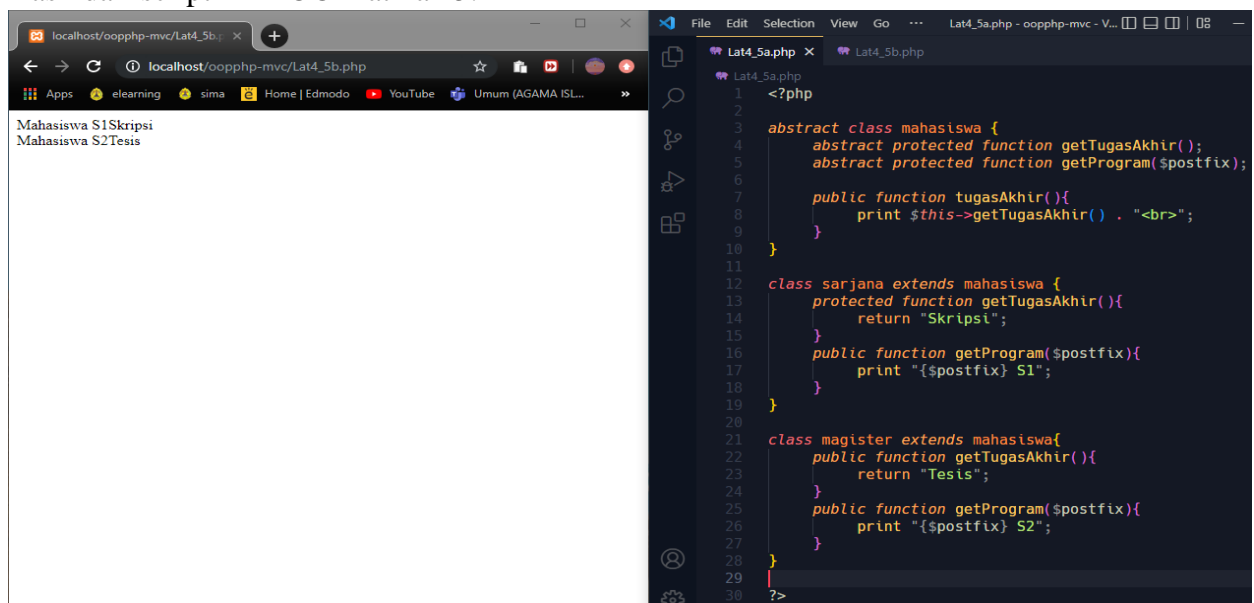


The screenshot shows a web browser window at localhost/oopphp-mvc/Lat4_4b.php displaying the output 'tes'. To the right, a code editor shows the PHP files. Lat4_4a.php defines a class 'asisten' that extends 'mahasiswa' and implements the '__construct()' method. Lat4_4b.php uses 'require_once' to include Lat4_4a.php, creates an instance of 'asisten' with the name 'tes', and echoes the name using the 'getNama()' method.

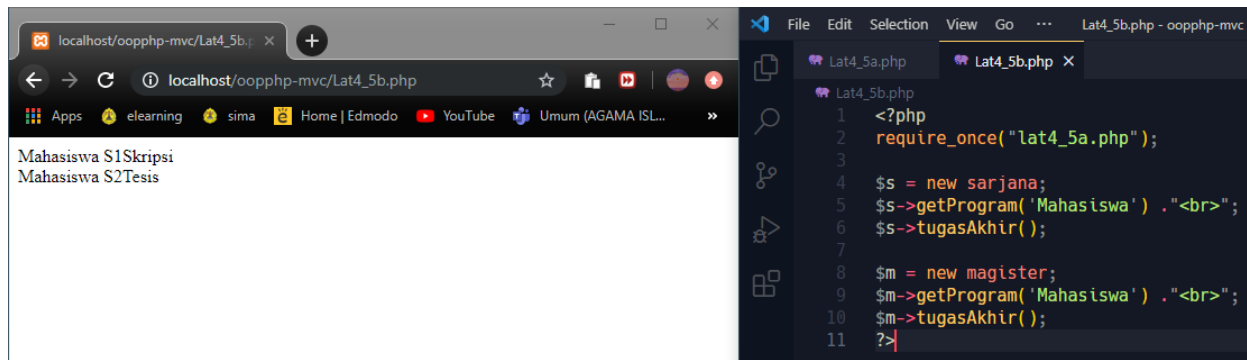
Pada PHP OOP latihan 4 ini, class asisten (sub class/child) bisa memanggil method dari mahasiswa (class/parent).

5. Latihan 5

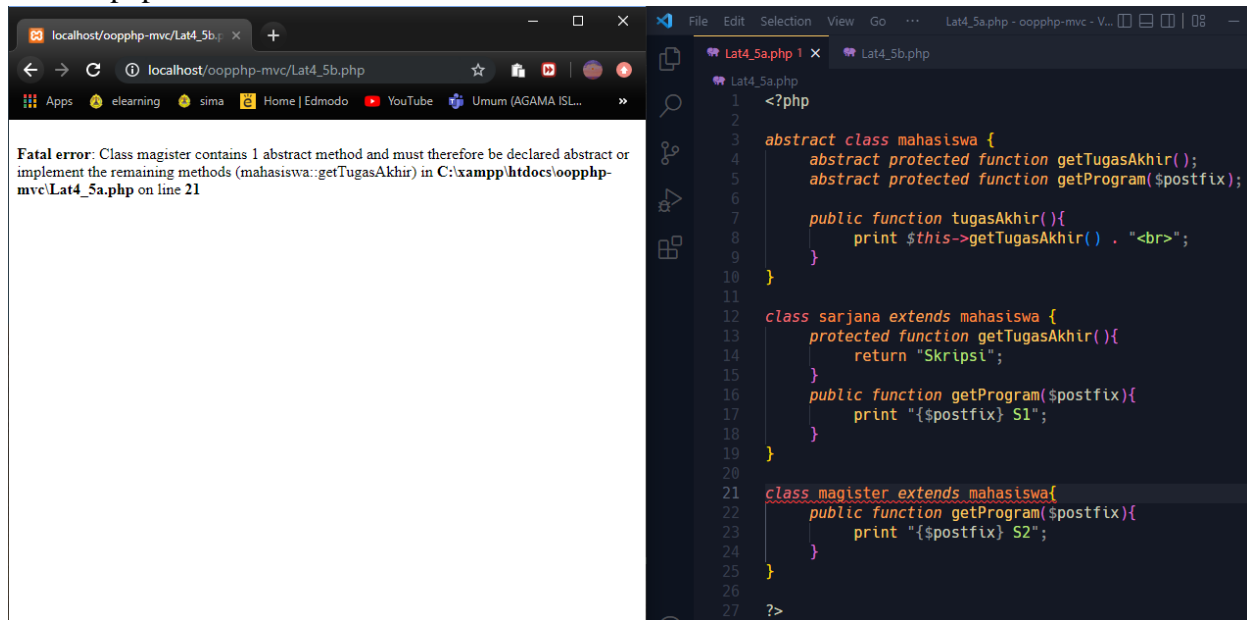
a. Hasil dari script PHP OOP latihan 5:



The screenshot shows a web browser window at localhost/oopphp-mvc/Lat4_5b.php displaying the output 'Mahasiswa S1Skripsi' and 'Mahasiswa S2Tesis'. To the right, a code editor shows the PHP files. Lat4_5a.php defines an abstract class 'mahasiswa' with abstract methods 'getTugasAkhir()' and 'getProgram(\$postfix)'. It also defines two subclasses: 'sarjana' and 'magister', both extending 'mahasiswa'. 'sarjana' implements 'getTugasAkhir()' to return 'Skripsi' and 'getProgram()' to return 'S1'. 'magister' implements 'getTugasAkhir()' to return 'Tesis' and 'getProgram()' to return 'S2'. Lat4_5b.php uses 'require_once' to include Lat4_5a.php and then creates instances of 'sarjana' and 'magister' to demonstrate the methods.



