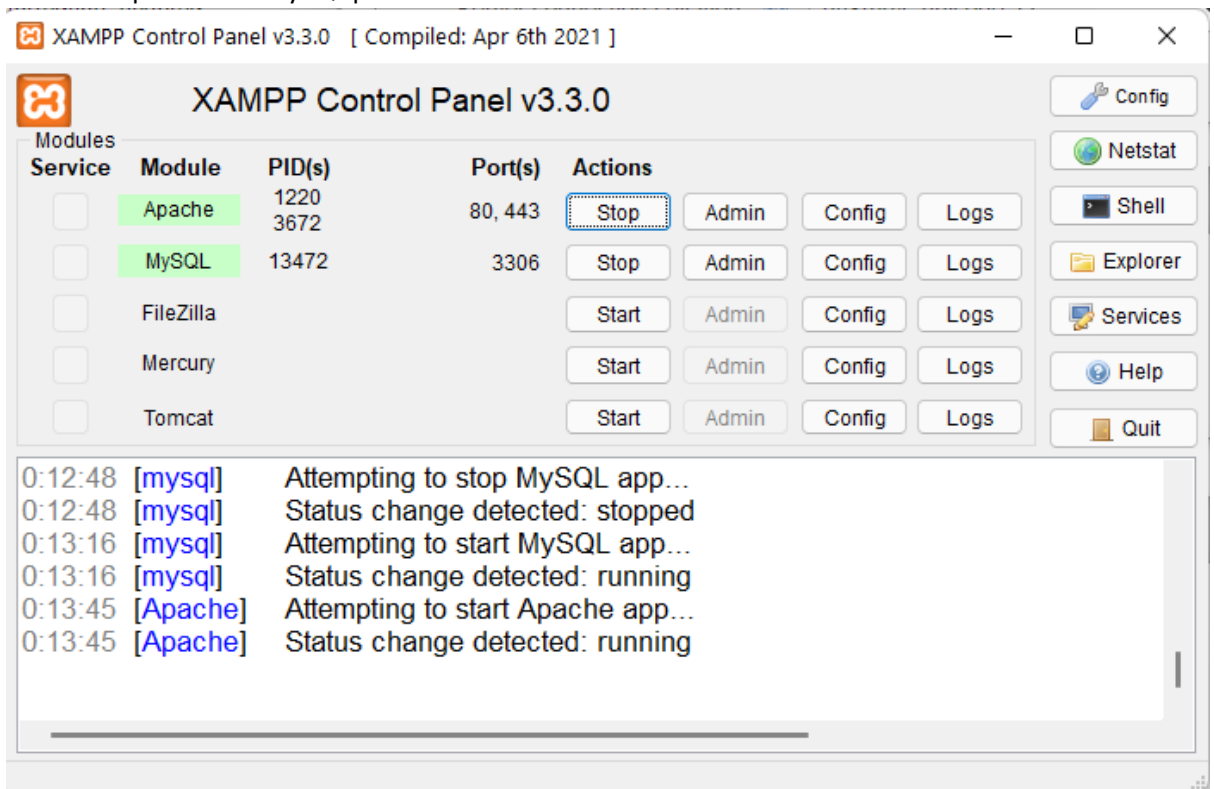
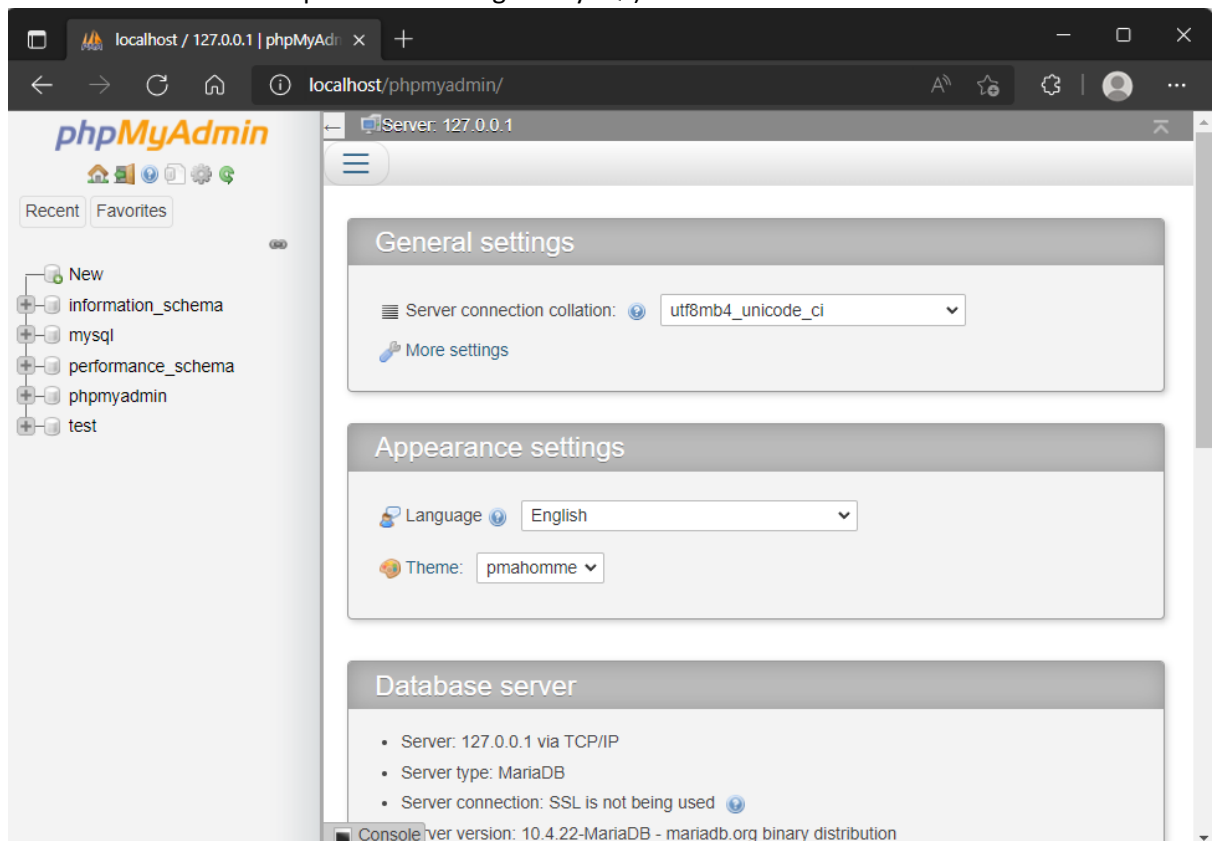


