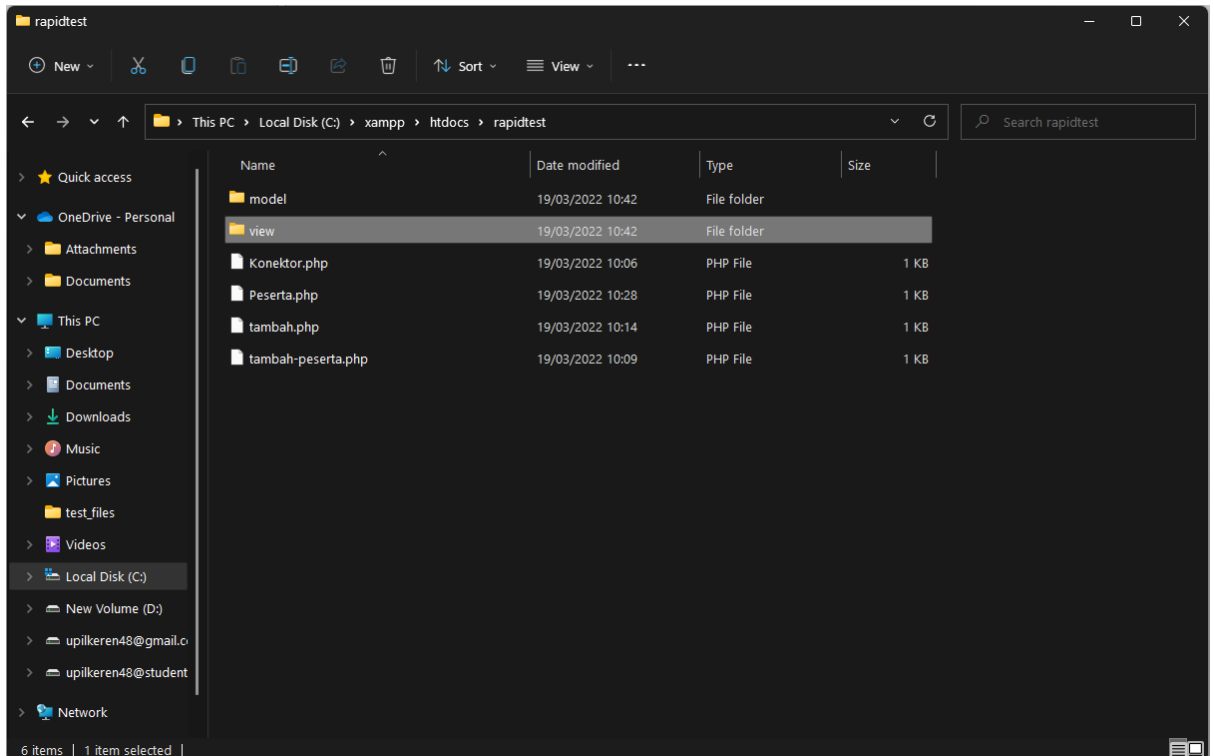
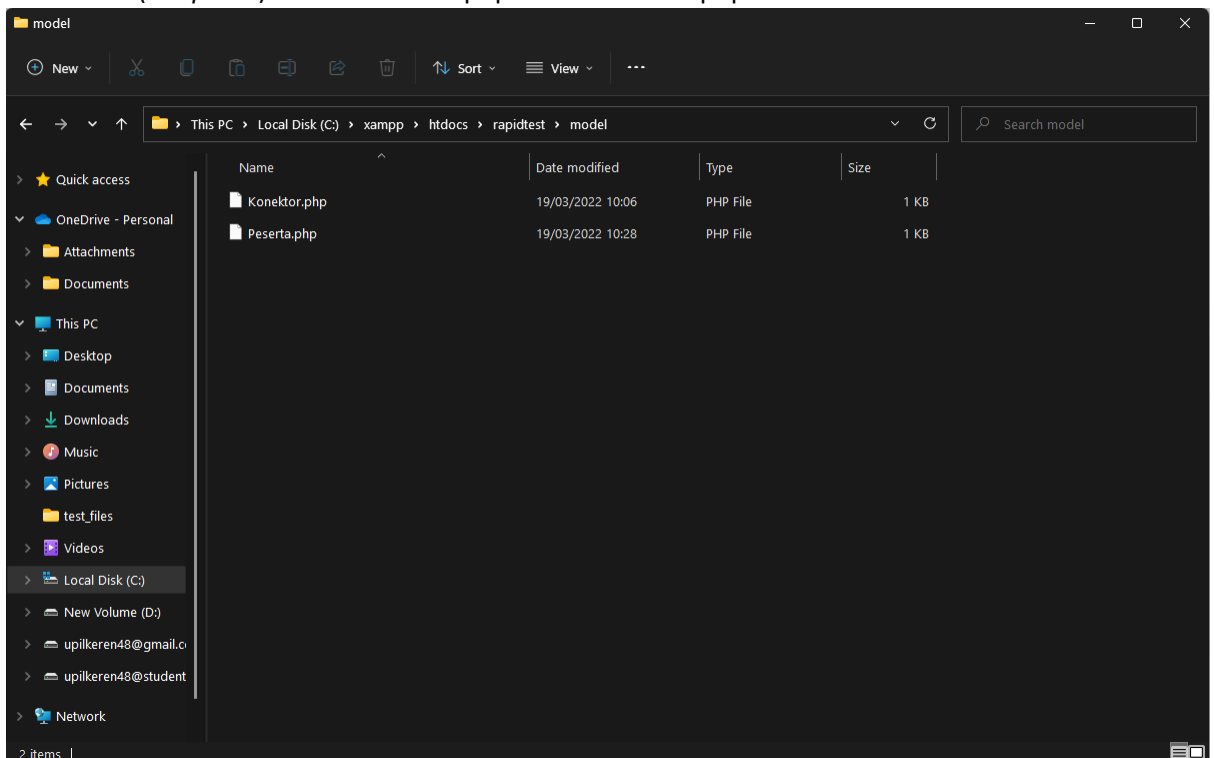


Persiapan

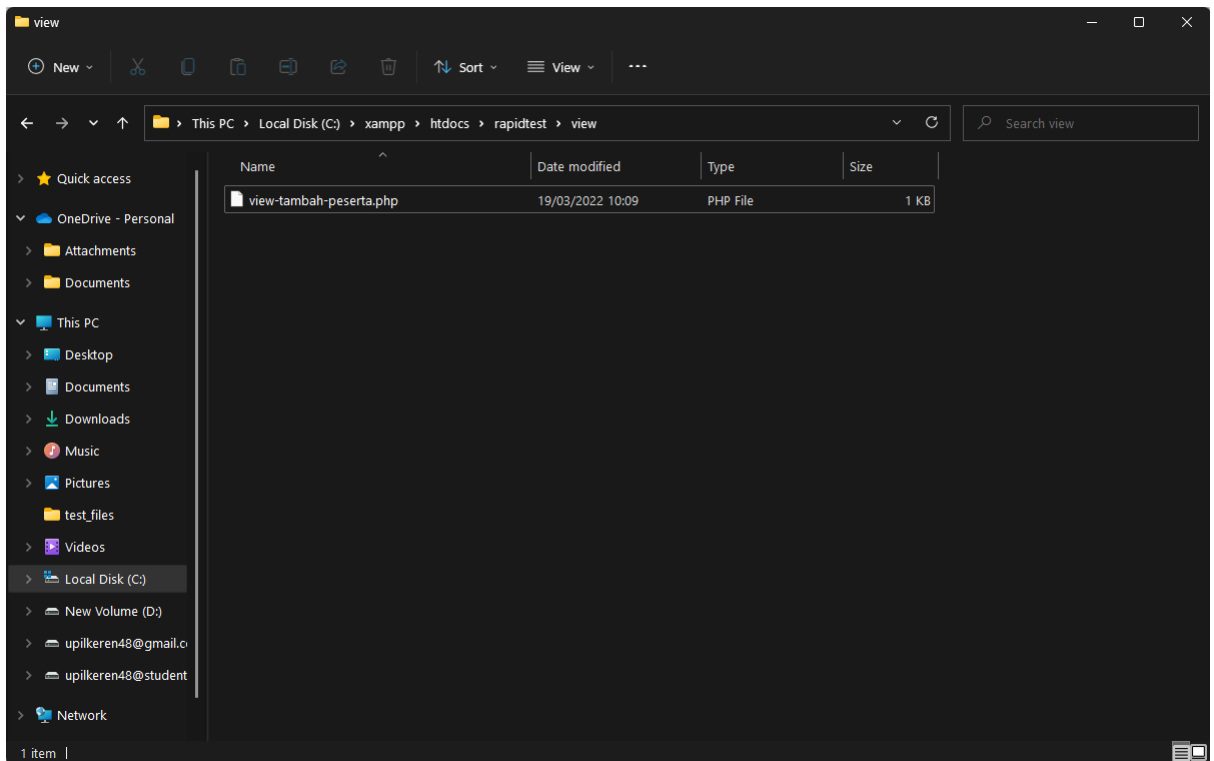
1. Pada folder 'rapidtest' yang telah kita buat di perkuliahan yang lalu, buatlah dua sub-folder baru, masing-masing dengan nama 'model' dan 'view'. Sesuai dengan namanya, folder 'model' akan berisi berkas-berkas model, demikian pula pada folder 'view' akan berisi berkas-berkas view



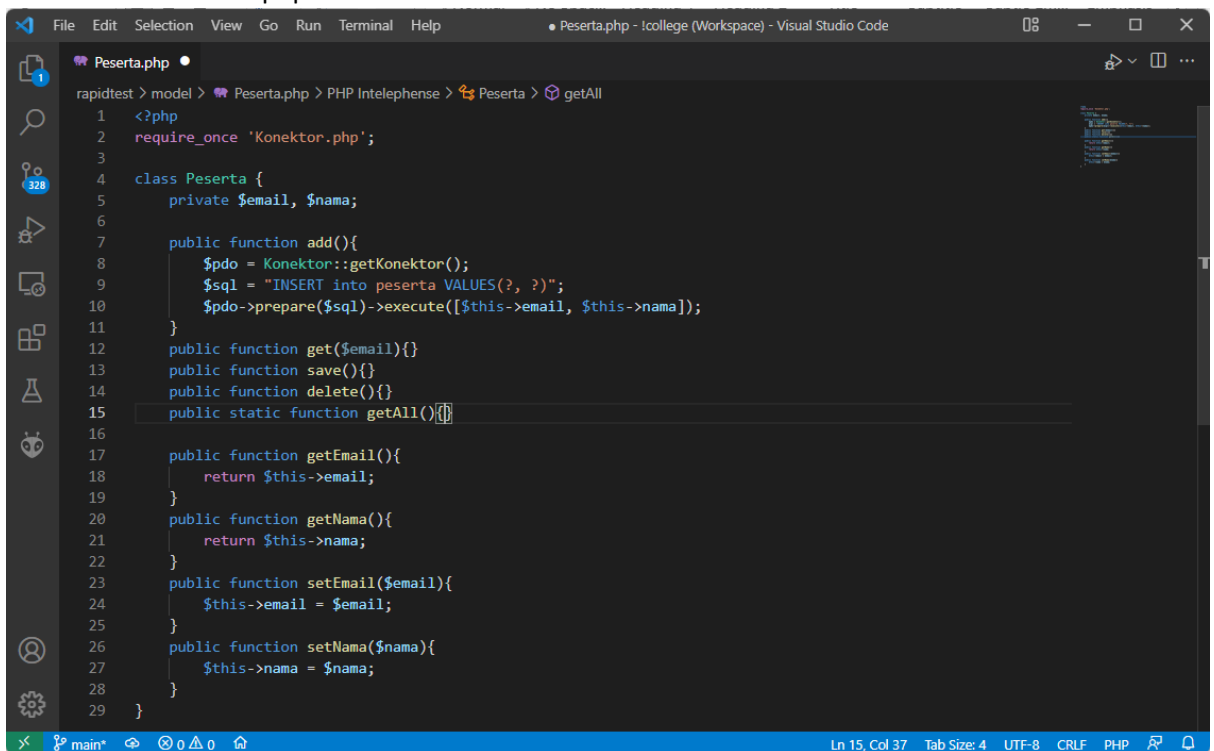
2. Pindahkan (*cut-paste*) berkas 'Peserta.php' dan 'Konektor.php' ke dalam folder 'model'



3. Pindahkan (*cut-paste*) berkas 'tambah-peserta.php' ke dalam folder 'view'. Setelah itu ubah namanya menjadi 'view-tambah-peserta.php'.