- b. Hasil dari menghapus script pada baris 29-32 (kalua pada script yang saya buat pada baris 22-24) pada `lat4_5a.php`:

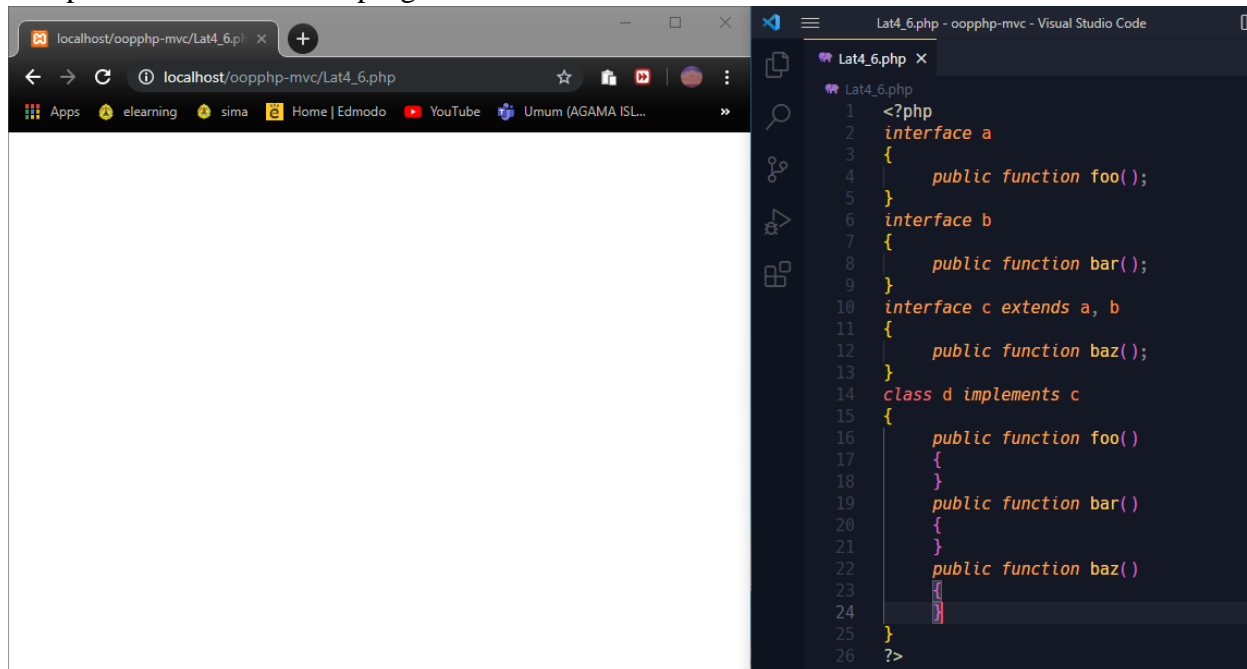


Error karena pada class child yang mewarisi parent class harus menuliskan semua method yang ada di abstrak dari parent class.

- c. Kesimpulan:
- Child class yang mewarisi super class/parent class harus menuliskan semua method yang ada pada abstrak dari super class/parent classnya.

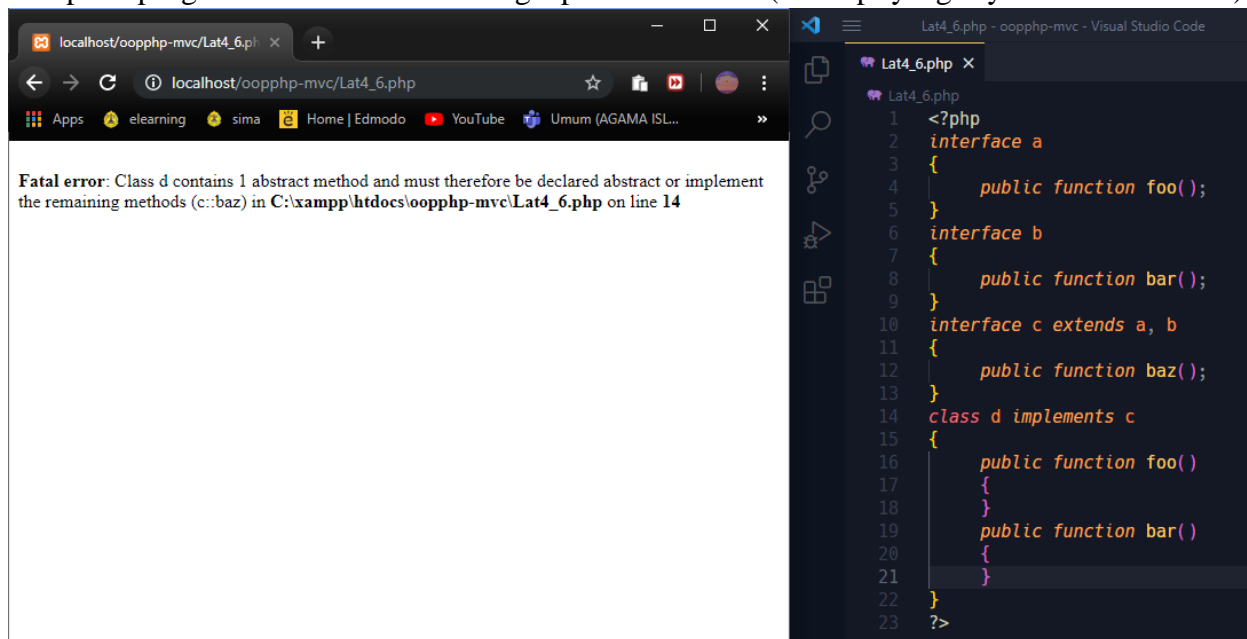
6. Latihan 6

a. Tampilan dan maksud dari program tersebut:



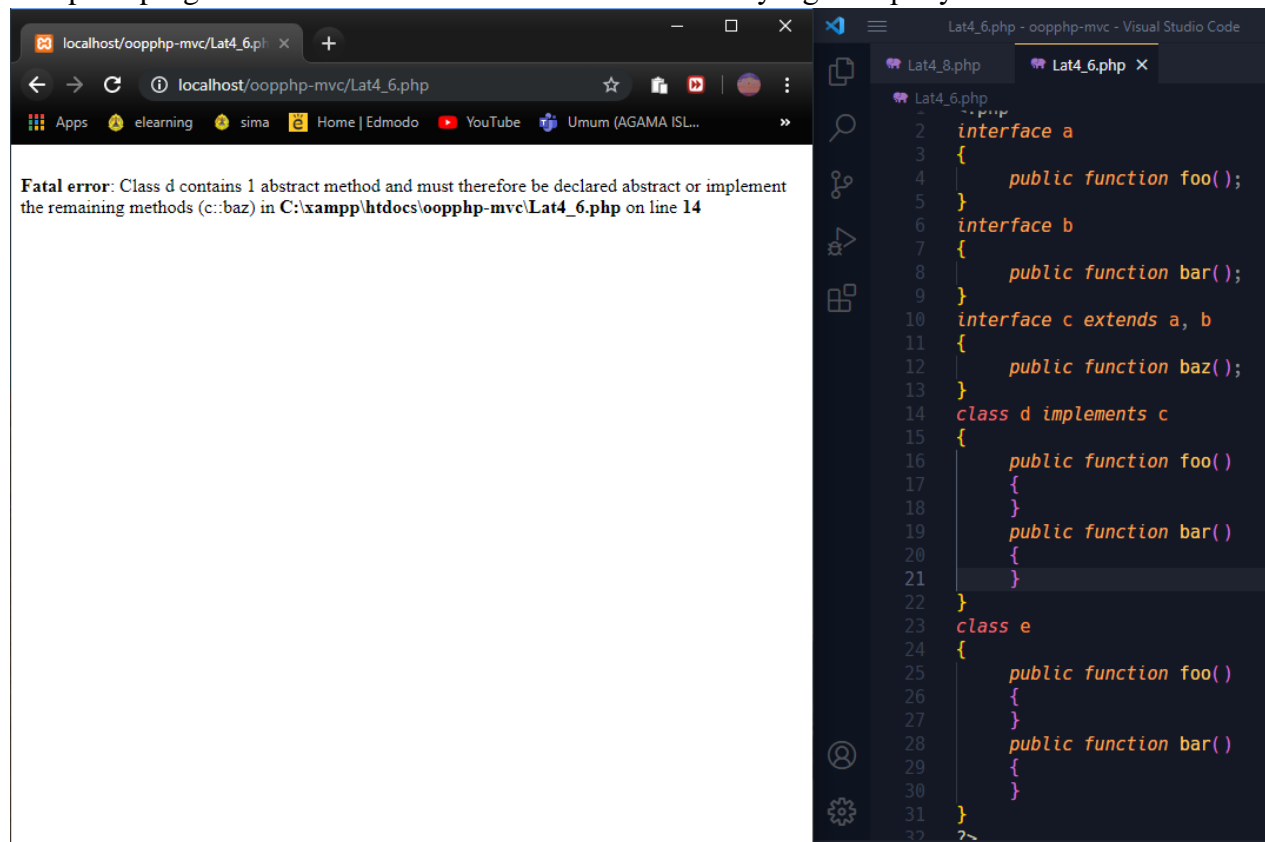
Maksud dari program tersebut adalah penggunaan object interface.

b. Tampilan program tersebut setelah menghapus baris 27-29 (di script yang saya buat baris 22-24):



Error karena pada object interface saat kita mengimplementasikan object tersebut, semua method yang ada pada interface harus diimplementasikan seluruhnya. Karena class `d` mengimplementasikan interface `c`, maka seluruh method yang terdapat pada interface `c` harus diimplementasikan.

c. Tampilan program tersebut setelah ditambahkan class “e” yang mempunyai method foo dan bar:



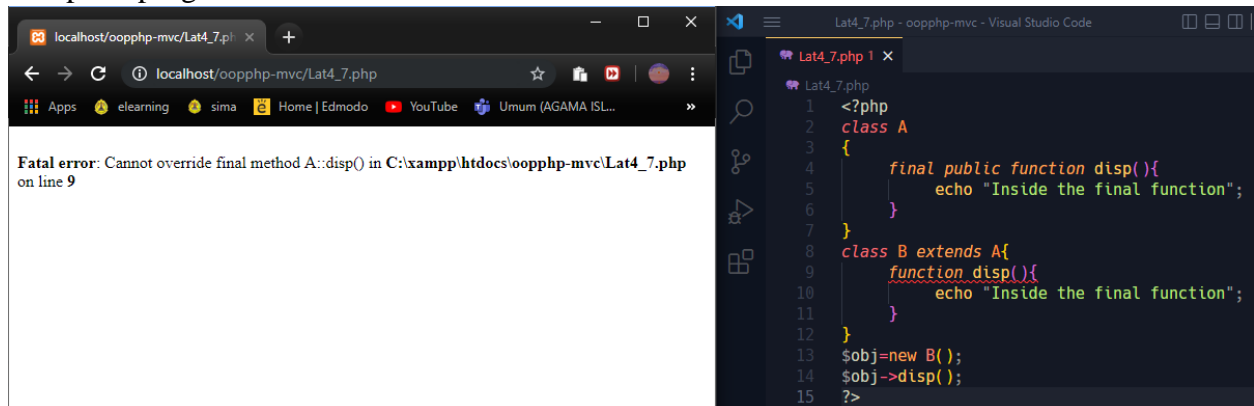
Error karena method foo dan bar terdapat pada interface a dan b yang pada penerapannya sebuah class baru tidak dapat mengimplementasikan method yang sama pada dua interface.

d. Kesimpulan:

- Interface didefinisikan dengan “Interface” keyword, mirip dengan script class biasa, namun definisi atau detail method tidak dituliskan.
- Seluruh method yang dituliskan pada interface harus memiliki modifier “public”.
- Untuk mengimplementasikan sebuah interface dapat menggunakan “implement” keyword.
- Semua method yang ada pada interface harus diimplementasikan semuanya. Sebuah class bisa mengimplementasikan lebih dari satu interface.
- Class tidak bisa mengimplementasikan dua interface yang mempunyai nama method yang sama.
- Interface dapat diwariskan seperti class menggunakan “extends”.
- Class yang mengimplementasikan interface harus menggunakan method-method yang ada pada interface tersebut dengan nama dan spesifikasi yang sama persis.