Mengakses basis data MySQL dengan PDO

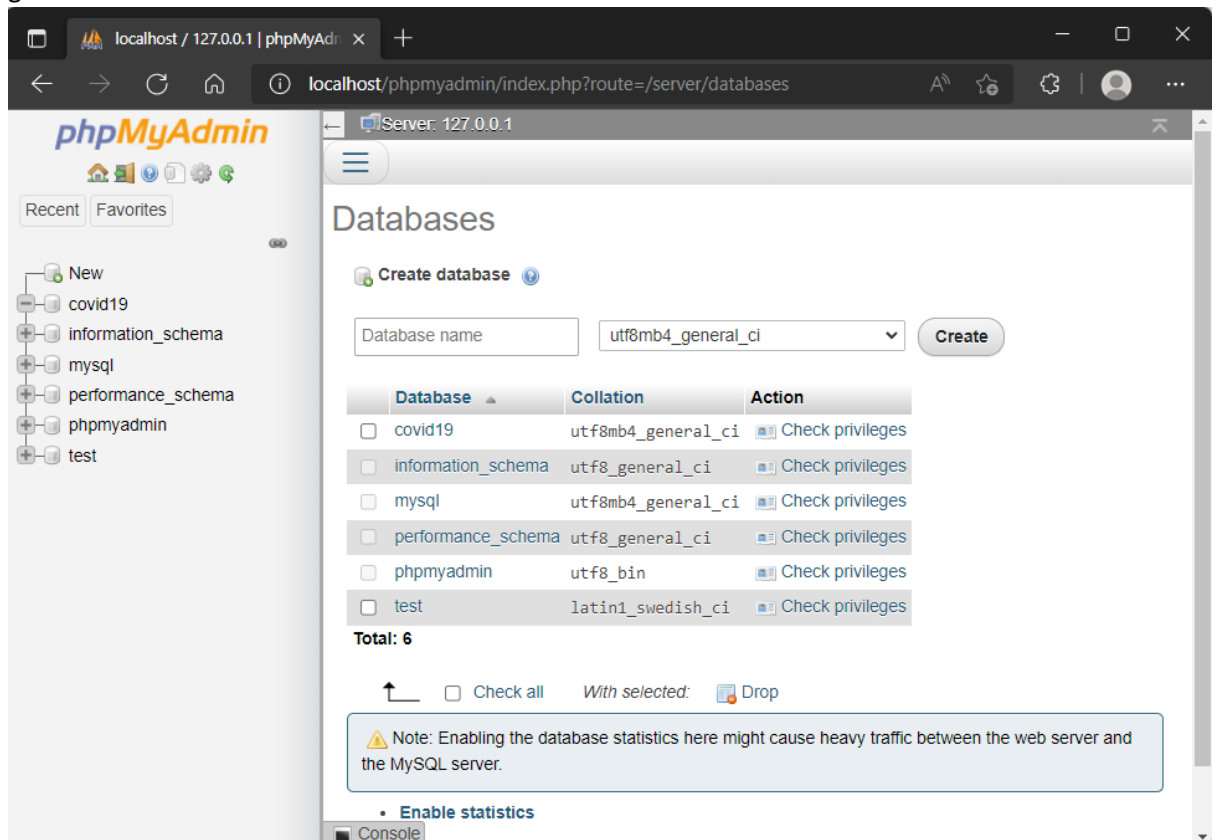
1. Jalankan Apache dan MySQL pada XAMPP



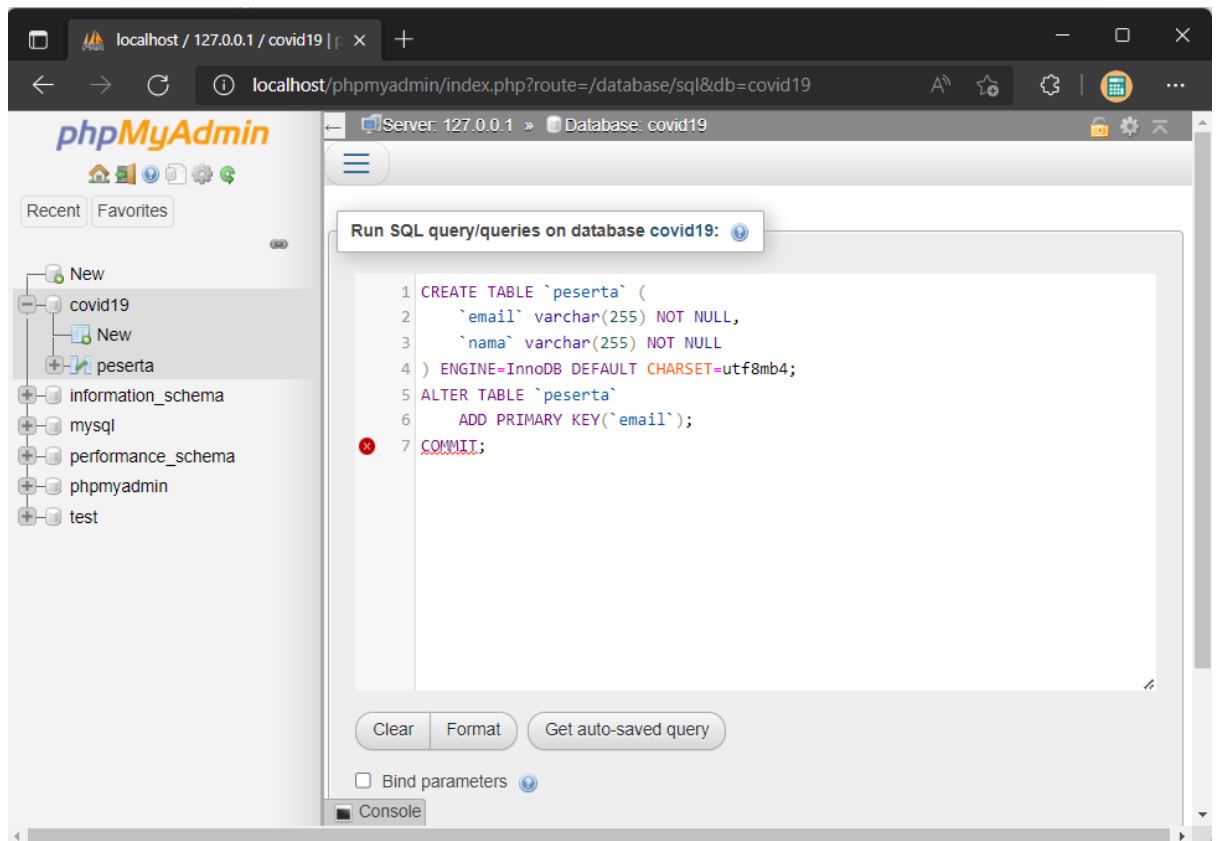
2. Buka phpMyAdmin melalui alamat <http://localhost/phpmyadmin/> (saudara juga bisa menekan tombol Admin pada XAMPP bagian MySQL)