1. Buka berkas 'Peserta.php'



2. Tambahkan *method* konstruktor pada kelas Peserta seperti kode di bawah ini:

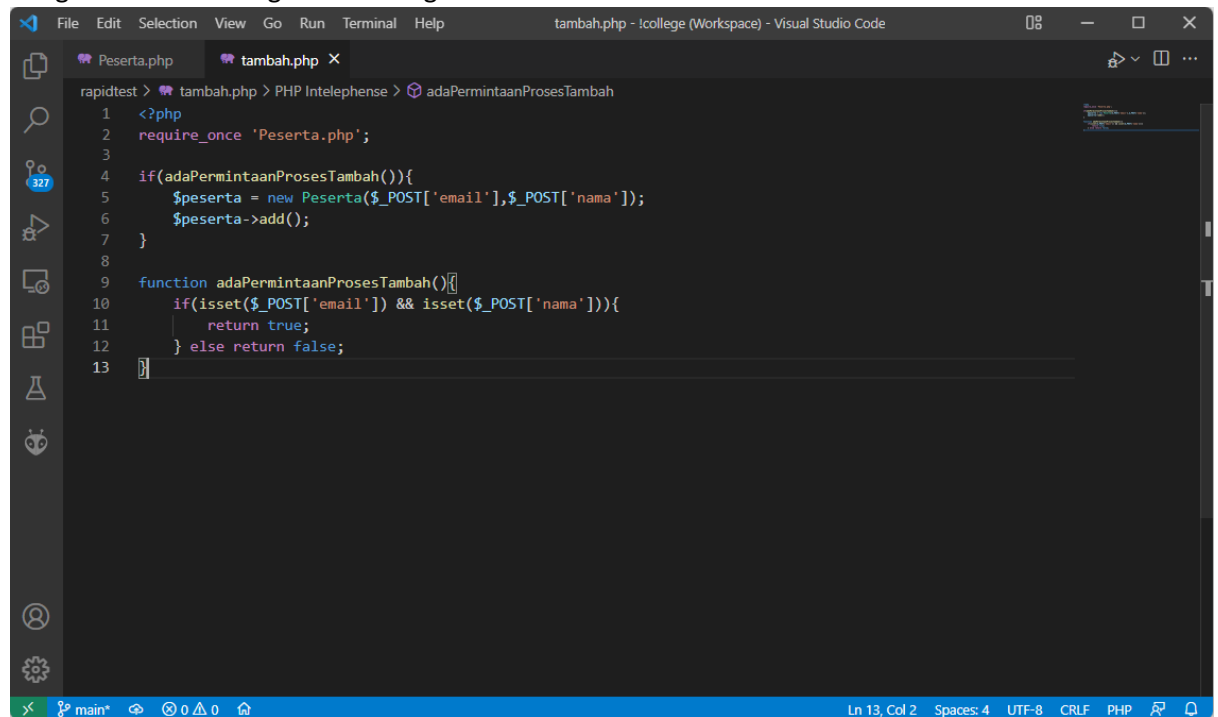
```
1 <?php
2 require_once 'Konektor.php';
3
4 class Peserta {
5     private $email, $nama;
6
7     public function __construct($email, $nama){
8         $this->email = $email;
9         $this->nama = $nama;
10    }
11
12    public function add(){
13        $pdo = Konektor::getKonektor();
14        $sql = "INSERT into peserta VALUES(?, ?)";
15        $pdo->prepare($sql)->execute([$this->email, $this->nama]);
16    }
17
18    public function get($email){}
19    public function save(){}
20    public function delete(){}
21    public static function getAll(){}
22
23
24    public function getEmail(){
25        return $this->email;
26    }
27
28    public function getNama(){
29        return $this->nama;
30    }
31
32    public function setEmail($email){
33        $this->email = $email;
34    }
35
36    public function setNama($nama){
37        $this->nama = $nama;
38    }
39 }
```

3. Karena kita menambahkan *constructor method* pada kelas Peserta, maka pemanggilan kosntruktor di tempat lain juga harus menyesuaikan. Maka dari itu, buka berkas ‘tambah.php’, kemudian ganti isinya menjadi seperti kode di bawah ini (kode merah yang dicoret menandakan kode sebelumnya yang harus dihapus, sedangkan kode yang ditandai warna abu-abu adalah kode yang harus ditambahkan):

```
1 <?php
2 require_once 'Peserta.php';
3
4 if(isset($_POST['email']) && isset($_POST['nama'])){}
5     $peserta = new Peserta($_POST['email'],$_POST['nama']);
6     $peserta->add();
7 }
```

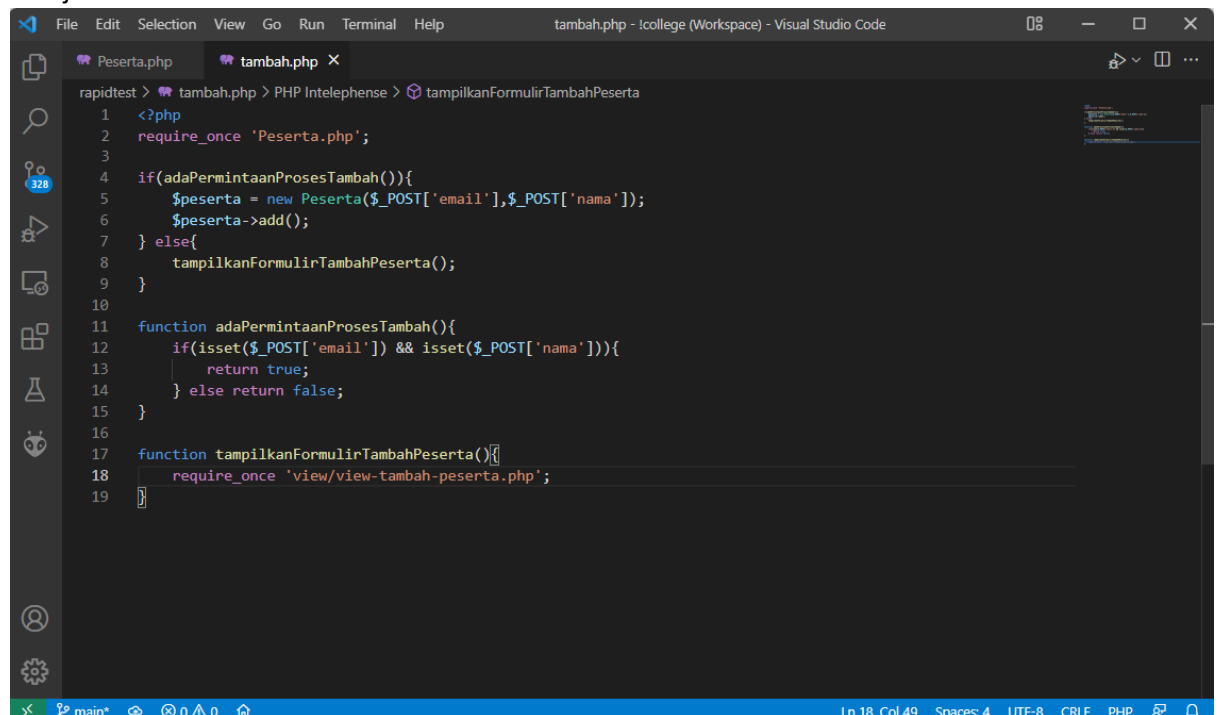
4. Pada kode di poin nomor 3, ada sesuatu yang tidak elegan, yaitu pada kondisi percabangan (*if statement*). Kondisi *if* tersebut kita buat untuk memeriksa apakah ada formulir yang dikirimkan oleh klien atau tidak. Kita akan perbaiki dengan melakukan abstraksi terhadap

kondisi tersebut sehingga kode menjadi lebih deskriptif. Abstraksi yang kita lakukan adalah dengan membuat fungsi baru sebagai berikut:



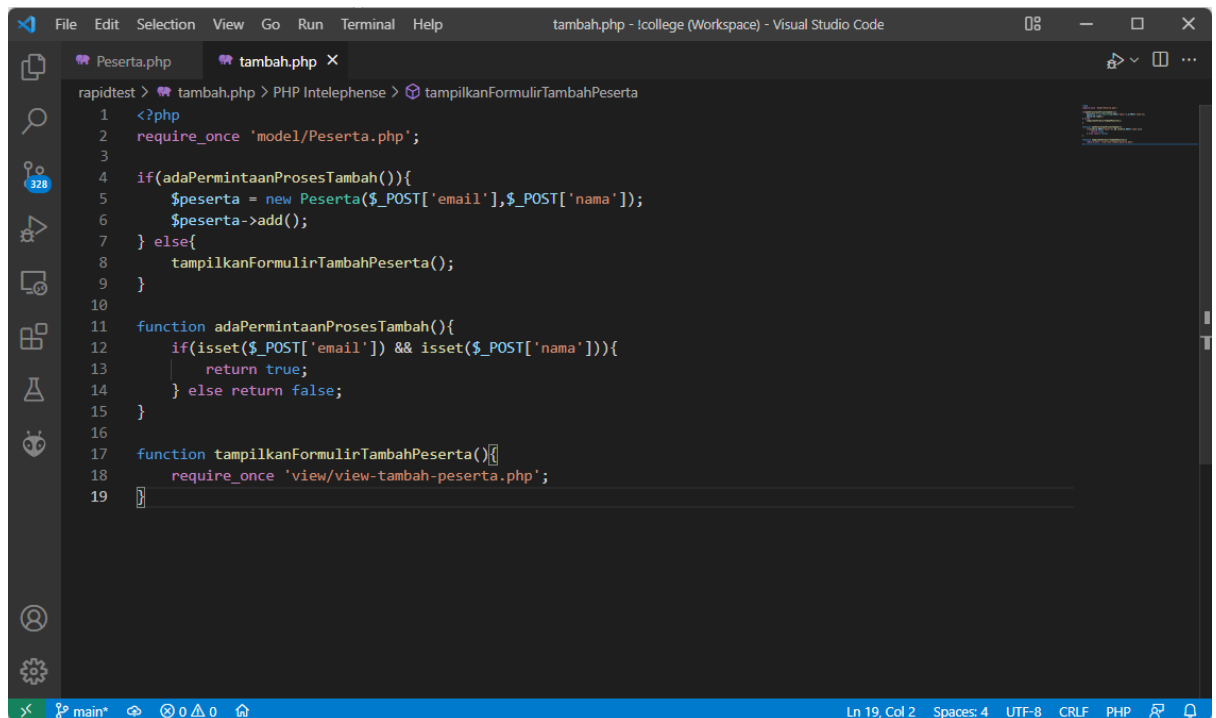
```
1 <?php
2 require_once 'Peserta.php';
3
4 if(adaPermintaanProsesTambah()){
5     $peserta = new Peserta($_POST['email'],$_POST['nama']);
6     $peserta->add();
7 }
8
9 function adaPermintaanProsesTambah(){
10     if(isset($_POST['email']) && isset($_POST['nama'])){
11         return true;
12     } else return false;
13 }
```

5. Pada perkuliahan sebelumnya, formulir pendaftaran peserta *rapid test* langsung diakses oleh klien. Sekarang, setelah menggunakan MVC, maka formulir tersebut tidak langsung diakses oleh klien, melainkan harus melalui perantara *controller* (lihat kembali gambar 1). Formulir akan ditampilkan oleh *controller* apabila tidak ada permintaan proses tambah (atau dengan kata lain, fungsi `adaPermintaanProsesTambah()` bernilai *false*). Mari kita tambahkan kode menjadi di bawah ini:



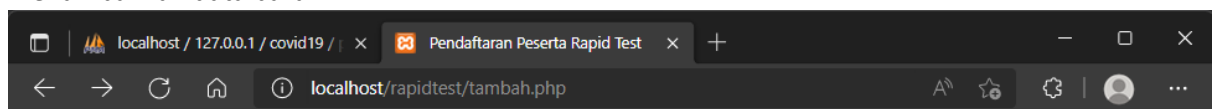
```
1 <?php
2 require_once 'Peserta.php';
3
4 if(adaPermintaanProsesTambah()){
5     $peserta = new Peserta($_POST['email'],$_POST['nama']);
6     $peserta->add();
7 } else{
8     tampilkanFormulirTambahPeserta();
9 }
10
11 function adaPermintaanProsesTambah(){
12     if(isset($_POST['email']) && isset($_POST['nama'])){
13         return true;
14     } else return false;
15 }
16
17 function tampilkanFormulirTambahPeserta(){
18     require_once 'view/view-tambah-peserta.php';
19 }
```

6. Karena tadi kita juga memindahkan berkas 'Peserta.php' ke folder 'model', maka kita juga perlu mengubah isi dari baris pernyataan `require_once` menjadi:



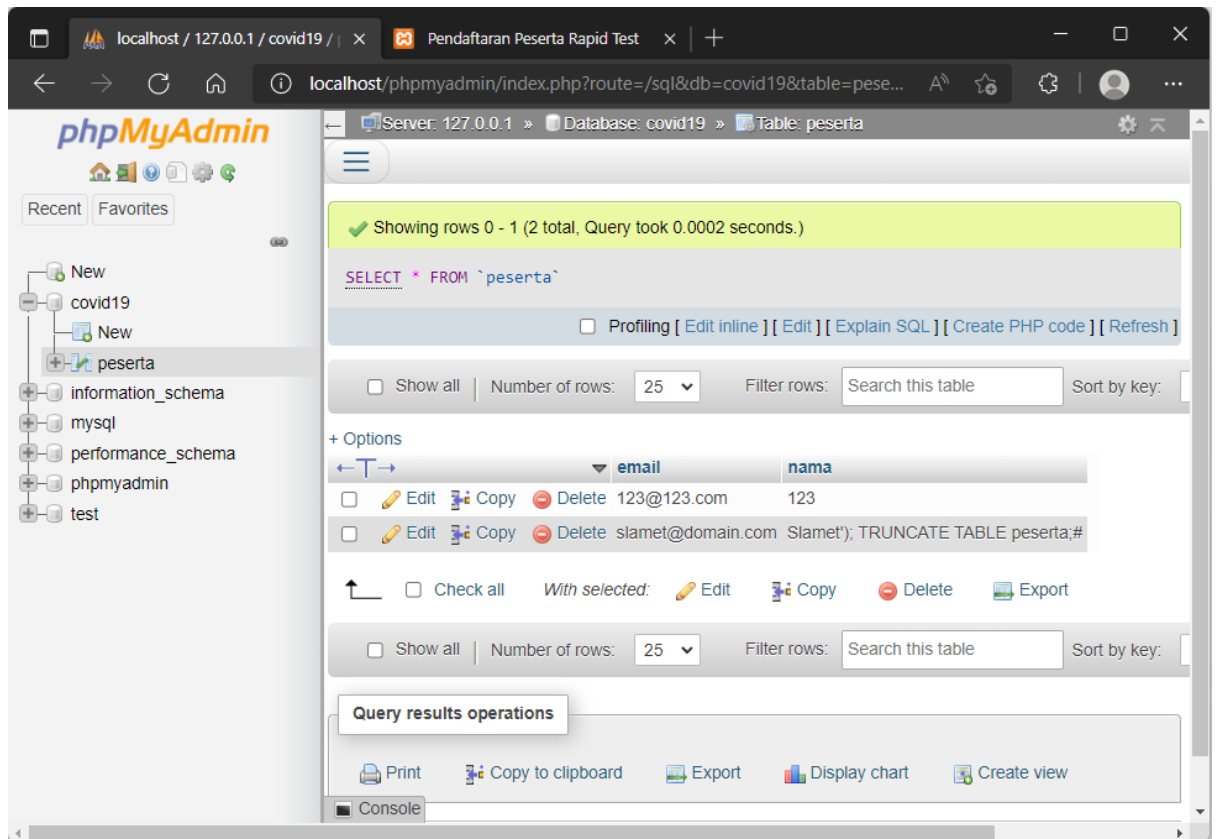
```
1 <?php
2 require_once 'model/Peserta.php';
3
4 if(adaPermintaanProsesTambah()){
5     $peserta = new Peserta($_POST['email'],$_POST['nama']);
6     $peserta->add();
7 } else{
8     tampilkanFormulirTambahPeserta();
9 }
10
11 function adaPermintaanProsesTambah(){
12     if(isset($_POST['email']) && isset($_POST['nama']))){
13         return true;
14     } else return false;
15 }
16
17 function tampilkanFormulirTambahPeserta(){
18     require_once 'view/view-tambah-peserta.php';
19 }
```