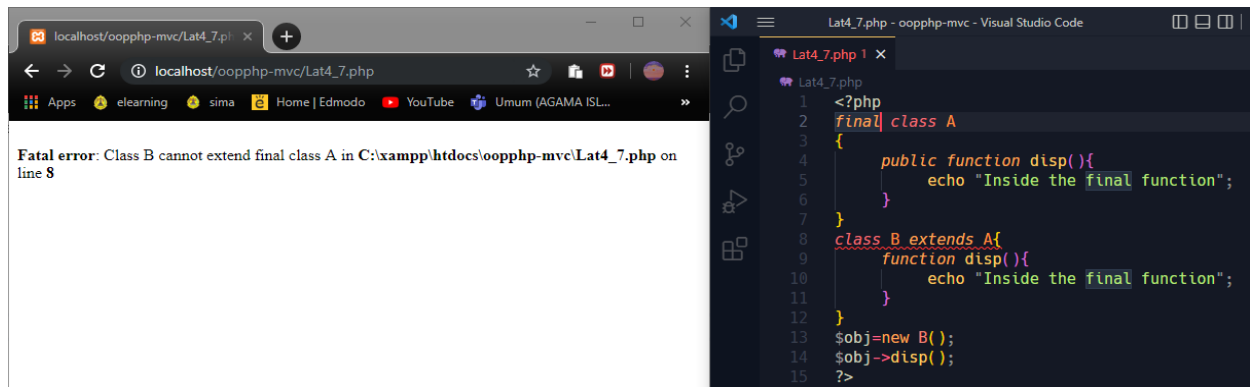
7. Latihan 7

a. Tampilan program di atas:



Error karena method pada class A terjadi override.

b. Tampilan setelah menghapus kata final pada baris 5 (di script 4) dan menambahkan kata final pada baris 2:



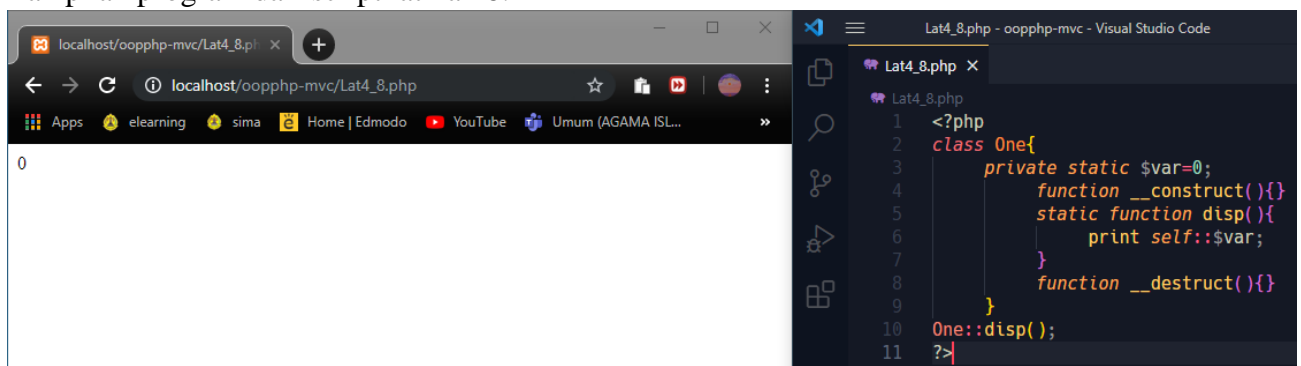
Tampilan seperti gambar di atas karena tujuan digunakan keyword final agar class tidak akan override.

c. Kesimpulan:

- Keyword “final” akan mencegah proses overriding method pada sub-class.
- Apabila method diberikan status final, maka method tersebut tidak akan bisa di override, sama juga pada class, apabila diberikan keyword “final” pada deklarasi class maka class tersebut tidak bisa diturunkan (diwariskan).

8. Latihan 8

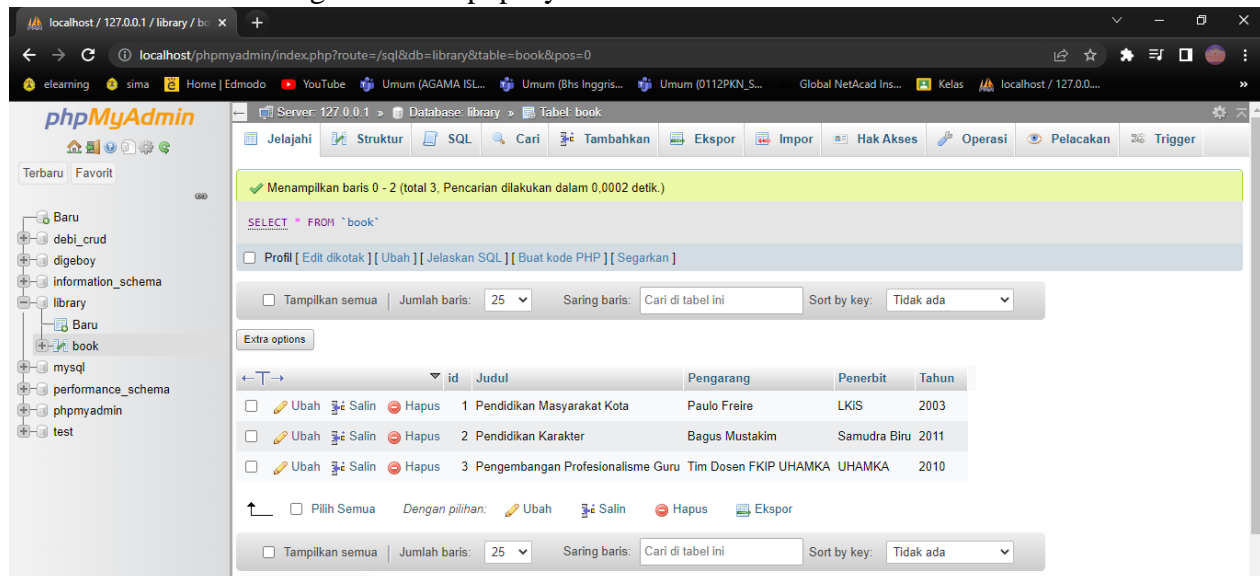
Tampilan program dari script latihan 8:



- Latihan 8 mengimplementasikan penggunaan property static dengan modifier private. Properti statis didefinisikan dengan menggunakan kata kunci **static** sebelum modifier. Sintaks:
`Modifier static $nama_property = nilai;`
- Sifat statis dapat diakses tanpa perlu sebuah contoh objek dari class, menggunakan nama kelas bersama dengan ::
Sintaks:
`Classname :: $nama_property method_name();`
 Properti statis memakai tanda dolar \$. Properti statis tidak dapat diakses melalui objek menggunakan operator tanda panah “->”
- Untuk mengakses metode statis atau property dari dalam kelas yang sama dapat menggunakan kata kunci **self**.
Sintaks:
`Self :: $ properti;`

9. Latihan 9

- Membuat Database dengan table di phpMyAdmin



The screenshot shows the phpMyAdmin web interface. The left sidebar displays a database structure with a 'library' database containing a 'book' table. The main area shows the 'book' table with the following data:

	id	Judul	Pengarang	Penerbit	Tahun
<input type="checkbox"/> Ubah <input type="checkbox"/> Salin <input type="checkbox"/> Hapus	1	Pendidikan Masyarakat Kota	Paulo Freire	LKIS	2003
<input type="checkbox"/> Ubah <input type="checkbox"/> Salin <input type="checkbox"/> Hapus	2	Pendidikan Karakter	Bagus Mustakim	Samudra Biru	2011
<input type="checkbox"/> Ubah <input type="checkbox"/> Salin <input type="checkbox"/> Hapus	3	Pengembangan Profesionalisme Guru	Tim Dosen FKIP UHAMKA	UHAMKA	2010