3. Pada halaman phpMyAdmin, buatlah basis data baru dengan nama covid19, seperti pada gambar 1

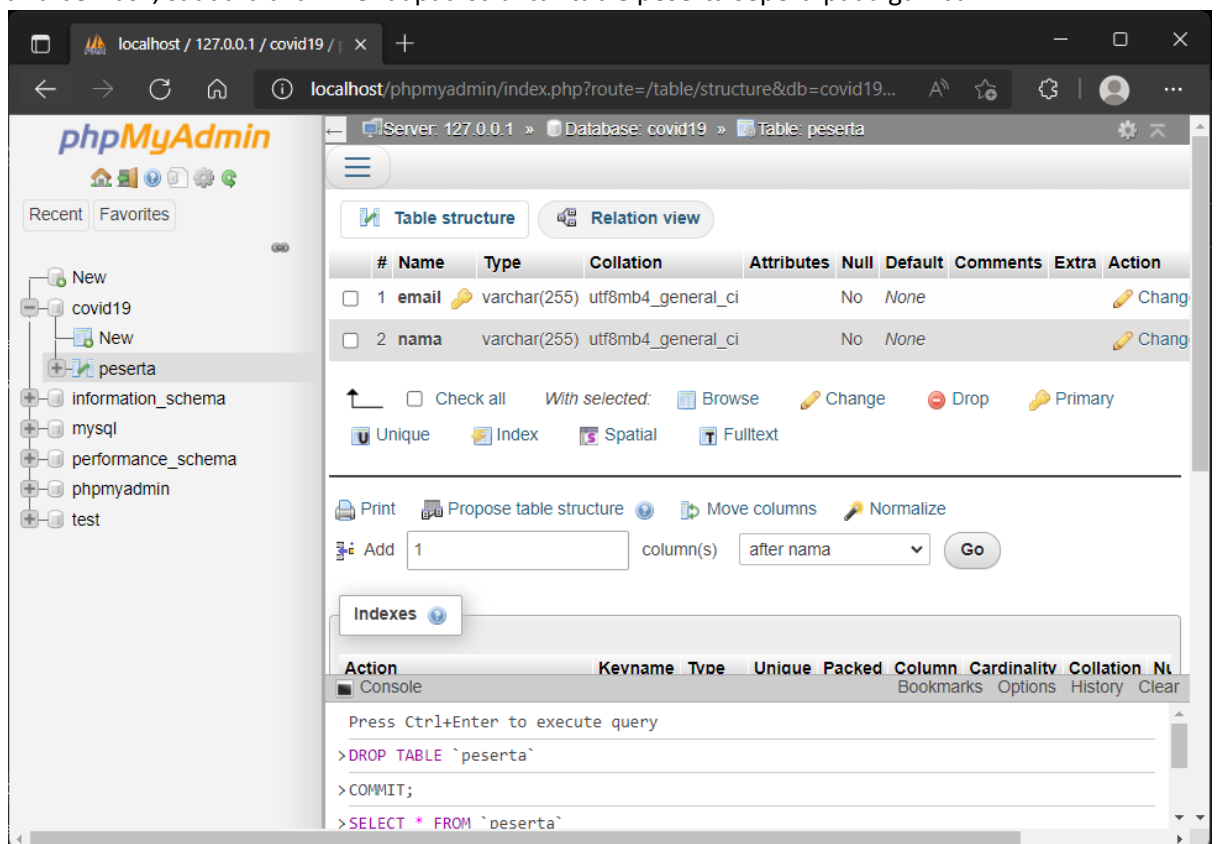


4. Pada basis data covid19 yang baru dibuat, buatlah tabel peserta dengan rincian seperti kode berikut ini (saudara dapat menuliskan kode ini pada kolom SQL di phpMyAdmin)



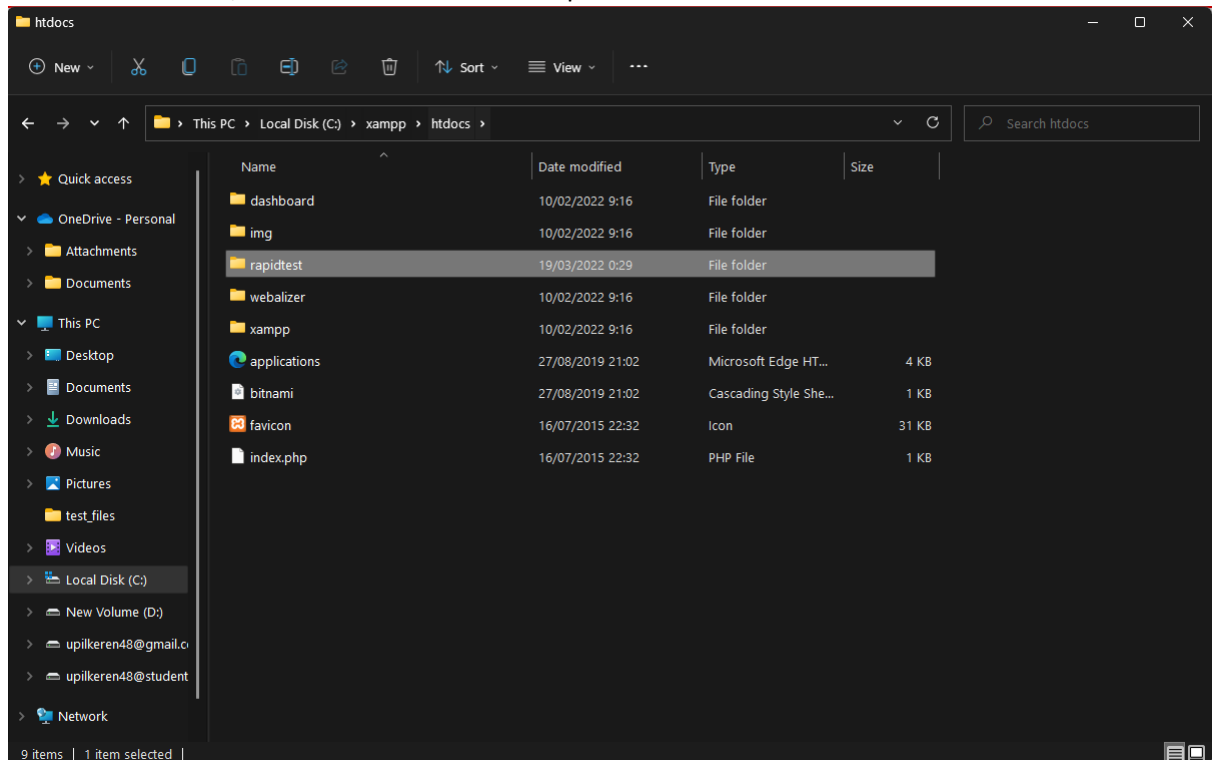
5.

Jika berhasil, saudara akan mendapati struktur table peserta seperti pada gambar 2.