7. Jika sudah, silakan dicoba dijalankan pada peramban web dan dicoba pula untuk menambahkan data baru.



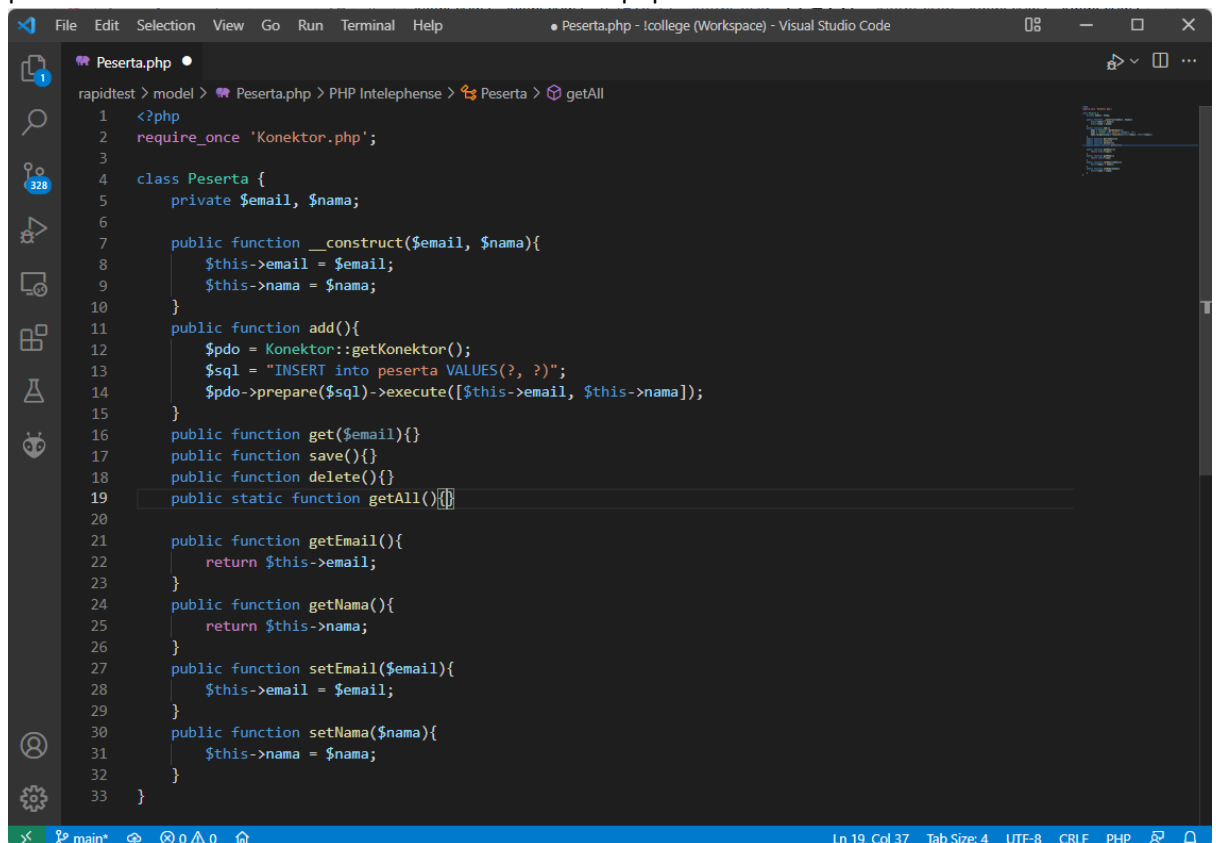
Formulir Pendaftaran Rapid Test

Email:	<input type="text" value="123@123.com"/>
Nama:	<input type="text" value="123"/>
<input type="button" value="Daftar"/>	

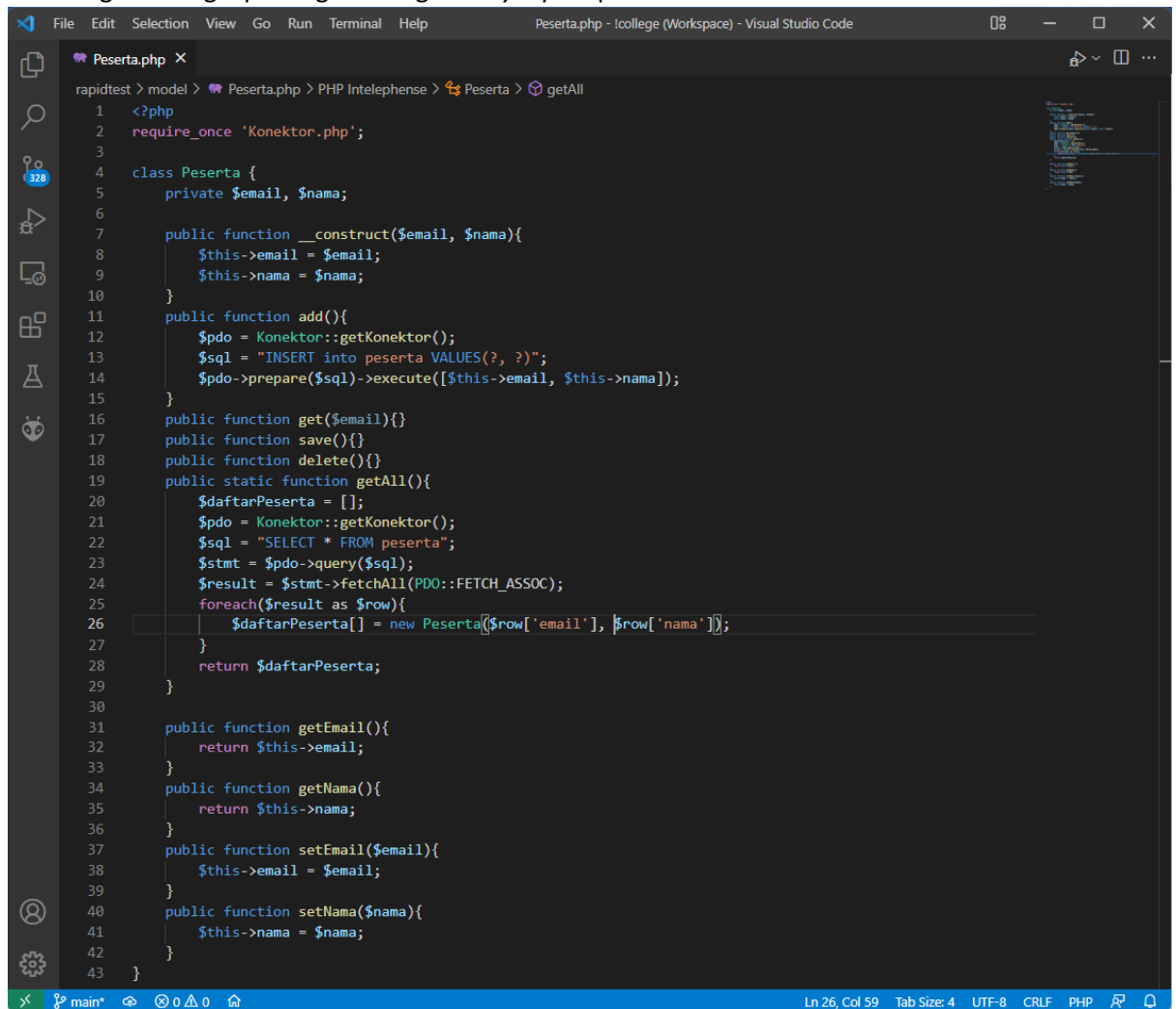


Operasi melihat daftar, mengubah dan menghapus peserta

1. Untuk menampilkan daftar peserta *rapid test*, kita perlu menambahkan *business logic* baru pada model Peserta. Silahkan buka berkas 'Peserta.php'

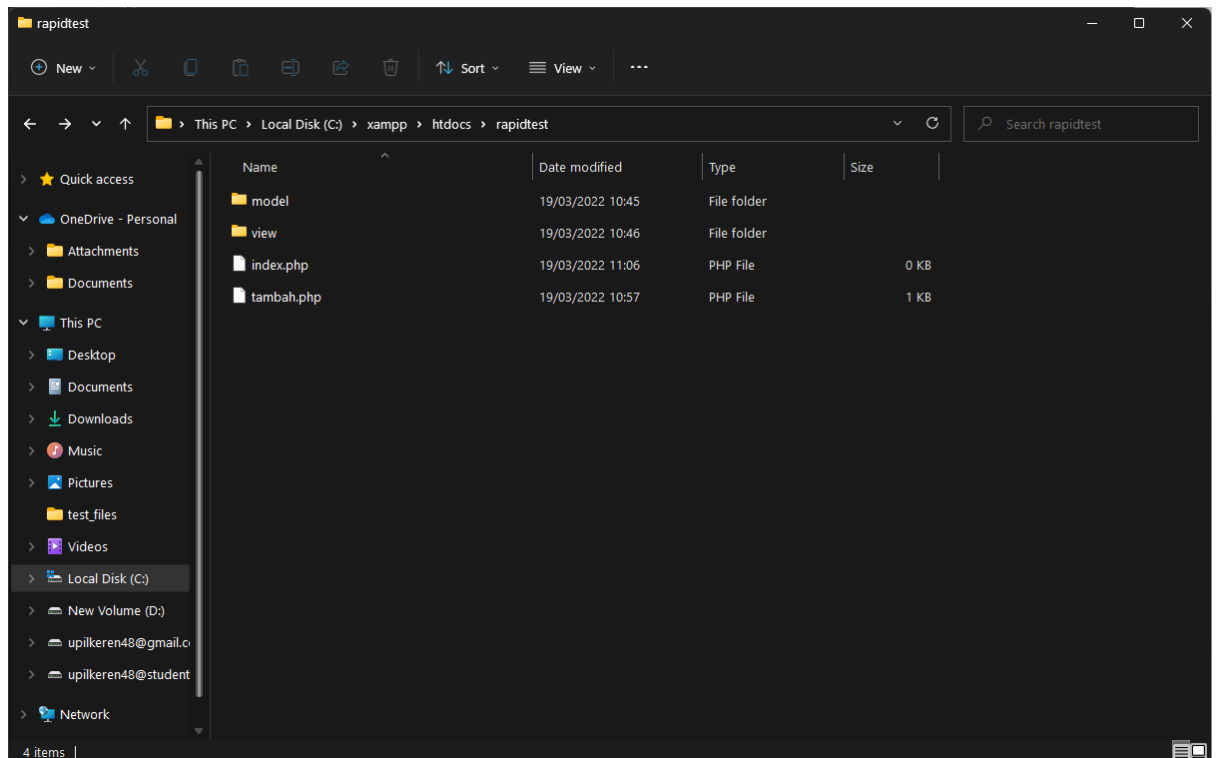


2. Pada perkuliahan sebelumnya, kita telah mendefinisikan fungsi statis dengan nama `getAll()`. Sekarang kita lengkapi dengan mengisi *body*-nya seperti kode di bawah ini:

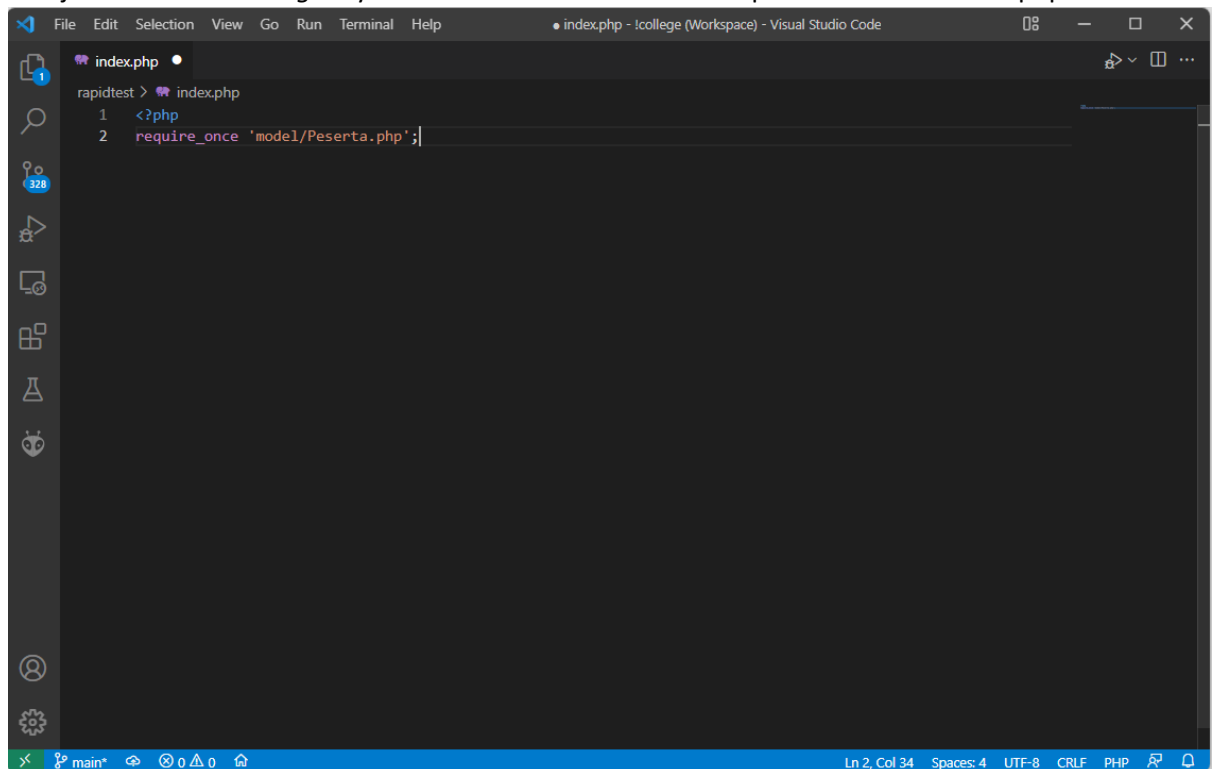


```
1 <?php
2 require_once 'Konektor.php';
3
4 class Peserta {
5     private $email, $nama;
6
7     public function __construct($email, $nama){
8         $this->email = $email;
9         $this->nama = $nama;
10    }
11    public function add(){
12        $pdo = Konektor::getKonektor();
13        $sql = "INSERT into peserta VALUES(?, ?)";
14        $pdo->prepare($sql)->execute([$this->email, $this->nama]);
15    }
16    public function get($email){}
17    public function save(){}
18    public function delete(){}
19    public static function getAll(){
20        $daftarPeserta = [];
21        $pdo = Konektor::getKonektor();
22        $sql = "SELECT * FROM peserta";
23        $stmt = $pdo->query($sql);
24        $result = $stmt->fetchAll(PDO::FETCH_ASSOC);
25        foreach($result as $row){
26            $daftarPeserta[] = new Peserta($row['email'], $row['nama']);
27        }
28        return $daftarPeserta;
29    }
30
31    public function getEmail(){
32        return $this->email;
33    }
34    public function getNama(){
35        return $this->nama;
36    }
37    public function setEmail($email){
38        $this->email = $email;
39    }
40    public function setNama($nama){
41        $this->nama = $nama;
42    }
43 }
```