Below the table, there are options to 'Pilih Semua' (Select All), 'Ubah' (Edit), 'Salin' (Copy), 'Hapus' (Delete), and 'Ekspor' (Export). The interface also includes a search bar and a 'Tampilkan semua' (Show all) checkbox.

```

10 SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
11 START TRANSACTION;
12 SET time_zone = "+00:00";
13
14
15 /*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
16 /*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
17 /*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
18 /*!40101 SET NAMES utf8mb4 */;
19
20 --
21 -- Database: `library`
22 --
23
24 -- -----
25
26 --
27 -- Struktur dari tabel `book`
28 --

```

book.sql

C:\> Users\> User\> Downloads\> book.sql

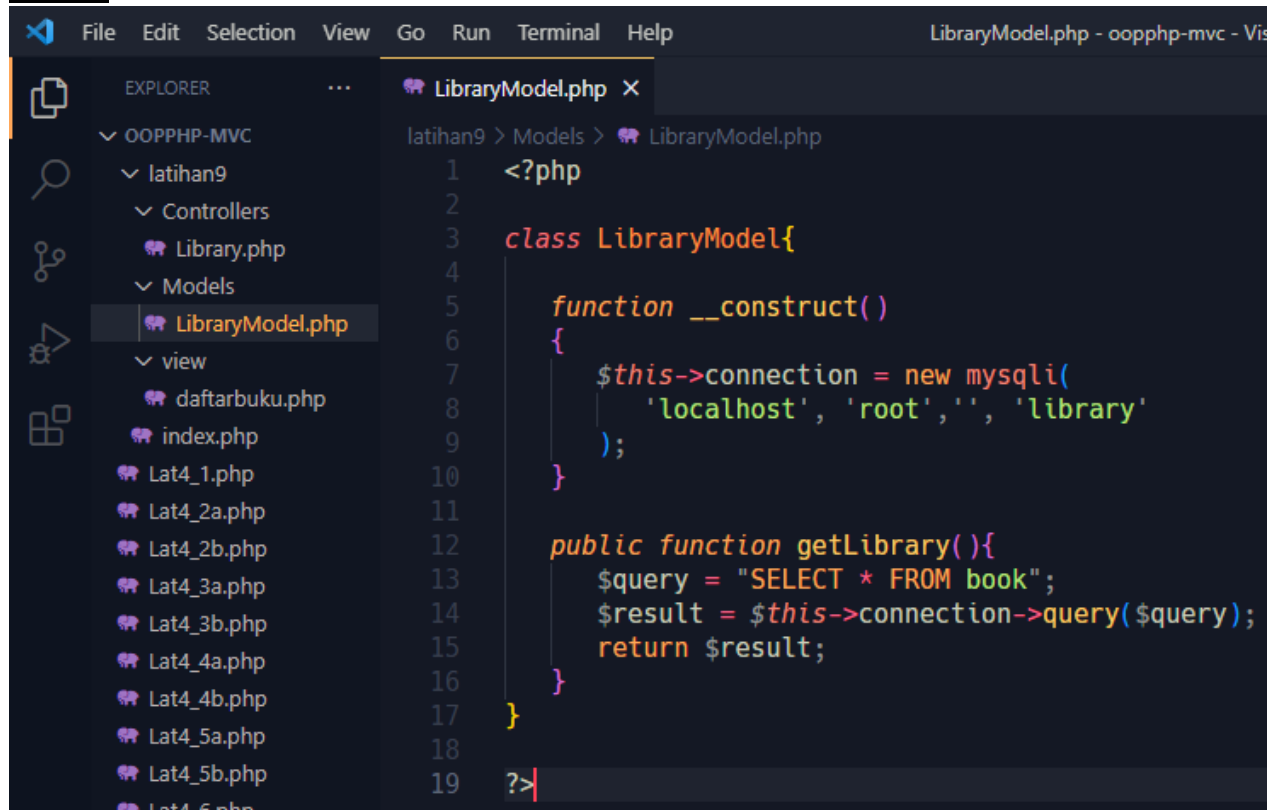
```

30 CREATE TABLE `book` (
31   `id` int(11) NOT NULL,
32   `Judul` varchar(40) NOT NULL,
33   `Pengarang` varchar(30) NOT NULL,
34   `Penerbit` varchar(25) NOT NULL,
35   `Tahun` varchar(10) NOT NULL
36 ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
37
38 --
39 -- Dumping data untuk tabel `book`
40 --
41
42 INSERT INTO `book` (`id`, `Judul`, `Pengarang`, `Penerbit`, `Tahun`) VALUES
43 (1, 'Pendidikan Masyarakat Kota', 'Paulo Freire', 'LKIS', '2003'),
44 (2, 'Pendidikan Karakter', 'Bagus Mustakim', 'Samudra Biru', '2011'),
45 (3, 'Pengembangan Profesionalisme Guru', 'Tim Dosen FKIP UHAMKA', 'UHAMKA', '2010');
46
47 --
48 -- Indexes for dumped tables
49 --
50
51 --
52 -- Indeks untuk tabel `book`
53 --
54 ALTER TABLE `book`
55   ADD PRIMARY KEY (`id`);
56 COMMIT;
57