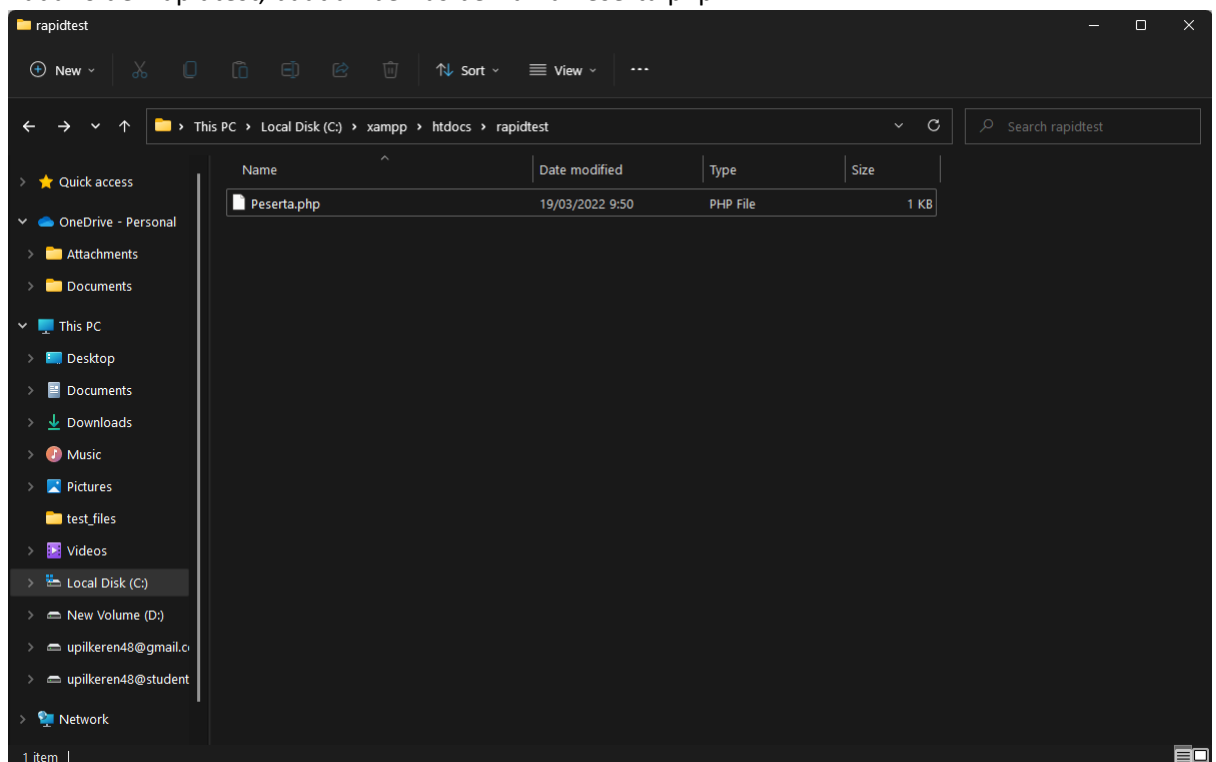


Object-relational mapping (ORM)

1. Pada folder htdocs, buatlah folder bernama: rapidtest



2. Pada folder rapidtest, buatlah berkas bernama Peserta.php



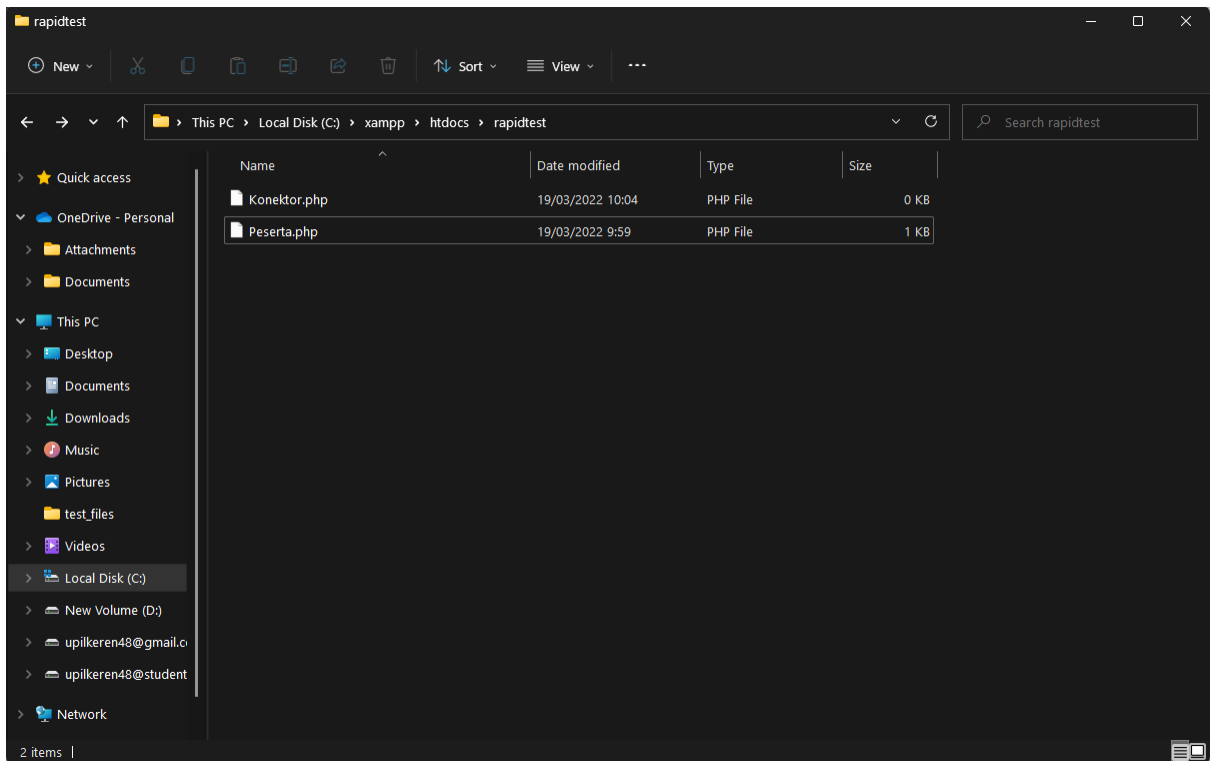
3. Tambahkan atribut dan tanggung jawab/operasi pada kelas Peserta, sehingga menjadi kode di bawah ini:

```
1 <?php
2
3 class Peserta {
4     private $email, $nama;
5
6     public function add(){}
7     public function get($email){}
8     public function save(){}
9     public function delete(){}
10    public static function getAll(){}
11
12    public function getEmail(){
13        return $this->email;
14    }
15    public function getNama(){
16        return $this->nama;
17    }
18    public function setEmail($email){
19        $this->email = $email;
20    }
21    public function setNama($nama){
22        $this->nama = $nama;
23    }
24 }
```