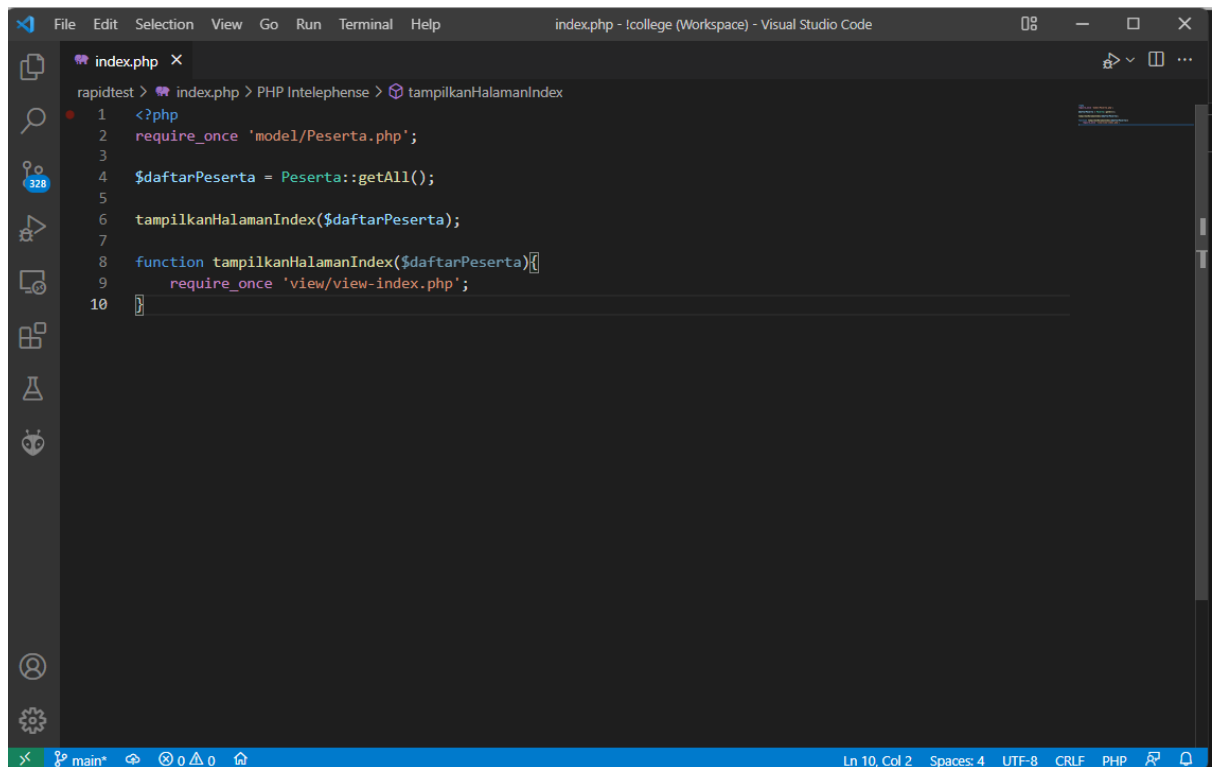
1. Buatlah berkas *controller* baru dengan nama 'index.php'



2. Karena kita akan menampilkan daftar peserta, maka kita perlu bantuan *model* Peserta untuk menjalankan *business logic*-nya. Tambahkan kode berikut ini pada *controller* 'index.php'

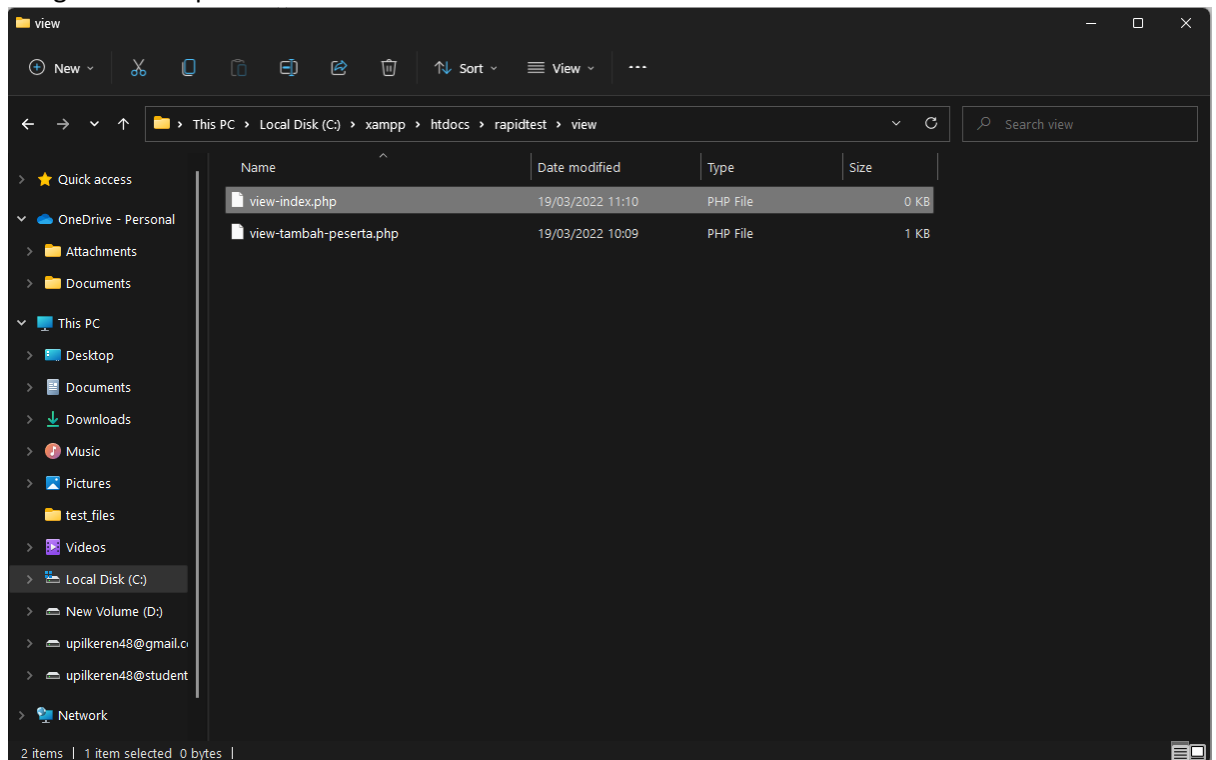


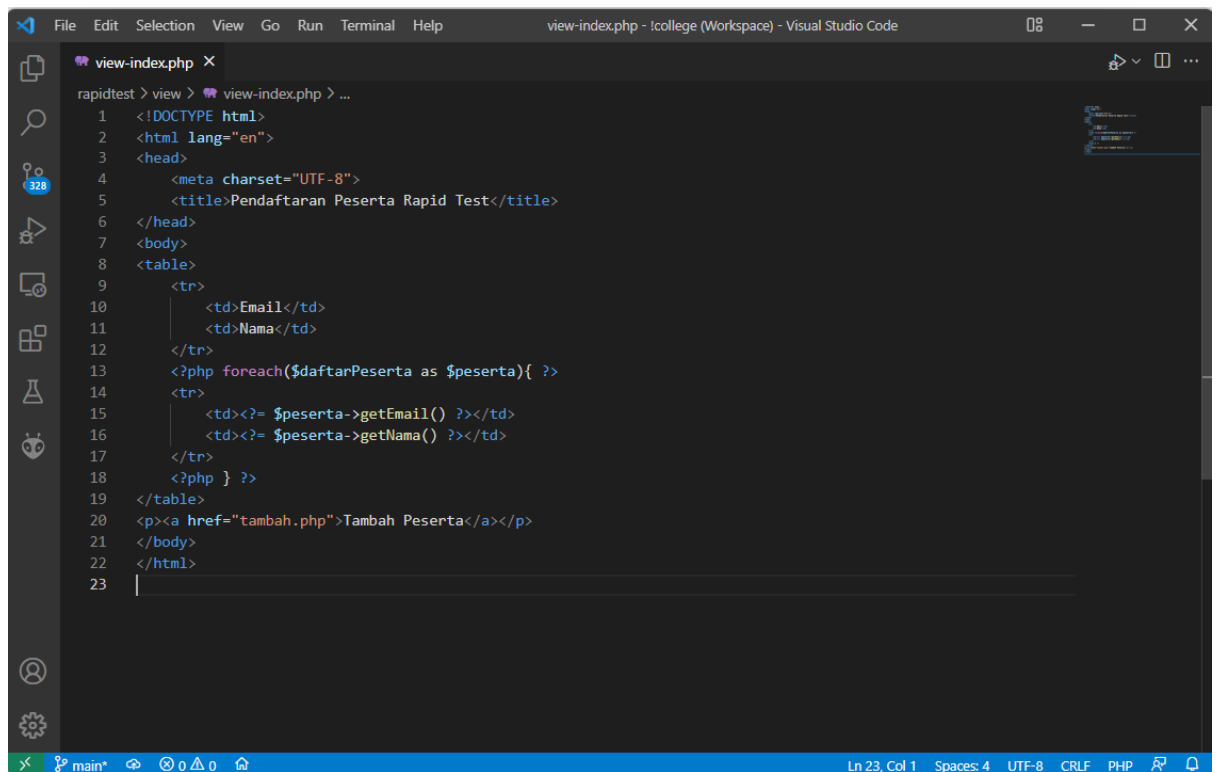
3. Berikutnya, *controller* membutuhkan *view* untuk menampilkan data. Tambahkan kode berikut ini pada *controller* 'index.php'



```
index.php
1 <?php
2 require_once 'model/Peserta.php';
3
4 $daftarPeserta = Peserta::getAll();
5
6 tampilkanHalamanIndex($daftarPeserta);
7
8 function tampilkanHalamanIndex($daftarPeserta){
9     require_once 'view/view-index.php';
10 }
```

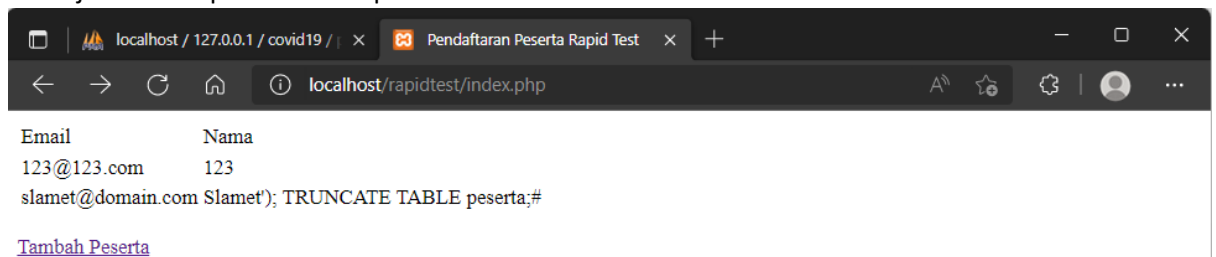
4. Pada kode di poin nomor 3, terlihat bahwa fungsi `tampilkanHalamanIndex` melakukan *passing* variabel bernama `$daftarPeserta` yang diperoleh dari *model*. Variabel `$daftarPeserta` berisi daftar peserta dalam bentuk larik (*array*) `Peserta`. Larik tersebut akan kita proses pada *view*. Pada folder *view*, mari kita buat file dengan nama `'view-index.php'`, kemudian isi dengan kode seperti di bawah ini:

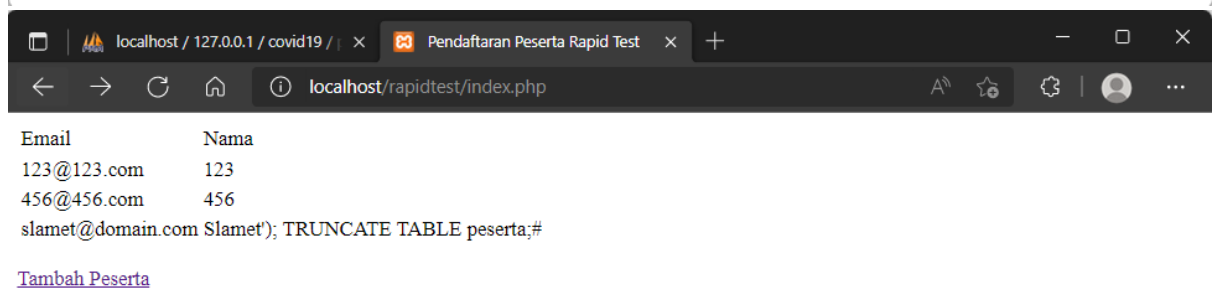
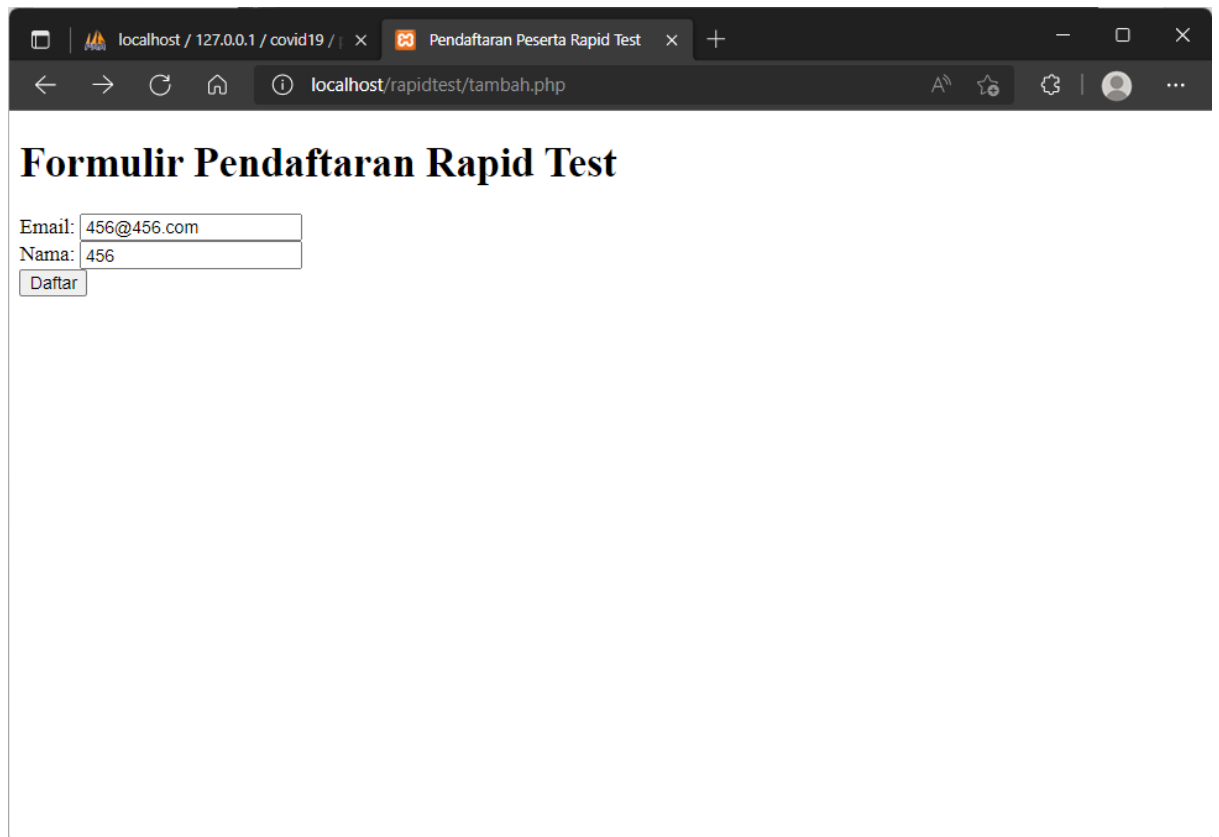