```

- Membuat file Models, View, dan Controllers

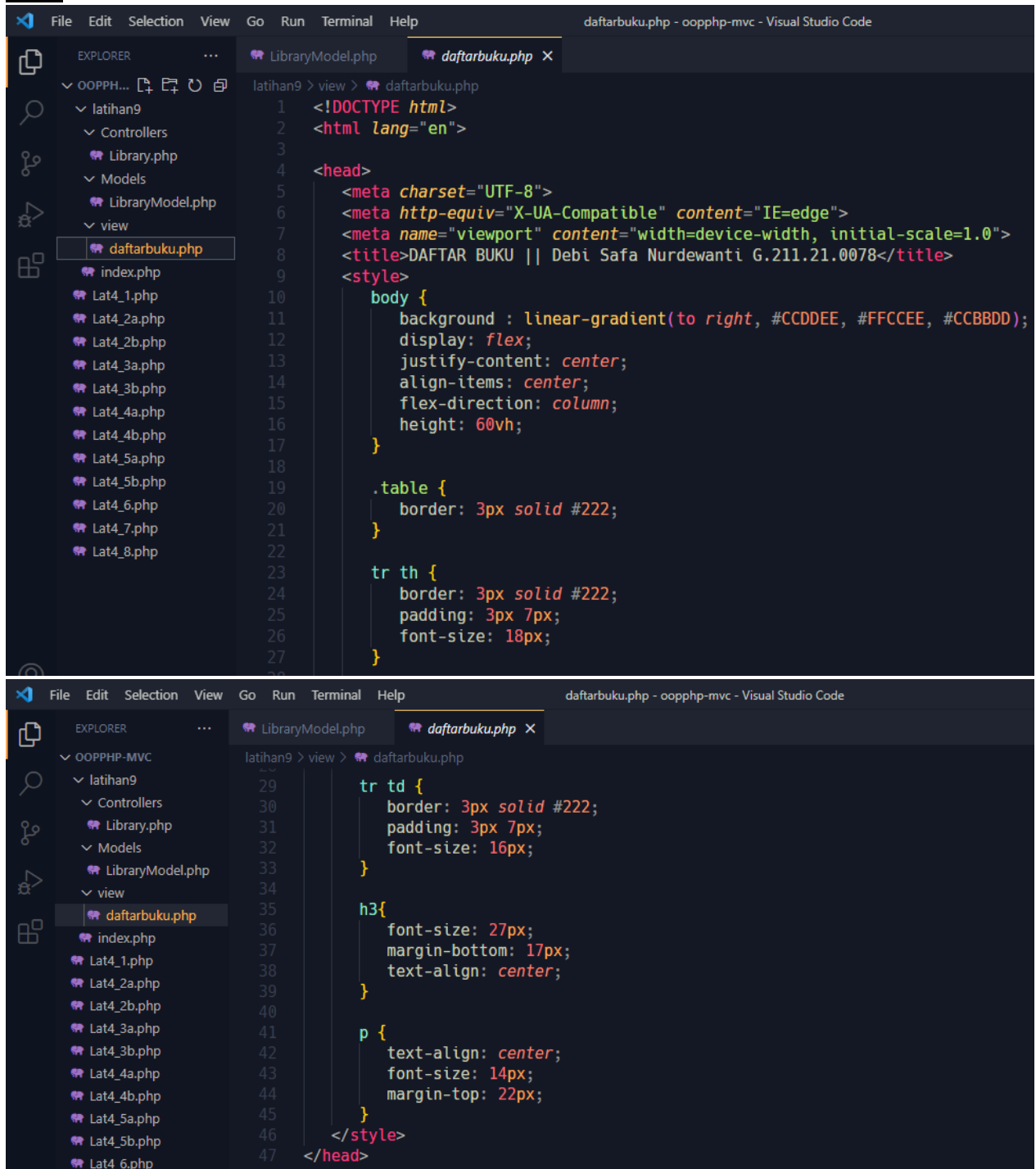
Models



The screenshot shows the Visual Studio Code interface with a project named 'OOPPHP-MVC'. The Explorer sidebar on the left shows the file structure: 'latihan9' contains 'Controllers' (with 'Library.php'), 'Models' (with 'LibraryModel.php'), and 'view' (with 'daftarbuku.php', 'index.php', and several 'Lat4' files). The main editor window displays the code for 'LibraryModel.php'.

```
1 <?php
2
3 class LibraryModel{
4
5     function __construct()
6     {
7         $this->connection = new mysqli(
8             'localhost', 'root','', 'library'
9         );
10    }
11
12    public function getLibrary(){
13        $query = "SELECT * FROM book";
14        $result = $this->connection->query($query);
15        return $result;
16    }
17 }
18
19 ?>
```