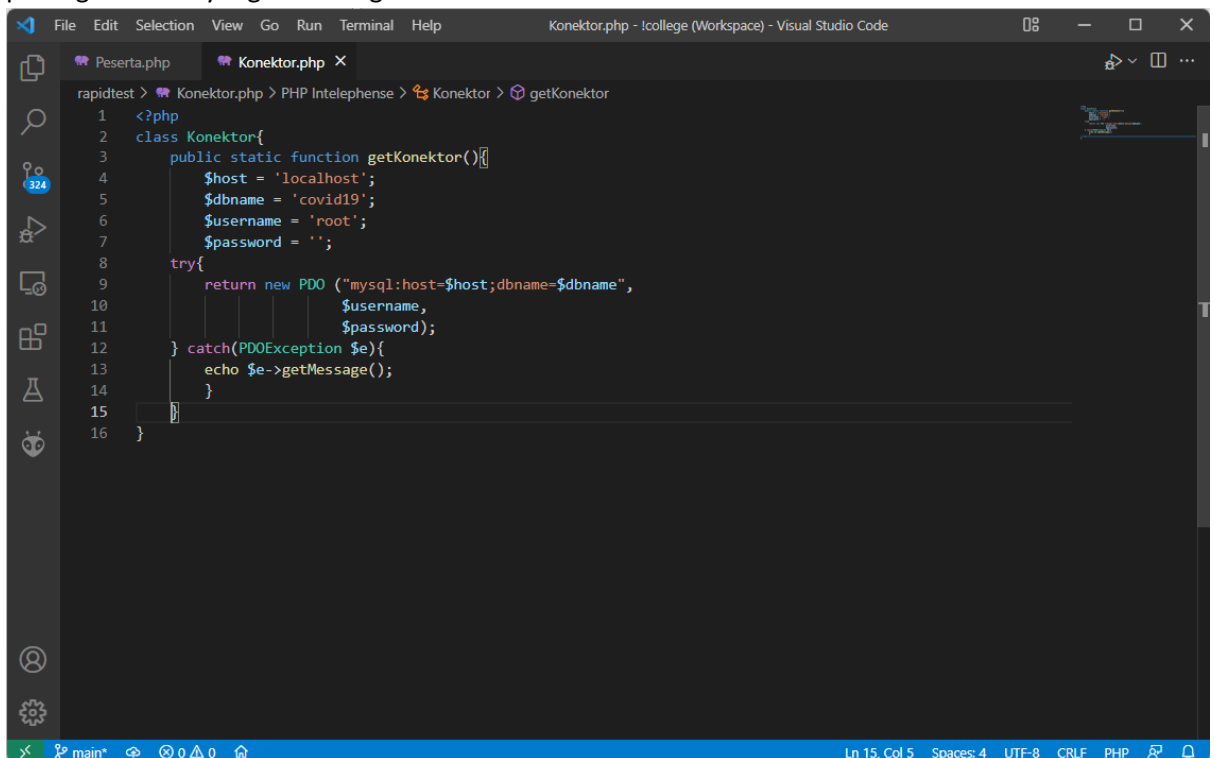
4. Dari kode di atas, sementara kita biarkan kosong isi (*body*) dari *methods*: `add()`, `get(email)`, `save()`, `delete()` dan `getAll()`. Jika diperhatikan, *method* `getAll()` berjenis *static*. *Method* `getAll()` ini nantinya disiapkan untuk mengambil data semua peserta yang terdaftar. Menurut pendapat saudara, apa kira-kira alasan *method* ini dijadikan *static*? Sertakan jawaban saudara pada saat mengisi presensi.

Akses basis data dengan konektor PDO

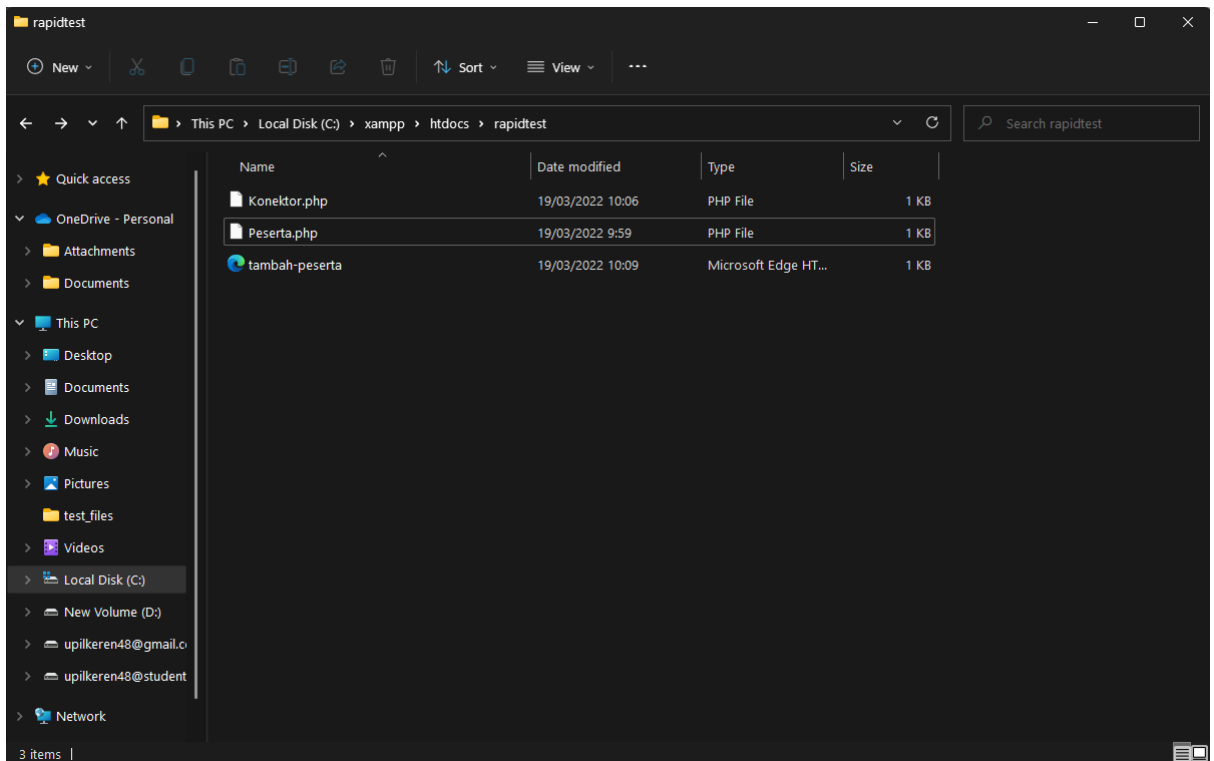
1. Pada folder `rapidtest`, buatlah berkas bernama: `Konektor.php`



- Isilah berkas tersebut dengan kode seperti di bawah ini dengan menyesuaikan kondisi perangkat lunak yang saudara gunakan:



- Pada folder rapidtest, buat berkas dengan nama tambah-peserta.html



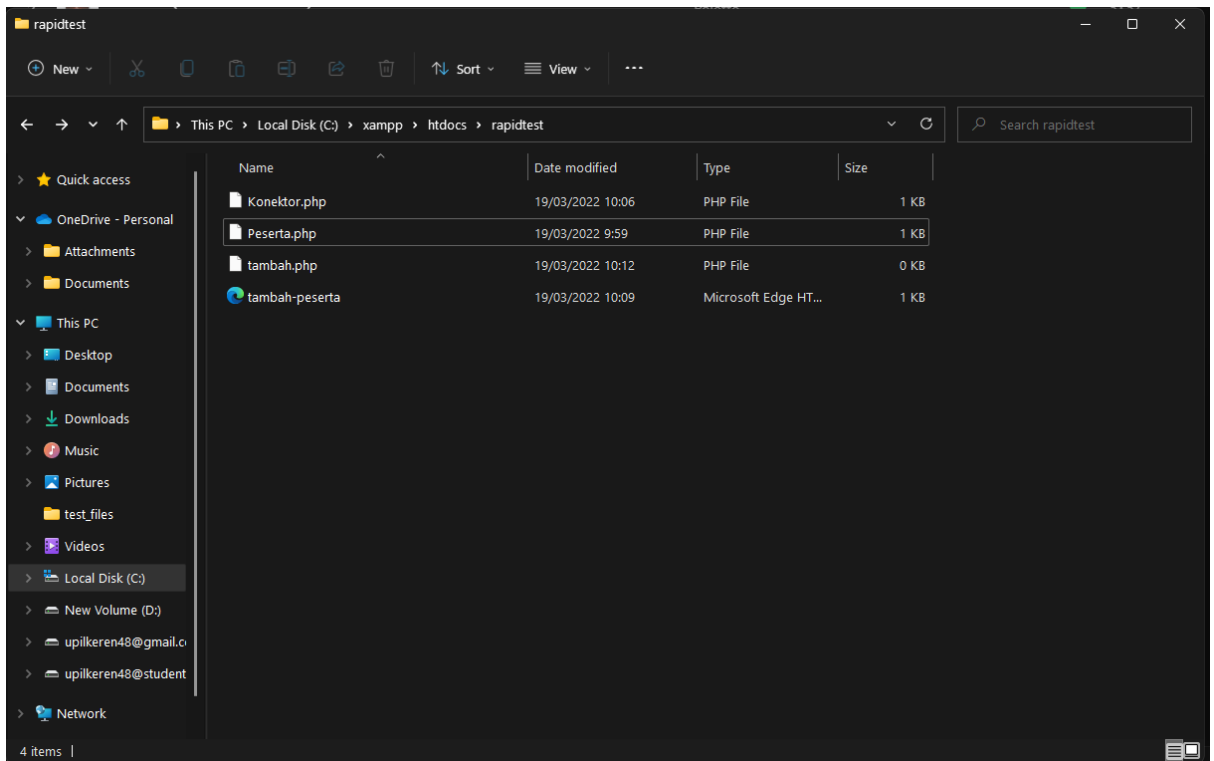
4. Berkas tersebut berisi formulir dengan dua buah *input*, yaitu email dan nama. Formulir akan dikirimkan menuju tambah.php dengan *method* POST. Isilah berkas tambah-peserta.php dengan kode sebagai berikut:

```

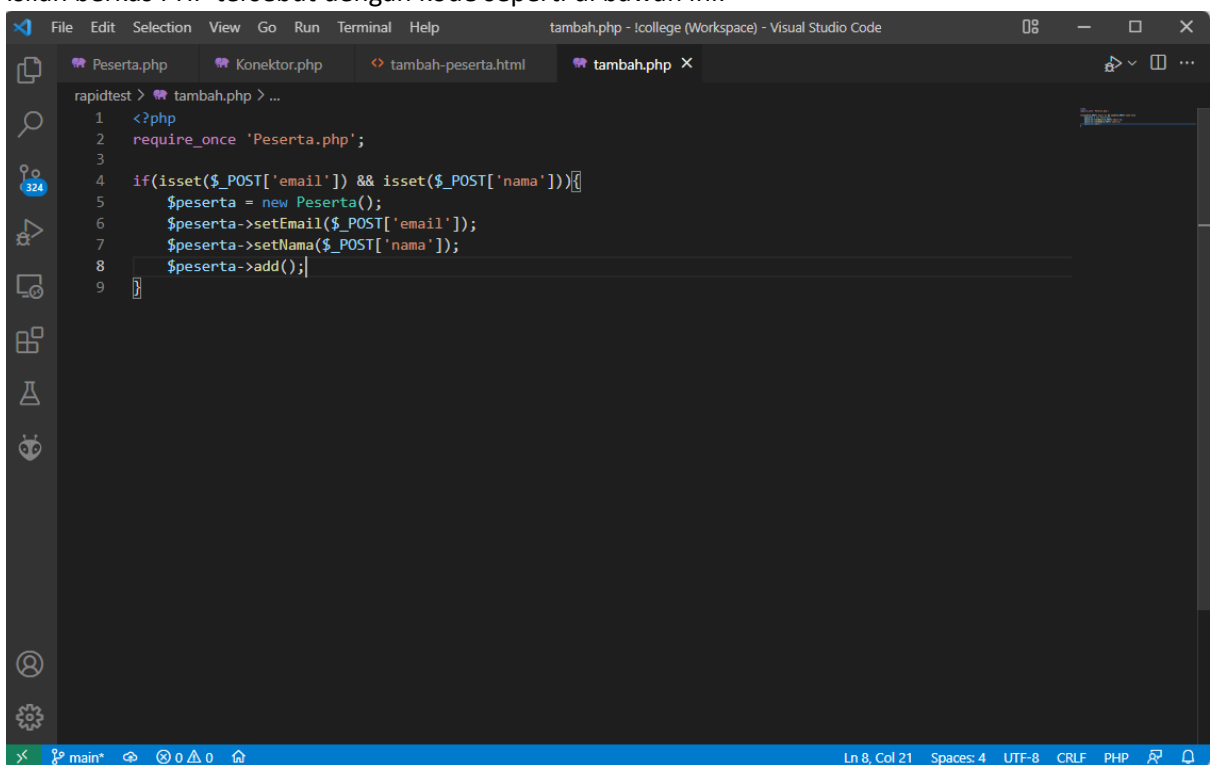
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <title>Pendaftaran Peserta Rapid Test</title>
6 </head>
7 <body>
8 <h1>Formulir Pendaftaran Rapid Test</h1>
9 <form action="tambah.php" method="post">
10   Email: <input type="email" name="email"><br>
11   Nama: <input type="nama" name="nama"><br>
12   <input type="submit" value="Daftar">
13 </form>
14 </body>
15 </html>

```

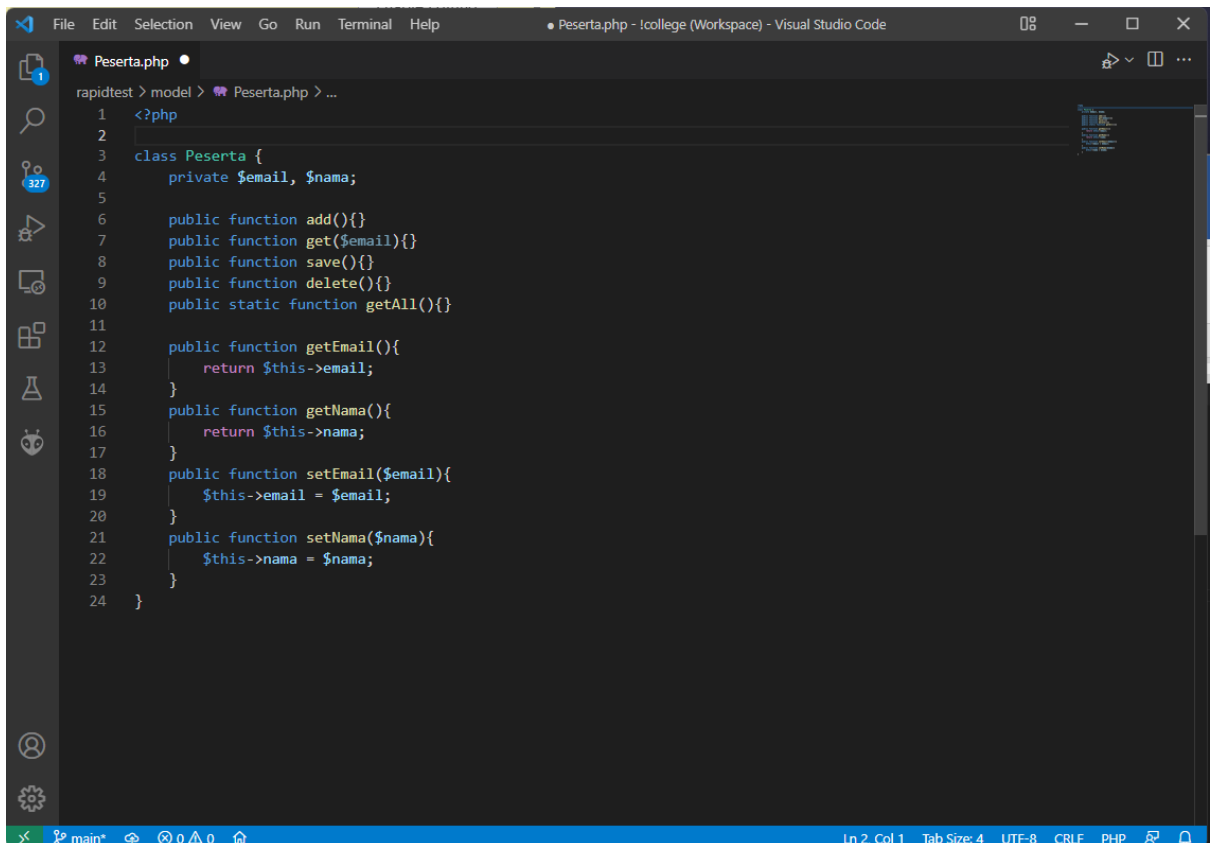
5. Setelah itu, buatlah berkas PHP dengan nama tambah.php untuk menerima kiriman formulir pendaftaran yang baru saja kita buat.