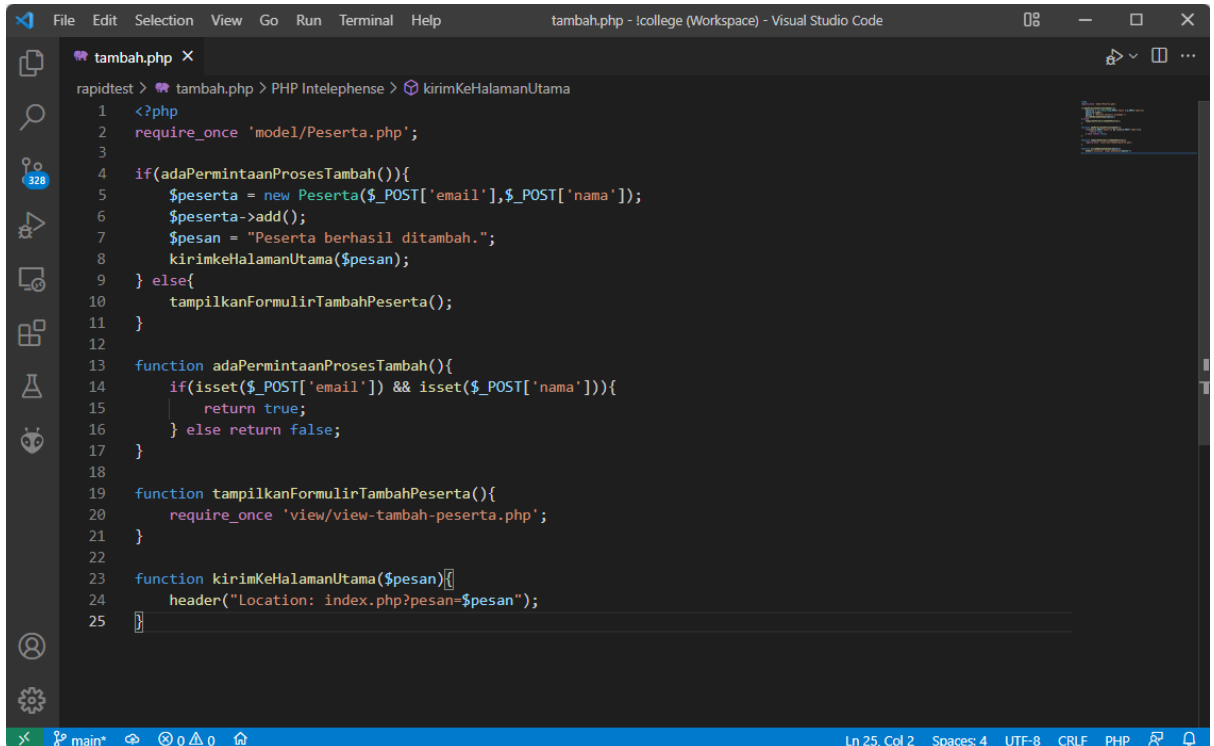
```
view-index.php X
rapidtest > view > view-index.php > ...
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <title>Pendaftaran Peserta Rapid Test</title>
6 </head>
7 <body>
8   <table>
9     <tr>
10      <td>Email</td>
11      <td>Nama</td>
12    </tr>
13    <?php foreach($daftarPeserta as $peserta){ ?>
14      <tr>
15        <td><?= $peserta->getEmail() ?></td>
16        <td><?= $peserta->getNama() ?></td>
17      </tr>
18    <?php } ?>
19  </table>
20  <p><a href="tambah.php">Tambah Peserta</a></p>
21 </body>
22 </html>
23
```

5. Bila sudah selesai, silahkan saudara akses <http://localhost/rapidtest/index.php> dengan peramban web. Jika berhasil, maka saudara akan dapat daftar peserta dan tautan untuk menuju formulir penambahan peserta.



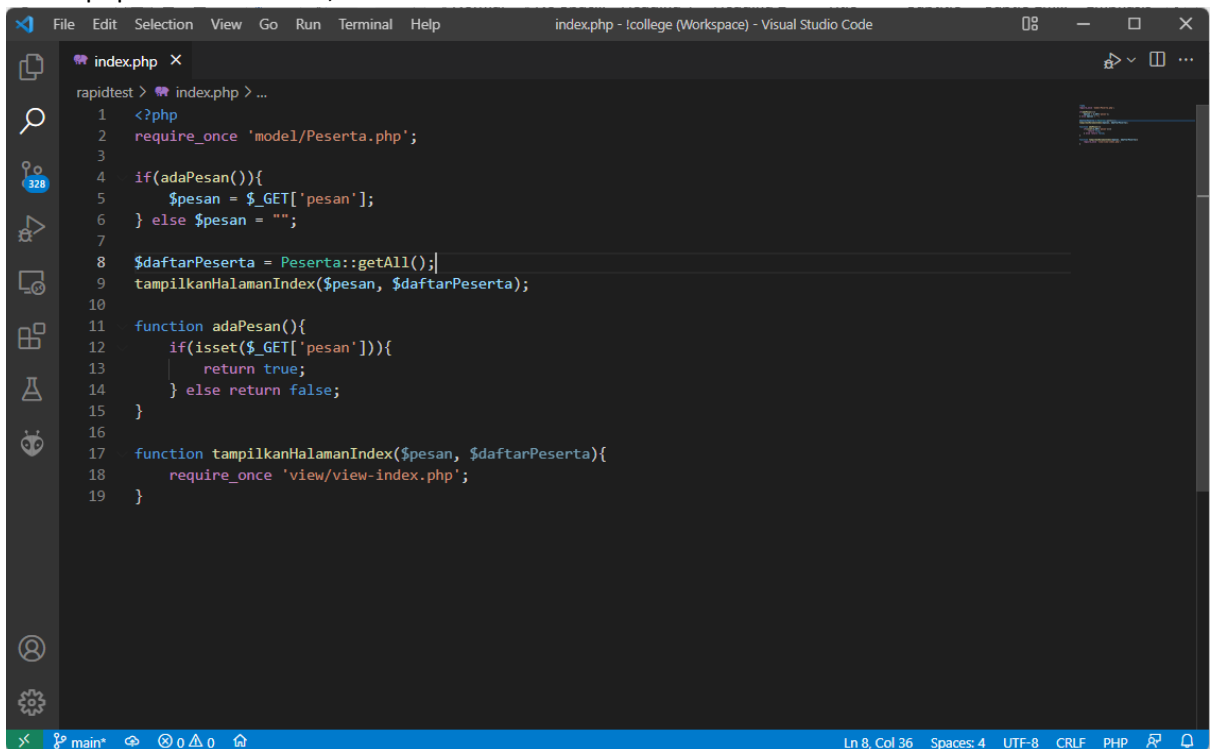


1. Buka berkas tambah.php, kemudian tambahkan menjadi seperti kode di bawah ini:



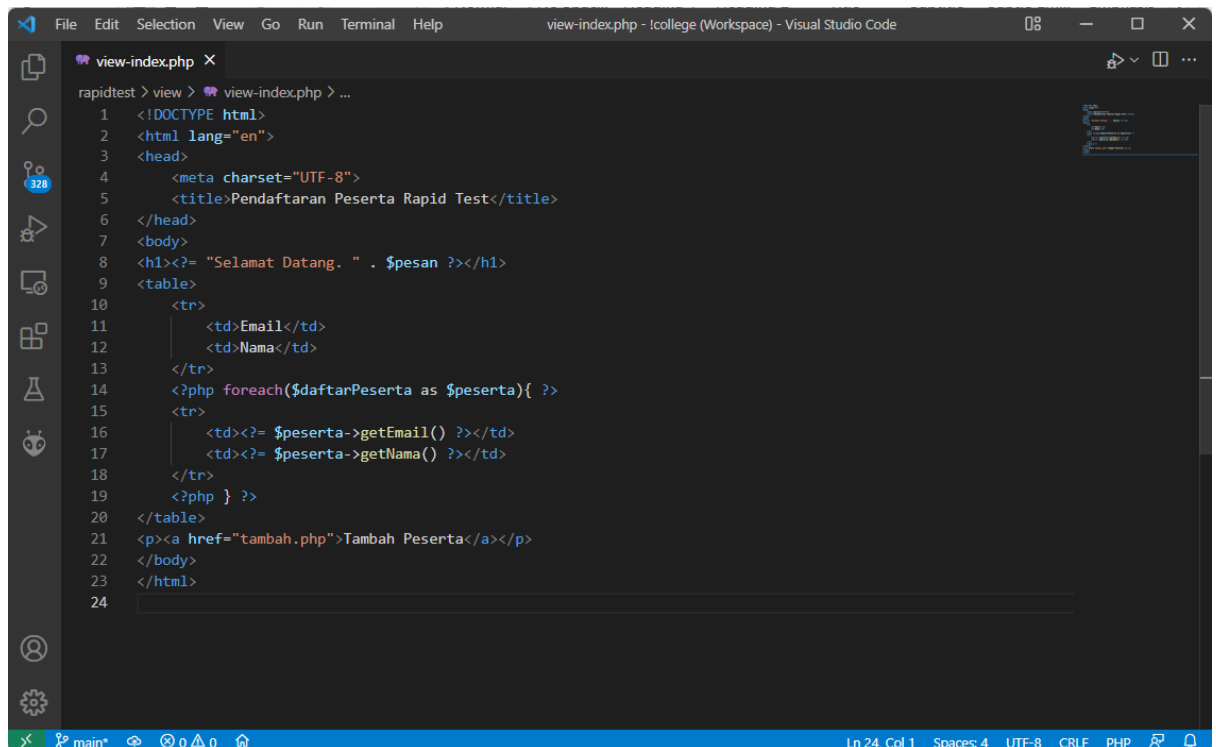
```
1 <?php
2 require_once 'model/Peserta.php';
3
4 if(adaPermintaanProsesTambah()){
5     $peserta = new Peserta($_POST['email'],$_POST['nama']);
6     $peserta->add();
7     $pesan = "Peserta berhasil ditambah.";
8     kirimKeHalamanUtama($pesan);
9 } else{
10     tampilkanFormulirTambahPeserta();
11 }
12
13 function adaPermintaanProsesTambah(){
14     if(isset($_POST['email']) && isset($_POST['nama'])){
15         return true;
16     } else return false;
17 }
18
19 function tampilkanFormulirTambahPeserta(){
20     require_once 'view/view-tambah-peserta.php';
21 }
22
23 function kirimKeHalamanUtama($pesan){
24     header("Location: index.php?pesan=$pesan");
25 }
```

2. Pada kode di poin nomor 1, kita ingin mengirim pesan “Peserta berhasil ditambah.” Setelah penambahkn peserta untuk ditampilkan pada halaman utama. Oleh karena itu, kita perlu menyesuaikan *controller* dan *view* untuk halamn utama. Pertama-tama kita ubah *controller* ‘index.php’ terlebih dahulu, tambahkan kode berikut ini:



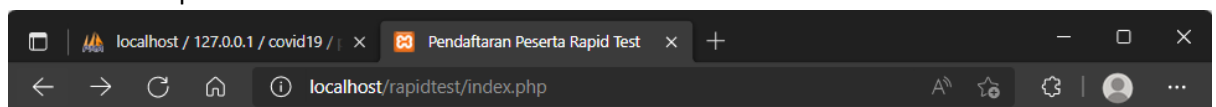
```
1 <?php
2 require_once 'model/Peserta.php';
3
4 if(adaPesan()){
5     $pesan = $_GET['pesan'];
6 } else $pesan = "";
7
8 $daftarPeserta = Peserta::getAll();
9 tampilkanHalamanIndex($pesan, $daftarPeserta);
10
11 function adaPesan(){
12     if(isset($_GET['pesan'])){
13         return true;
14     } else return false;
15 }
16
17 function tampilkanHalamanIndex($pesan, $daftarPeserta){
18     require_once 'view/view-index.php';
19 }
```

3. Berikutnya kita ubah view ‘view-index.php’ dengan menyisipkan kode berikut:



```
view-index.php X
rapidtest > view > view-index.php > ...
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <title>Pendaftaran Peserta Rapid Test</title>
6 </head>
7 <body>
8   <h1><?= "Selamat Datang. " . $pesan ?></h1>
9   <table>
10    <tr>
11      <td>Email</td>
12      <td>Nama</td>
13    </tr>
14    <?php foreach($daftarPeserta as $peserta){ ?>
15      <tr>
16        <td><?= $peserta->getEmail() ?></td>
17        <td><?= $peserta->getNama() ?></td>
18      </tr>
19    <?php } ?>
20  </table>
21  <p><a href="tambah.php">Tambah Peserta</a></p>
22 </body>
23 </html>
24
```

4. Setelah selesai, cobalah menambahkan daftar peserta baru. Setelah itu periksa apakah telah berhasil ditampilkan.

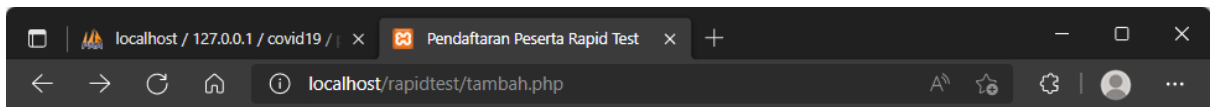


Selamat Datang.