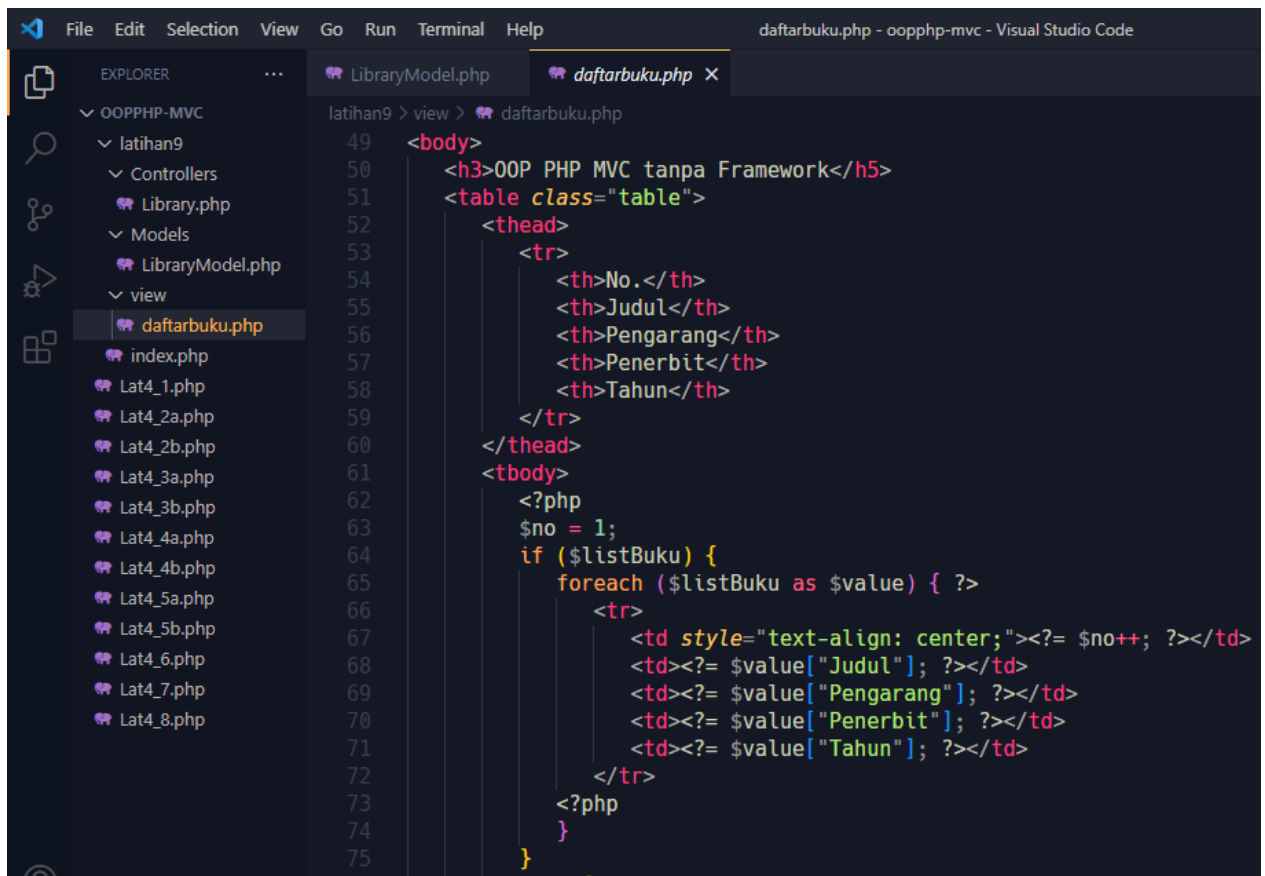
View



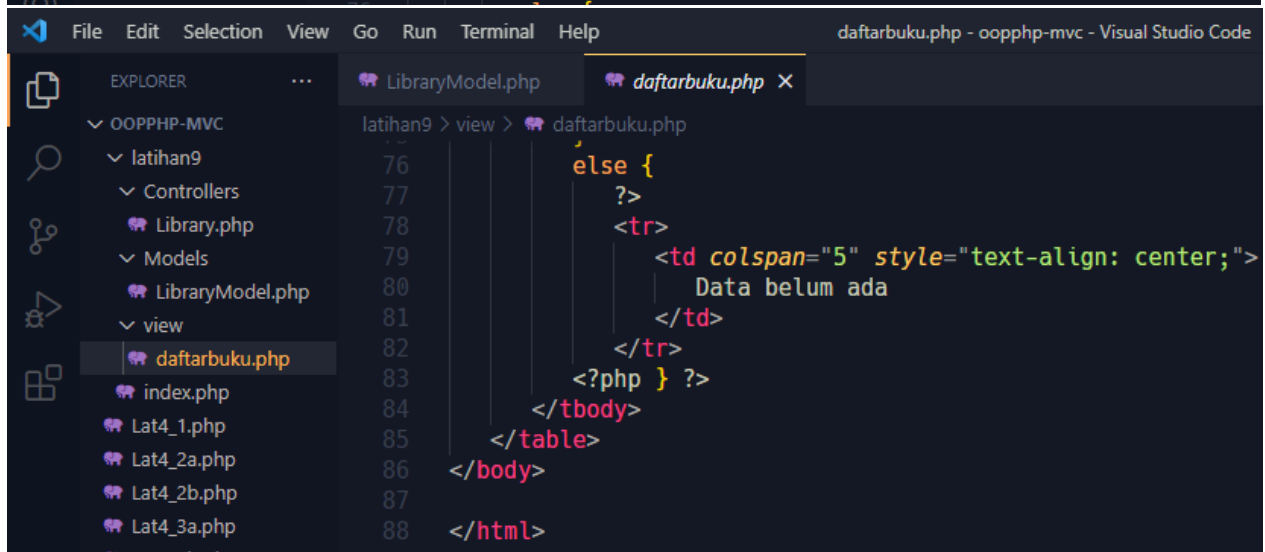
```
File Edit Selection View Go Run Terminal Help daftarbuku.php - oopphp-mvc - Visual Studio Code

EXPLORER
  OOPPH...
  latihan9
    Controllers
    Library.php
    Models
    LibraryModel.php
    view
      daftarbuku.php
      index.php
      Lat4_1.php
      Lat4_2a.php
      Lat4_2b.php
      Lat4_3a.php
      Lat4_3b.php
      Lat4_4a.php
      Lat4_4b.php
      Lat4_5a.php
      Lat4_5b.php
      Lat4_6.php
      Lat4_7.php
      Lat4_8.php

latihan9 > view > daftarbuku.php
1  <!DOCTYPE html>
2  <html lang="en">
3
4  <head>
5    <meta charset="UTF-8">
6    <meta http-equiv="X-UA-Compatible" content="IE=edge">
7    <meta name="viewport" content="width=device-width, initial-scale=1.0">
8    <title>DAFTAR BUKU || Debi Safa Nurdewanti G.211.21.0078</title>
9  <style>
10    body {
11      background : linear-gradient(to right, #CCDDEE, #FFCCEE, #CCBBDD);
12      display: flex;
13      justify-content: center;
14      align-items: center;
15      flex-direction: column;
16      height: 60vh;
17    }
18
19    .table {
20      border: 3px solid #222;
21    }
22
23    tr th {
24      border: 3px solid #222;
25      padding: 3px 7px;
26      font-size: 18px;
27    }
28
29    tr td {
30      border: 3px solid #222;
31      padding: 3px 7px;
32      font-size: 16px;
33    }
34
35    h3{
36      font-size: 27px;
37      margin-bottom: 17px;
38      text-align: center;
39    }
40
41    p {
42      text-align: center;
43      font-size: 14px;
44      margin-top: 22px;
45    }
46  </style>
47 </head>
```

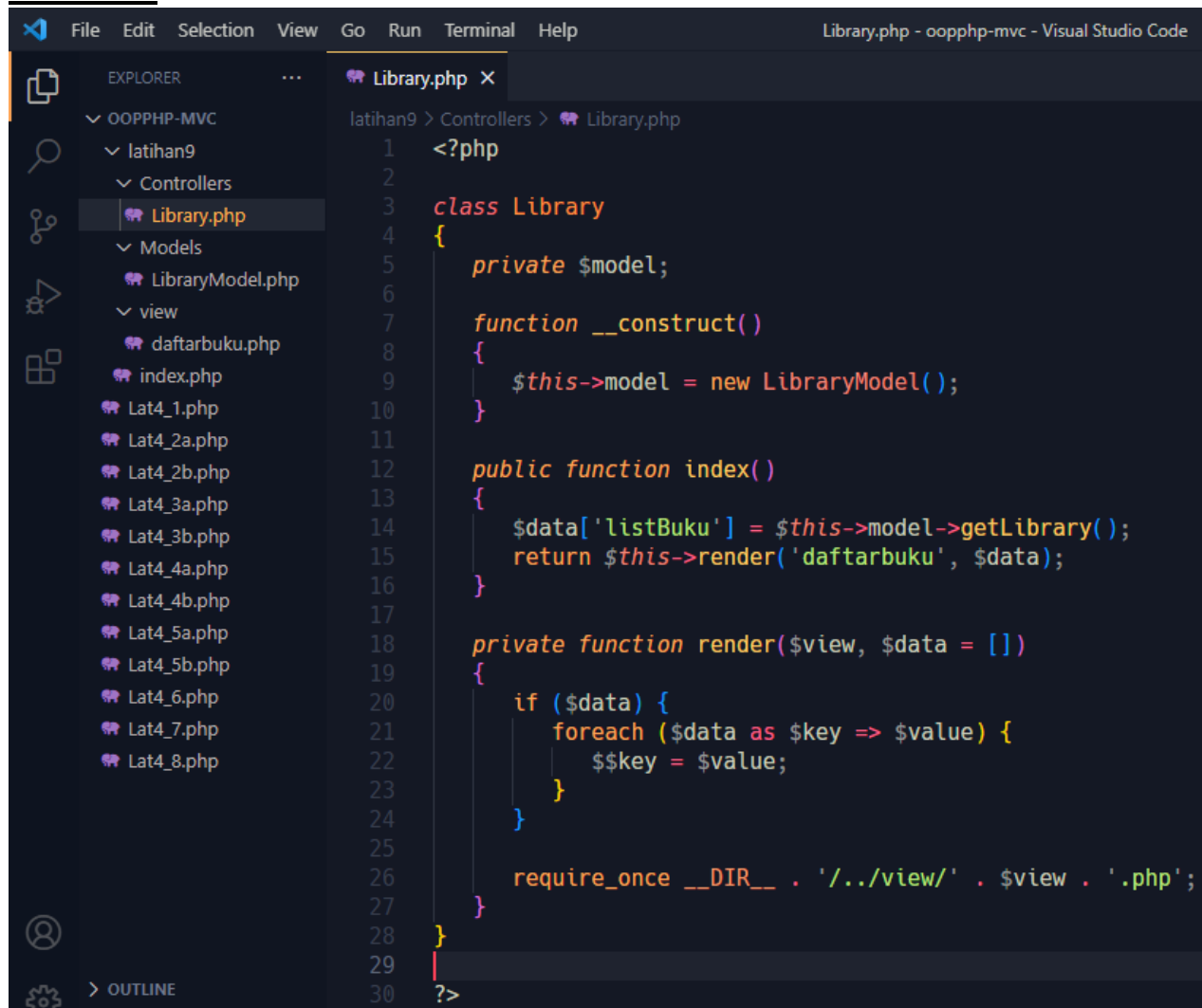


```
49 <body>
50 <h3>OOP PHP MVC tanpa Framework</h3>
51 <table class="table">
52   <thead>
53     <tr>
54       <th>No.</th>
55       <th>Judul</th>
56       <th>Pengarang</th>
57       <th>Penerbit</th>
58       <th>Tahun</th>
59     </tr>
60   </thead>
61   <tbody>
62     <?php
63     $no = 1;
64     if ($listBuku) {
65       foreach ($listBuku as $value) { ?>
66         <tr>
67           <td style="text-align: center;"><? $no++; ?></td>
68           <td><? $value["Judul"]; ?></td>
69           <td><? $value["Pengarang"]; ?></td>
70           <td><? $value["Penerbit"]; ?></td>
71           <td><? $value["Tahun"]; ?></td>
72         </tr>
73       <?php
74     }
75 }
```