6. Isilah berkas PHP tersebut dengan kode seperti di bawah ini:

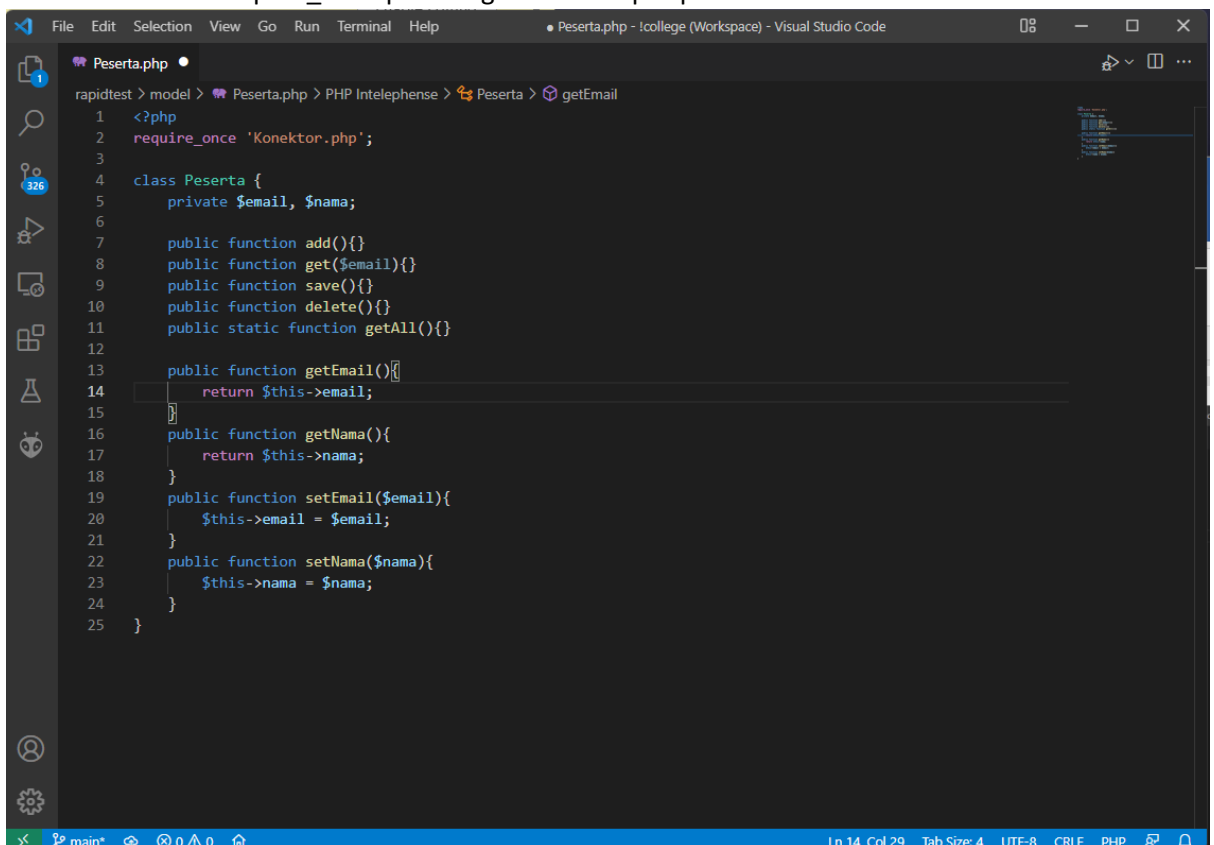


7. Buka berkas Peserta.php kemudian ubahlah isinya sesuai instruksi di bawah.



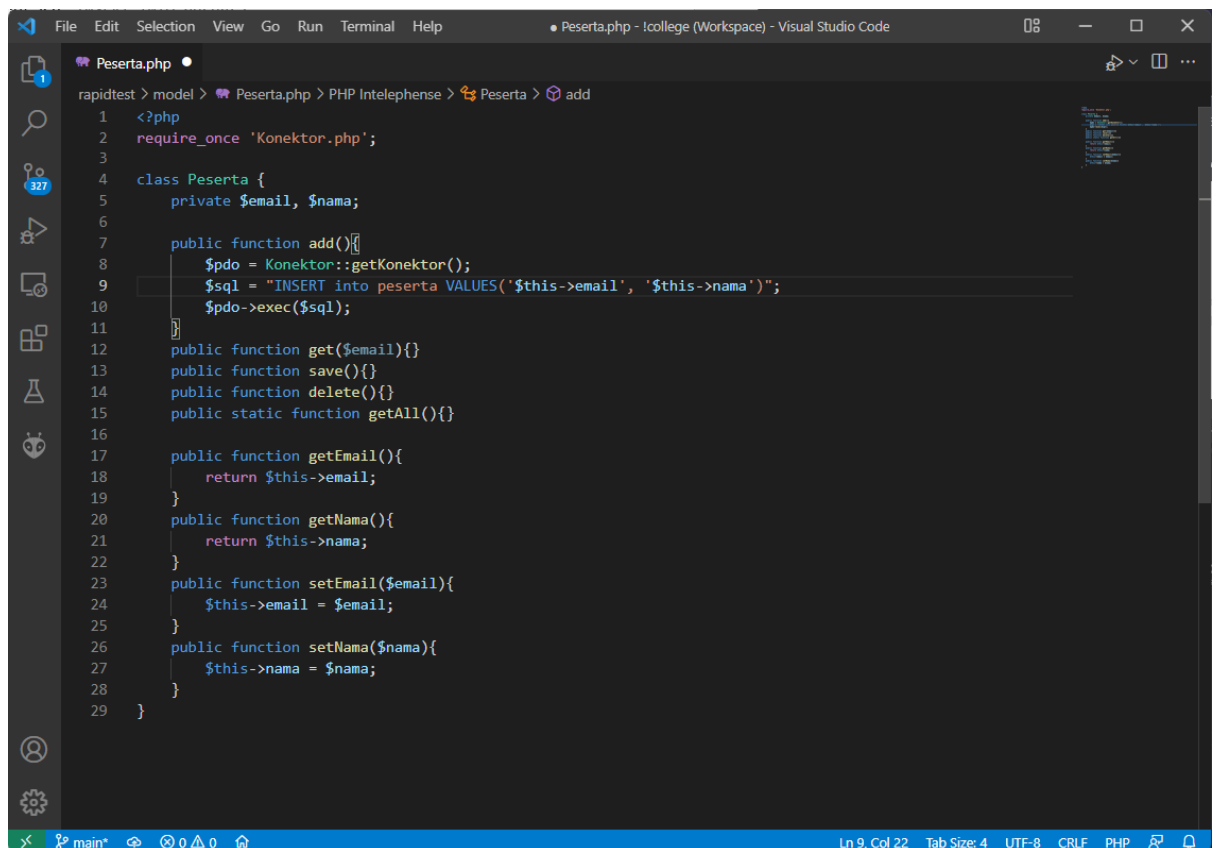
```
1 <?php
2
3 class Peserta {
4     private $email, $nama;
5
6     public function add(){}
7     public function get($email){}
8     public function save(){}
9     public function delete(){}
10    public static function getAll(){}
11
12    public function getEmail(){
13        return $this->email;
14    }
15    public function getNama(){
16        return $this->nama;
17    }
18    public function setEmail($email){
19        $this->email = $email;
20    }
21    public function setNama($nama){
22        $this->nama = $nama;
23    }
24 }
```