Email	Nama
123@123.com	123
456@456.com	456

slamet@domain.com Slamet"); TRUNCATE TABLE peserta;#

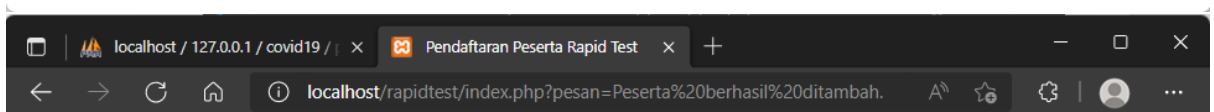
[Tambah Peserta](#)



Formulir Pendaftaran Rapid Test

Email:

Nama:



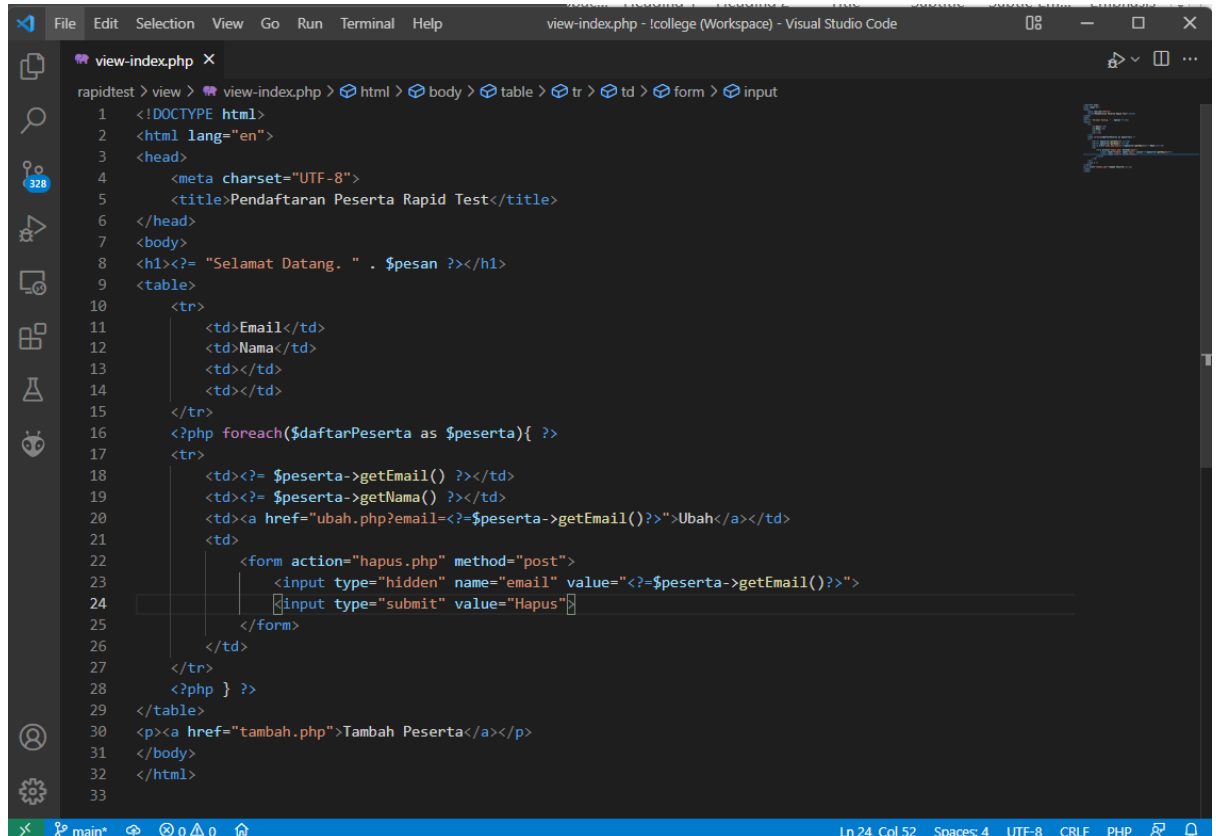
Selamat Datang. Peserta berhasil ditambah.

Email	Nama
123@123.com	123
456@456.com	456
daffa@123.com	daffa

slamet@domain.com Slamet"); TRUNCATE TABLE peserta;#

[Tambah Peserta](#)

1. Buka berkas `view-index.php`, kemudian sisipkan kode seperti di bawah ini:

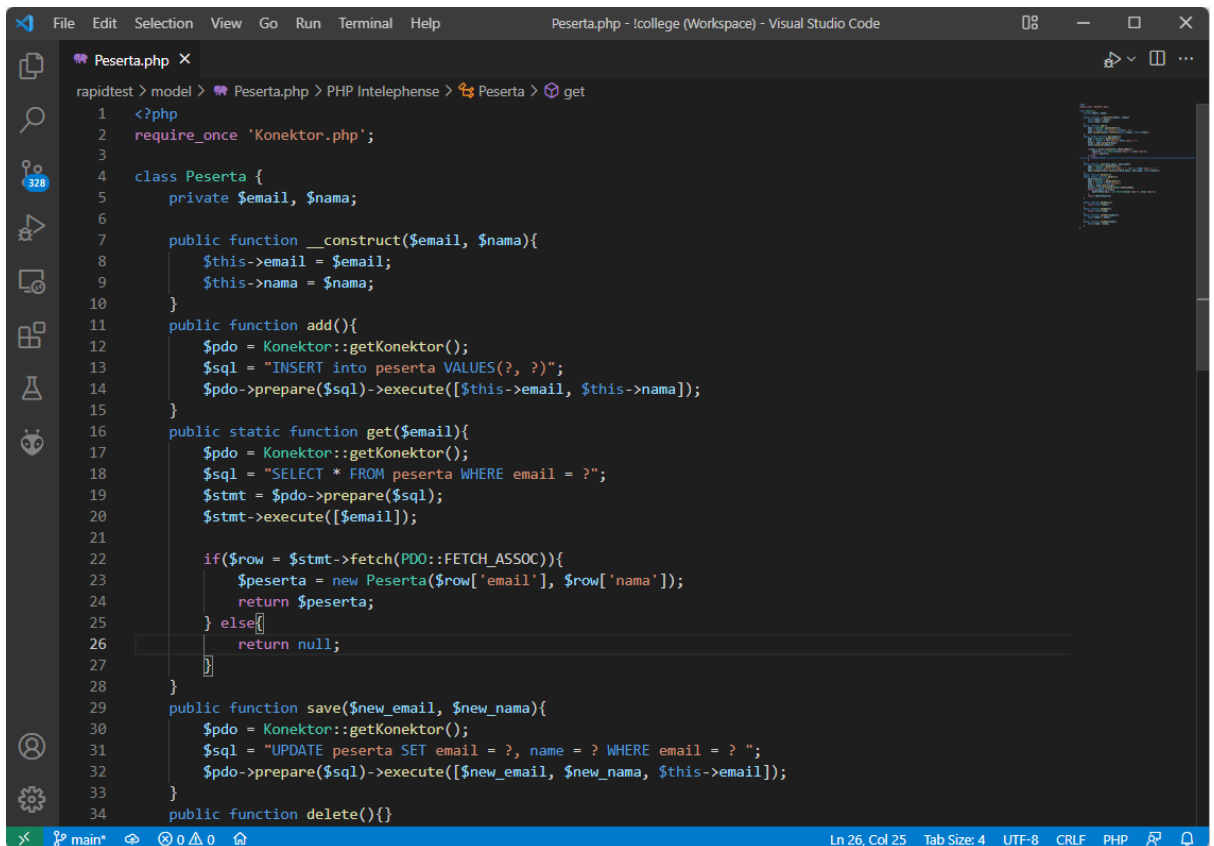


```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <title>Pendaftaran Peserta Rapid Test</title>
6 </head>
7 <body>
8   <h1><?= "Selamat Datang. " . $pesan ?></h1>
9   <table>
10    <tr>
11      <td>Email</td>
12      <td>Nama</td>
13    </tr>
14    <tr>
15      <td><?= $peserta->getEmail() ?></td>
16      <td><?= $peserta->getNama() ?></td>
17      <td><a href="ubah.php?email=<?= $peserta->getEmail() ?>">Ubah</a></td>
18      <td>
19        <form action="hapus.php" method="post">
20          <input type="hidden" name="email" value="<?= $peserta->getEmail() ?>">
21          <input type="submit" value="Hapus">
22        </form>
23      </td>
24    </tr>
25  </table>
26  <p><a href="tambah.php">Tambah Peserta</a></p>
27 </body>
28 </html>
```

2. Dapat dilihat bahwa kita menggunakan tautan dengan *query string* email untuk mengubah data, sedangkan kita menggunakan formulir berisi sebuah *hidden field* email untuk menghapus data berdasarkan email
1. Buka berkas 'Peserta.php', kemudian pada kelas Peserta kita akan mengisi *method* `save()` yang sebelumnya sudah kita buat namun masih kosong. Isilah *method* tersebut dengan kode seperti di bawah ini:

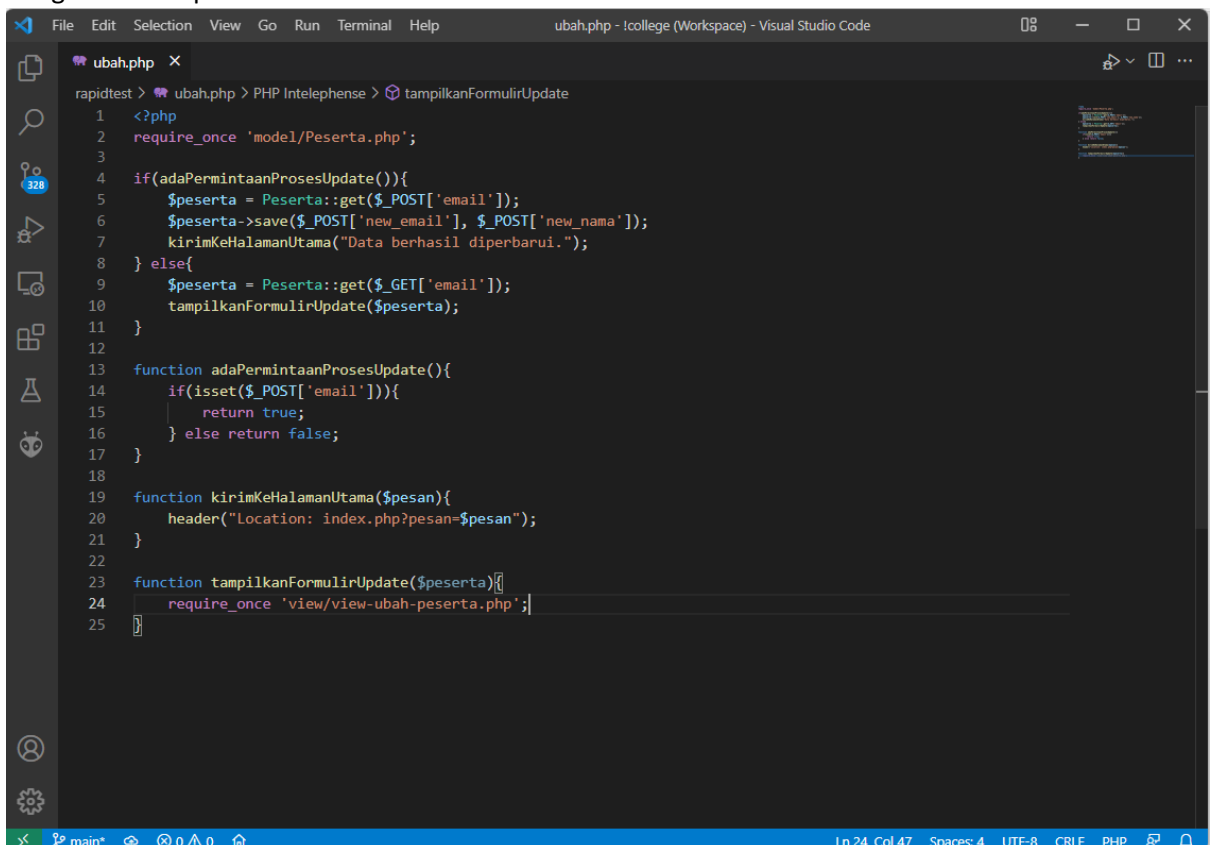
```
1 <?php
2 require_once 'Konektor.php';
3
4 class Peserta {
5     private $email, $nama;
6
7     public function __construct($email, $nama){
8         $this->email = $email;
9         $this->nama = $nama;
10    }
11
12    public function add(){
13        $pdo = Konektor::getKonektor();
14        $sql = "INSERT into peserta VALUES(?, ?)";
15        $pdo->prepare($sql)->execute([$this->email, $this->nama]);
16    }
17
18    public function get($email){
19        public function save($new_email, $new_nama){
20            $pdo = Konektor::getKonektor();
21            $sql = "UPDATE peserta SET email = ?, name = ? WHERE email = ? ";
22            $pdo->prepare($sql)->execute([$new_email, $new_nama, $this->email]);
23        }
24
25        public function delete(){}
26        public static function getAll(){
27            $daftarPeserta = [];
28            $pdo = Konektor::getKonektor();
29            $sql = "SELECT * FROM peserta";
30            $stmt = $pdo->query($sql);
31            $result = $stmt->fetchAll(PDO::FETCH_ASSOC);
32            foreach($result as $row){
33                $daftarPeserta[] = new Peserta($row['email'], $row['nama']);
34            }
35            return $daftarPeserta;
36        }
37    }
38 }
```