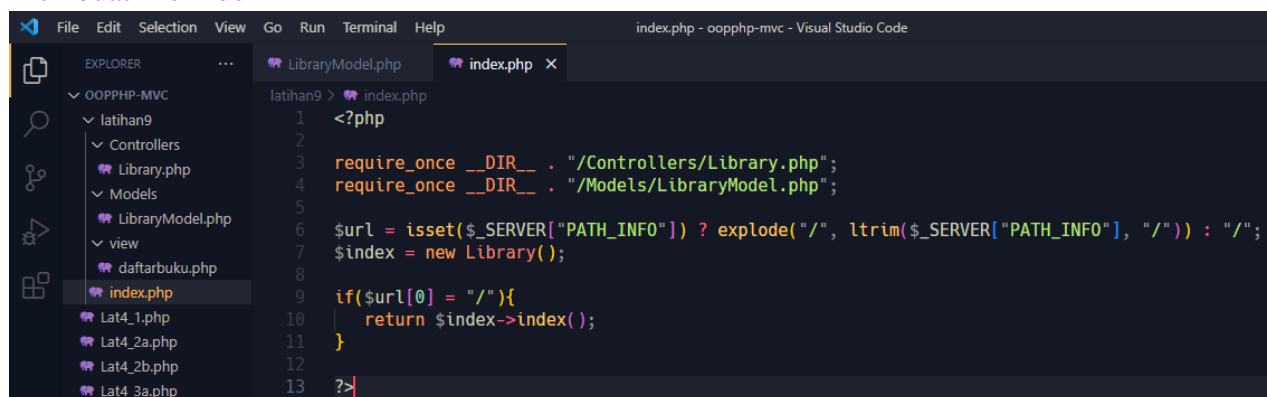
```
76 else {
77   ?>
78   <tr>
79     <td colspan="5" style="text-align: center;">
80       Data belum ada
81     </td>
82   </tr>
83   <?php } ?>
84 </tbody>
85 </table>
86 </body>
87
88 </html>
```

Controllers



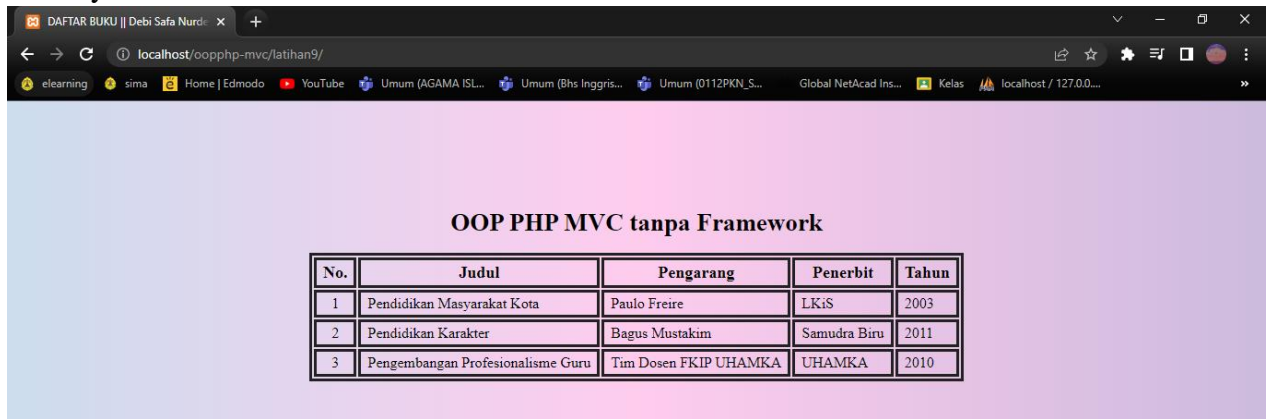
```
1 <?php
2
3 class Library
4 {
5     private $model;
6
7     function __construct()
8     {
9         $this->model = new LibraryModel();
10    }
11
12    public function index()
13    {
14        $data['listBuku'] = $this->model->getLibrary();
15        return $this->render('daftarbuku', $data);
16    }
17
18    private function render($view, $data = [])
19    {
20        if ($data) {
21            foreach ($data as $key => $value) {
22                $$key = $value;
23            }
24        }
25
26        require_once __DIR__ . '/../view/' . $view . '.php';
27    }
28 }
29
30 ?>
```

➤ Membuat file Index



```
1 <?php
2
3 require_once __DIR__ . "/Controllers/Library.php";
4 require_once __DIR__ . "/Models/LibraryModel.php";
5
6 $url = isset($_SERVER["PATH_INFO"]) ? explode("/", ltrim($_SERVER["PATH_INFO"], "/")) : "/";
7 $index = new Library();
8
9 if($url[0] == "/"){
10     return $index->index();
11 }
12
13 ?>
```


➤ Hasilnya



The screenshot shows a web browser window with the address bar displaying 'localhost/oopphp-mvc/latihan9/'. The page content features a table with the title 'OOP PHP MVC tanpa Framework'. The table has five columns: 'No.', 'Judul', 'Pengarang', 'Penerbit', and 'Tahun'. It contains three rows of data representing books.

No.	Judul	Pengarang	Penerbit	Tahun
1	Pendidikan Masyarakat Kota	Paulo Freire	LKiS	2003
2	Pendidikan Karakter	Bagus Mustakim	Samudra Biru	2011
3	Pengembangan Profesionalisme Guru	Tim Dosen FKIP UHAMKA	UHAMKA	2010