2. Perlu diingat bahwa saat kita membuat tabel peserta, kolom email adalah PRIMARY Key, sehingga *query* UPDATE menjadi seperti di atas. Dari *query* tersebut, dapat disimpulkan bahwa kita mengizinkan pengubahan data, baik perubahan pada email maupun nama.
3. Karena kita mengizinkan perubahan pada semua kolom, maka secara tampilan formulir nantinya, formulir untuk proses UPDATE (pengubahan data) hampir sama dengan formulir untuk proses INSERT (penambahan data) yang telah kita buat sebelumnya. Perbedaan utamanya adalah pada formulir penambahan data, kolom isian sudah berisi data awal yang akan kita ubah. Oleh karena itu, kita perlu mengimplementasikan fungsi untuk mengambil data peserta (tunggal). Pada perkuliahan sebelumnya, kita telah menyiapkan fungsi `get($email)`, kita akan ubah fungsi tersebut menjadi *static* dan kita lengkapi fungsi tersebut seperti kode di bawah ini:



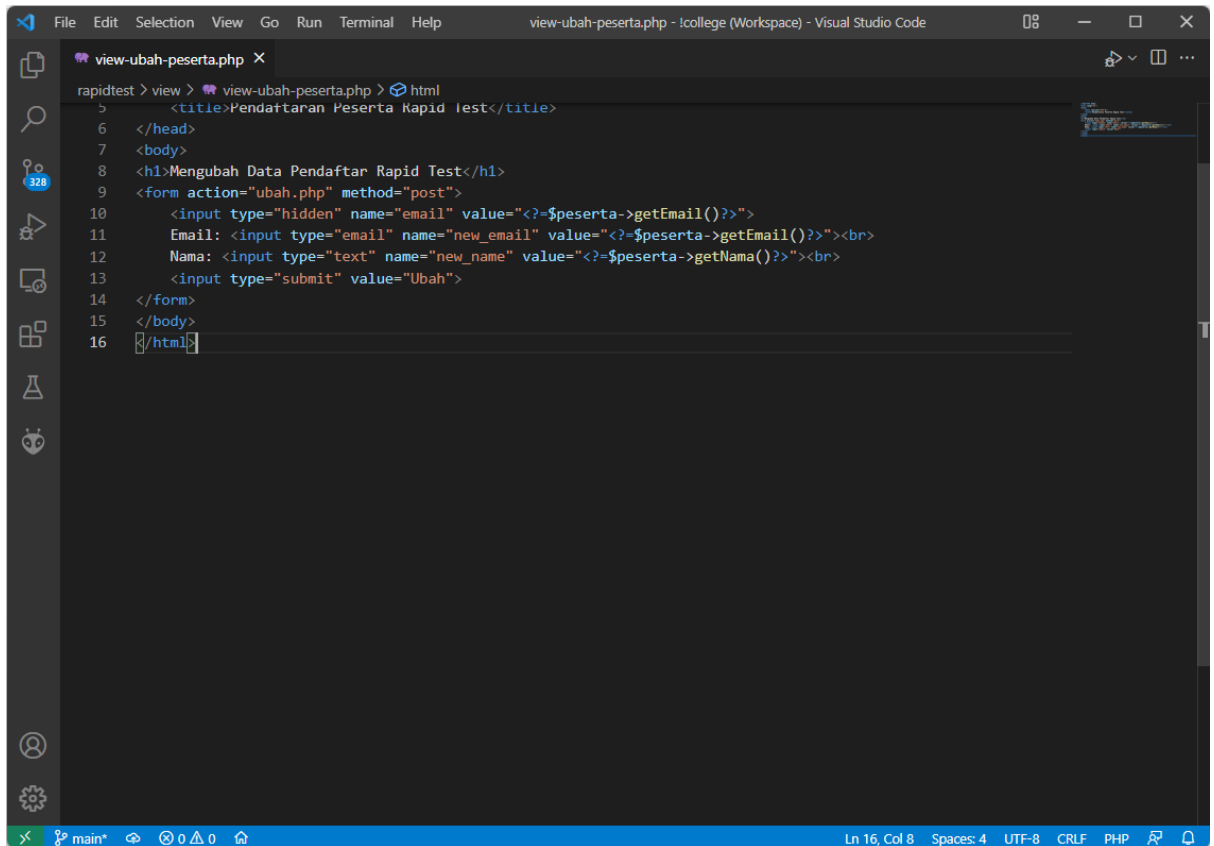
```
1 <?php
2 require_once 'Konektor.php';
3
4 class Peserta {
5     private $email, $nama;
6
7     public function __construct($email, $nama){
8         $this->email = $email;
9         $this->nama = $nama;
10    }
11    public function add(){
12        $pdo = Konektor::getKonektor();
13        $sql = "INSERT into peserta VALUES(?, ?)";
14        $pdo->prepare($sql)->execute([$this->email, $this->nama]);
15    }
16    public static function get($email){
17        $pdo = Konektor::getKonektor();
18        $sql = "SELECT * FROM peserta WHERE email = ?";
19        $stmt = $pdo->prepare($sql);
20        $stmt->execute([$email]);
21
22        if($row = $stmt->fetch(PDO::FETCH_ASSOC)){
23            $peserta = new Peserta($row['email'], $row['nama']);
24            return $peserta;
25        } else{
26            return null;
27        }
28    }
29    public function save($new_email, $new_nama){
30        $pdo = Konektor::getKonektor();
31        $sql = "UPDATE peserta SET email = ?, name = ? WHERE email = ? ";
32        $pdo->prepare($sql)->execute([$new_email, $new_nama, $this->email]);
33    }
34    public function delete(){}
}
```

4. Sampai di sini, model Peserta.php telah siap melayani proses *update* data. Sekarang kita beralih ke bagian *controller*. Buatlah *controller* baru dengan nama *ubah.php*, kemudian isi dengan kode seperti di bawah ini:



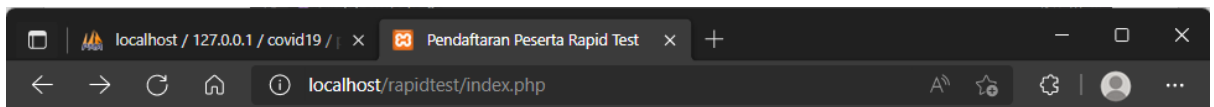
```
1 <?php
2 require_once 'model/Peserta.php';
3
4 if(adaPermintaanProsesUpdate()){
5     $peserta = Peserta::get($_POST['email']);
6     $peserta->save($_POST['new_email'], $_POST['new_nama']);
7     kirimKeHalamanUtama("Data berhasil diperbarui.");
8 } else{
9     $peserta = Peserta::get($_GET['email']);
10    tampilkanFormulirUpdate($peserta);
11 }
12
13 function adaPermintaanProsesUpdate(){
14     if(isset($_POST['email'])){
15         return true;
16     } else return false;
17 }
18
19 function kirimKeHalamanUtama($pesan){
20     header("Location: index.php?pesan=$pesan");
21 }
22
23 function tampilkanFormulirUpdate($peserta){
24     require_once 'view/view-ubah-peserta.php';
25 }
```

5. Sebagaimana *query* yang kita definisikan pada model Peserta.php, ada tiga parameter yang harus dipelihara, yaitu email (email asal sebelum diubah, sebagai *primary key*), email baru dan nama baru. Semuanya telah kita tangani pada *controller* di kode poin nomor 4 di atas.
6. Selanjutnya kita perlu membuat *view* untuk menampilkan formulir *update*. Silahkan saudara buat berkas dengan nama 'view-ubah-peserta.php' kemudian isilah kode seperti di bawah ini:



```
view-ubah-peserta.php X
rapidtest > view > view-ubah-peserta.php > html
5 <title>Pendaftaran Peserta Kapid Test</title>
6 </head>
7 <body>
8 <h1>Mengubah Data Pendaftar Rapid Test</h1>
9 <form action="ubah.php" method="post">
10 <input type="hidden" name="email" value="<?=$peserta->getEmail()?">
11 Email: <input type="email" name="new_email" value="<?=$peserta->getEmail()?"><br>
12 Nama: <input type="text" name="new_name" value="<?=$peserta->getNama()?"><br>
13 <input type="submit" value="Ubah">
14 </form>
15 </body>
16 </html>
```

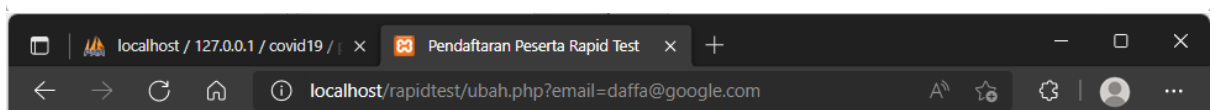
7. Jika diperhatikan, kode formulir ubah peserta sangat mirip dengan formulir tambah peserta (saudara bisa langsung *copy-paste* kemudian melakukan sedikit penyesuaian). Pada formulir ubah peserta, kita menggunakan *hidden input* untuk menyimpan data email asal sebagai *primary key* pada proses *update* agar nilainya tidak diubah oleh pengguna, sedangkan untuk email baru dan nama baru menggunakan *input* teks yang dapat dimanipulasi dengan mudah oleh pengguna
8. Silahkan saudara coba jalankan di peramban web untuk memastikan semua telah berjalan sesuai harapan



Selamat Datang.

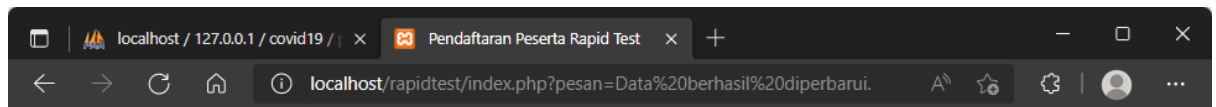
Email	Nama		
123@123.com	123	Ubah	<input type="button" value="Hapus"/>
456@456.com	456	Ubah	<input type="button" value="Hapus"/>
daffa@123.com	daffa	Ubah	<input type="button" value="Hapus"/>
slamet@domain.com Slamet"); TRUNCATE TABLE peserta;# Ubah <input type="button" value="Hapus"/>			

[Tambah Peserta](#)



Mengubah Data Pendaftar Rapid Test

Email:	<input type="text" value="daffa@google.com"/>
Nama:	<input type="text" value="daffa123"/>
<input type="button" value="Ubah"/>	

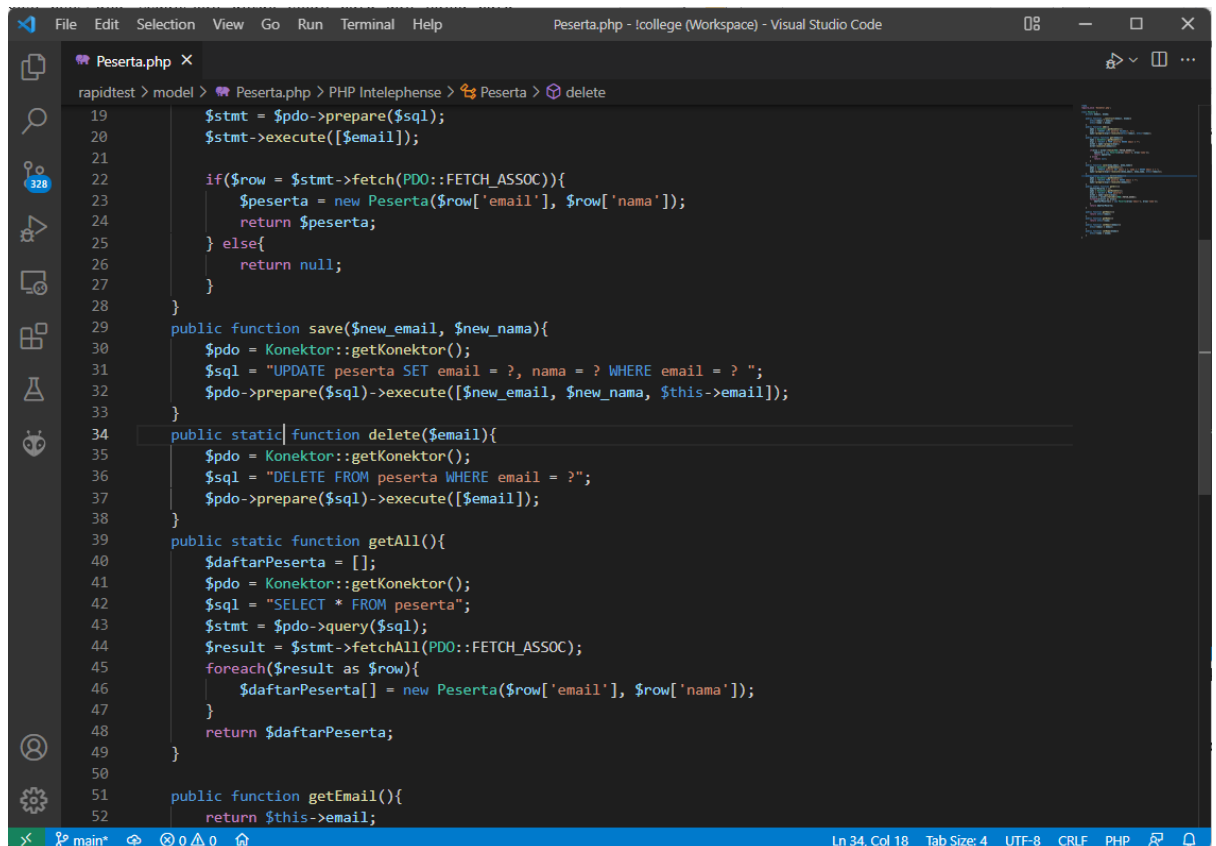


Selamat Datang. Data berhasil diperbarui.

Email	Nama	
123@123.com	123	Ubah <input type="button" value="Hapus"/>
456@456.com	456	Ubah <input type="button" value="Hapus"/>
daffa@google.com	daffa123	Ubah <input type="button" value="Hapus"/>
slamet@domain.com	Slamet	Ubah <input type="button" value="Hapus"/>

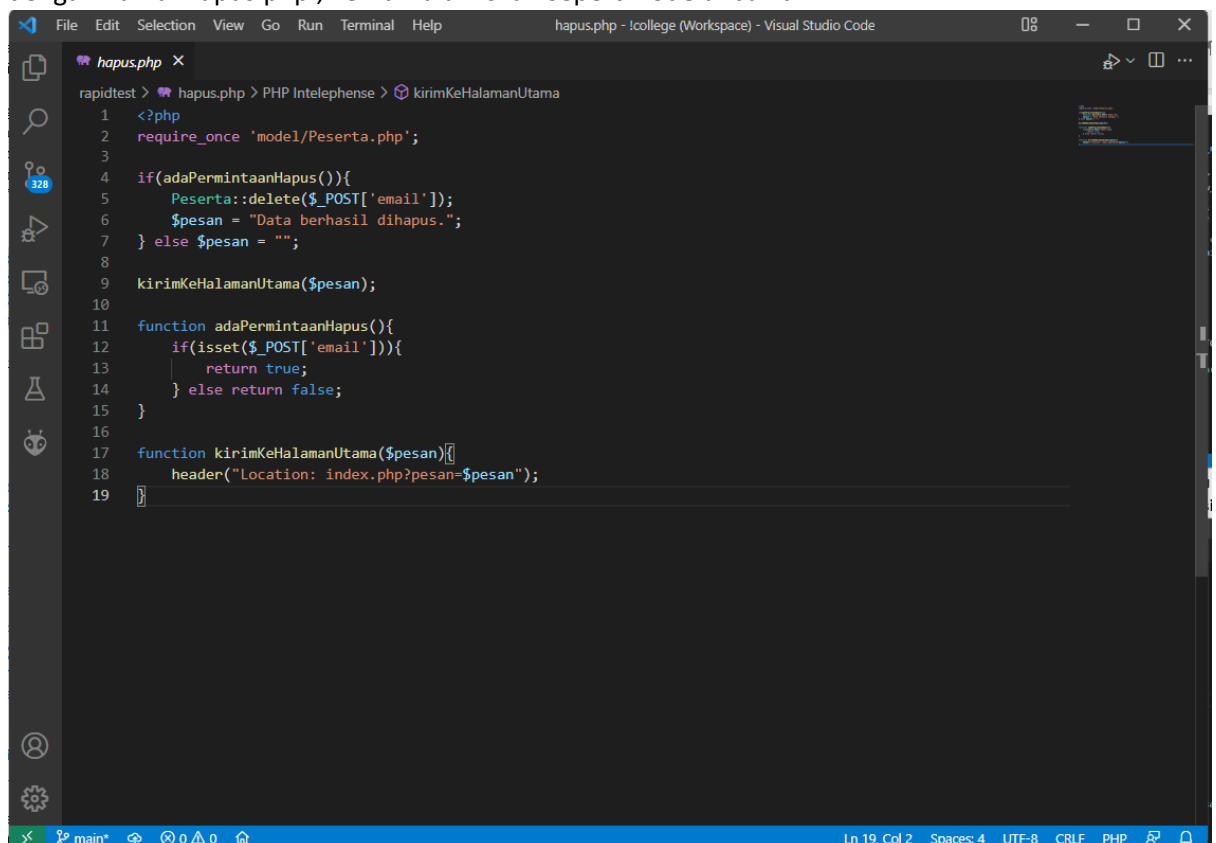
[Tambah Peserta](#)

1. Buka berkas 'Peserta.php', kemudian pada kelas peserta kita ubah *method* delete() yang sebelumnya sudah kita definisikan menjadi seperti kode di bawah ini:



```
19 $stmt = $pdo->prepare($sql);
20 $stmt->execute([$email]);
21
22 if($row = $stmt->fetch(PDO::FETCH_ASSOC)){
23     $peserta = new Peserta($row['email'], $row['nama']);
24     return $peserta;
25 } else{
26     return null;
27 }
28
29 public function save($new_email, $new_nama){
30     $pdo = Konektor::getKonektor();
31     $sql = "UPDATE peserta SET email = ?, nama = ? WHERE email = ? ";
32     $pdo->prepare($sql)->execute([$new_email, $new_nama, $this->email]);
33 }
34 public static function delete($email){
35     $pdo = Konektor::getKonektor();
36     $sql = "DELETE FROM peserta WHERE email = ?";
37     $pdo->prepare($sql)->execute([$email]);
38 }
39 public static function getAll(){
40     $daftarPeserta = [];
41     $pdo = Konektor::getKonektor();
42     $sql = "SELECT * FROM peserta";
43     $stmt = $pdo->query($sql);
44     $result = $stmt->fetchAll(PDO::FETCH_ASSOC);
45     foreach($result as $row){
46         $daftarPeserta[] = new Peserta($row['email'], $row['nama']);
47     }
48     return $daftarPeserta;
49 }
50
51 public function getEmail(){
52     return $this->email;
```

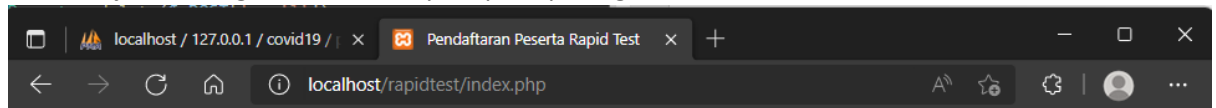
- Setelah itu, kita membuat *controller* baru untuk menghapus data. Buatlah *controller* baru dengan nama 'hapus.php', kemudian isilah seperti kode di bawah ini:



```
1 <?php
2 require_once 'model/Peserta.php';
3
4 if(adaPermintaanHapus()){
5     Peserta::delete($_POST['email']);
6     $pesan = "Data berhasil dihapus.";
7 } else $pesan = "";
8
9 kirimKeHalamanUtama($pesan);
10
11 function adaPermintaanHapus(){
12     if(isset($_POST['email'])){
13         return true;
14     } else return false;
15 }
16
17 function kirimKeHalamanUtama($pesan){
18     header("Location: index.php?pesan=$pesan");
19 }
```

- Pada kasus hapus ini, kita tidak memerlukan *view* tersendiri, karena tombol hapus kita ikutkan pada 'view-index.php'

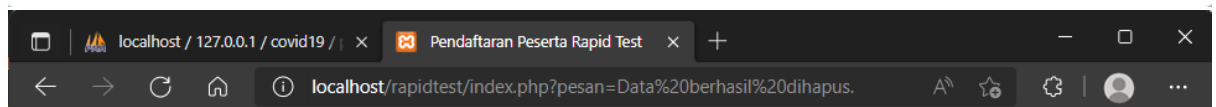
4. Silahkan saudara coba jalankan dan buka di peramban web, pastikan semua fungsionalitas telah berjalan sebagaimana mestinya seperti pada gambar 2



Selamat Datang.

Email	Nama	
123@123.com	123	Ubah <input type="button" value="Hapus"/>
456@456.com	456	Ubah <input type="button" value="Hapus"/>
daffa@google.com	daffa123	Ubah <input type="button" value="Hapus"/>
slamet@domain.com	Slamet"); TRUNCATE TABLE peserta;#	Ubah <input type="button" value="Hapus"/>

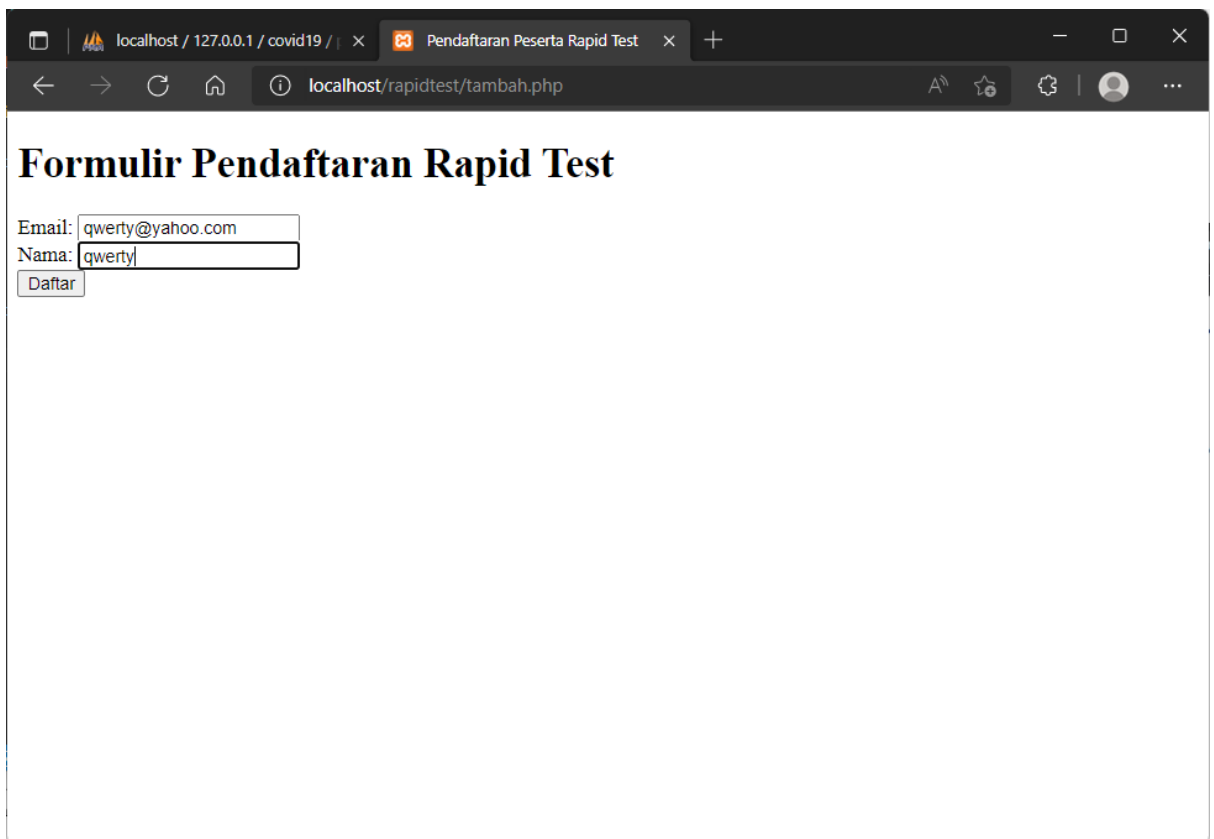
[Tambah Peserta](#)

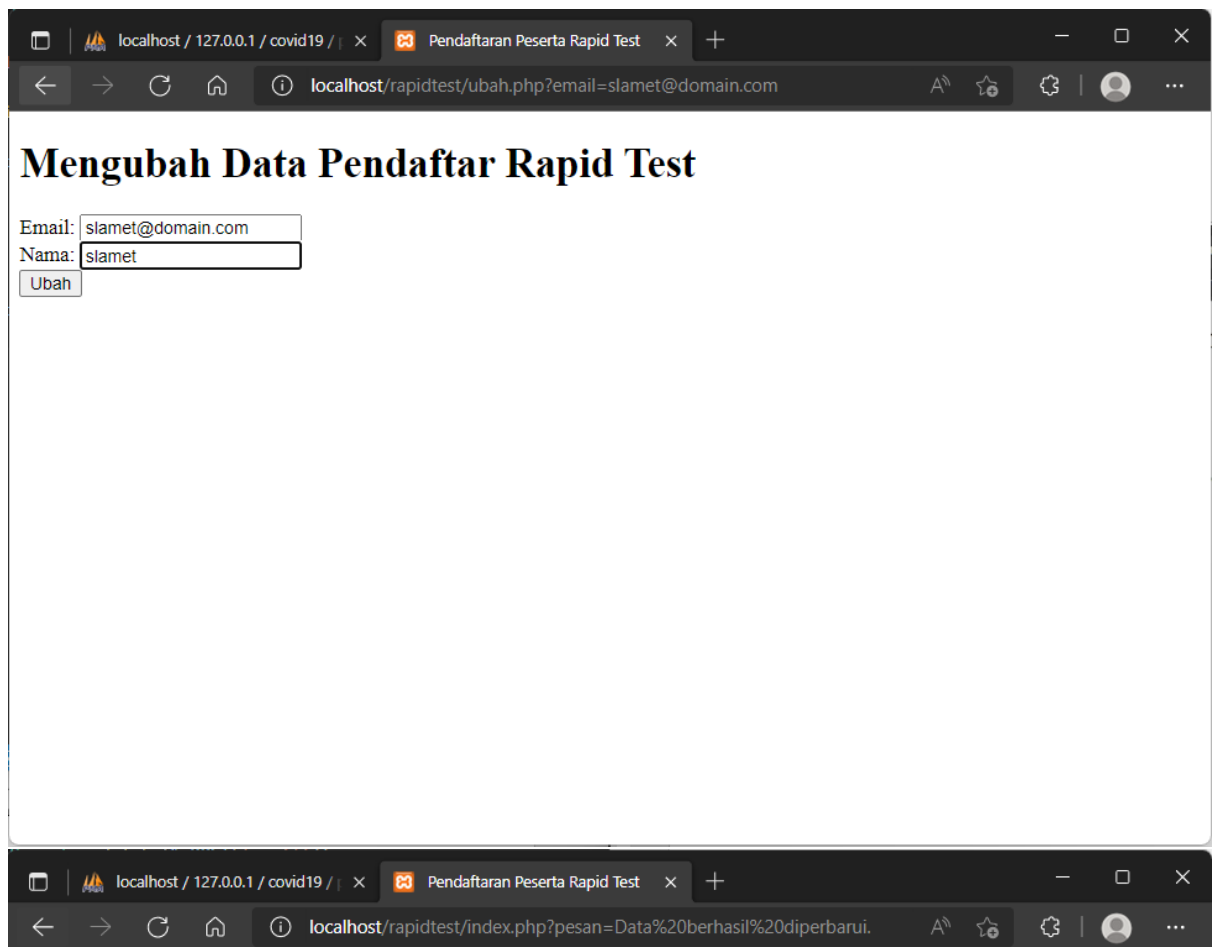


Selamat Datang. Data berhasil dihapus.

Email	Nama	
456@456.com	456	Ubah <input type="button" value="Hapus"/>
daffa@google.com	daffa123	Ubah <input type="button" value="Hapus"/>
slamet@domain.com	Slamet"); TRUNCATE TABLE peserta;#	Ubah <input type="button" value="Hapus"/>

[Tambah Peserta](#)





Selamat Datang. Data berhasil diperbarui.

Email	Nama		
456@456.com	456	Ubah	<input type="button" value="Hapus"/>
daffa@google.com	daffa123	Ubah	<input type="button" value="Hapus"/>
qwerty@yahoo.com	qwerty	Ubah	<input type="button" value="Hapus"/>
slamet@domain.com	slamet	Ubah	<input type="button" value="Hapus"/>

[Tambah Peserta](#)