8. Kita akan berinteraksi dengan basis data, oleh karena itu kita membutuhkan konektor basis data. Tambahkan `require_once` pada bagian awal skip seperti kode di bawah ini:



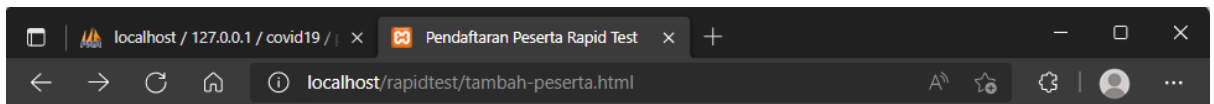
```
1 <?php
2 require_once 'Konektor.php';
3
4 class Peserta {
5     private $email, $nama;
6
7     public function add(){}
8     public function get($email){}
9     public function save(){}
10    public function delete(){}
11    public static function getAll(){}
12
13    public function getEmail(){
14        return $this->email;
15    }
16    public function getNama(){
17        return $this->nama;
18    }
19    public function setEmail($email){
20        $this->email = $email;
21    }
22    public function setNama($nama){
23        $this->nama = $nama;
24    }
25 }
```

9. Setelah itu, pada bagian *method* `add()`, kita isikan kode seperti berikut ini (warna abu-abu menandakan kode yang ditambahkan):



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

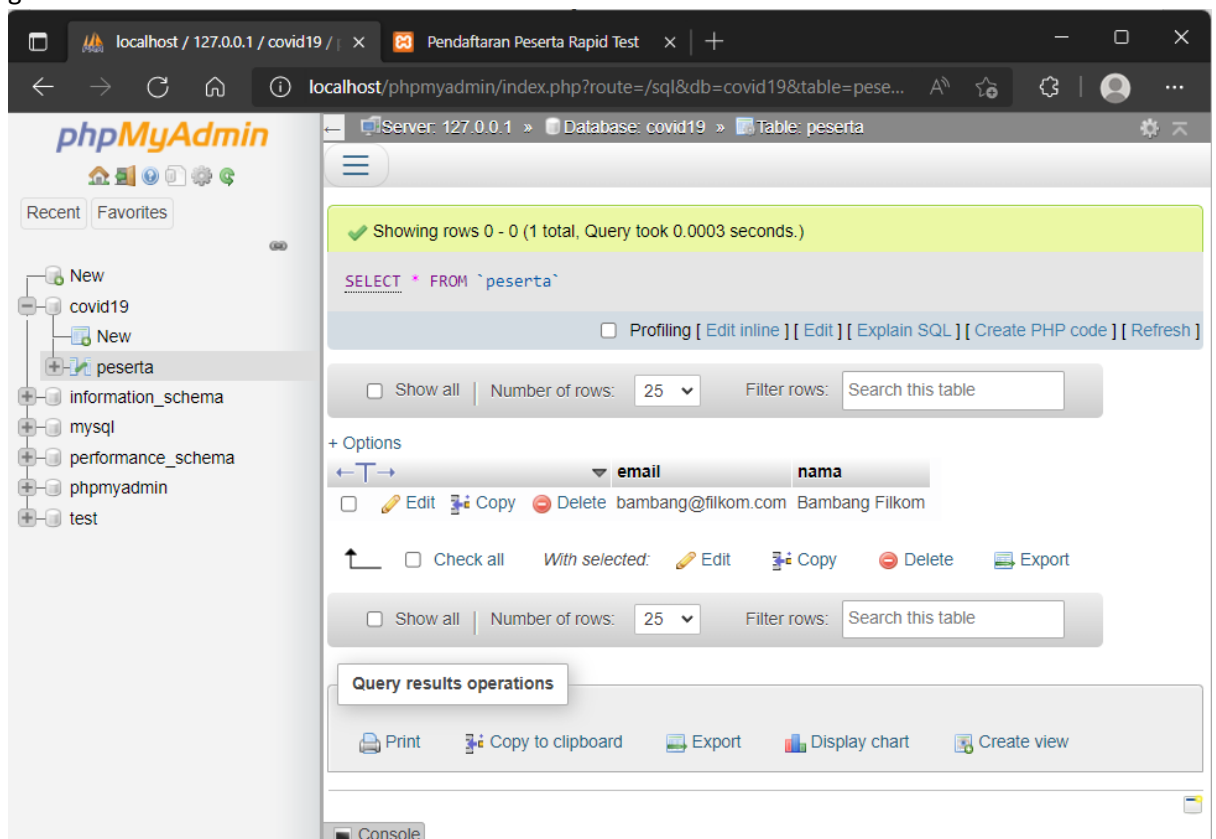
10. Pada potongan skrip PHP di atas (poin no. 3), dapat dilihat bahwa untuk mengakses *static members* menggunakan tanda :: sedangkan untuk mengakses *instance members* menggunakan tanda ->
11. Silahkan dicoba dengan mengakses halaman <http://localhost/rapidtest/tambah-peserta.html> kemudian cobalah untuk mengisi dengan data tertentu seperti pada gambar 3.



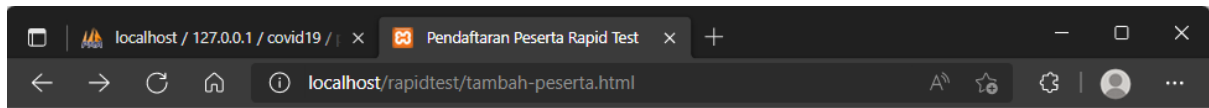
Formulir Pendaftaran Rapid Test

Email: bambang@filkom.com
Nama: Bambang Filkom
Daftar

12. Kirim formulir tersebut dengan menekan tombol Daftar. Saudara akan mendapati layar kosong, karena kita memang tidak mencetak apapun. Silahkan buka tabel peserta pada phpMyAdmin, jika benar maka saudara akan dapati data tersimpan dalam basis data seperti gambar 4.



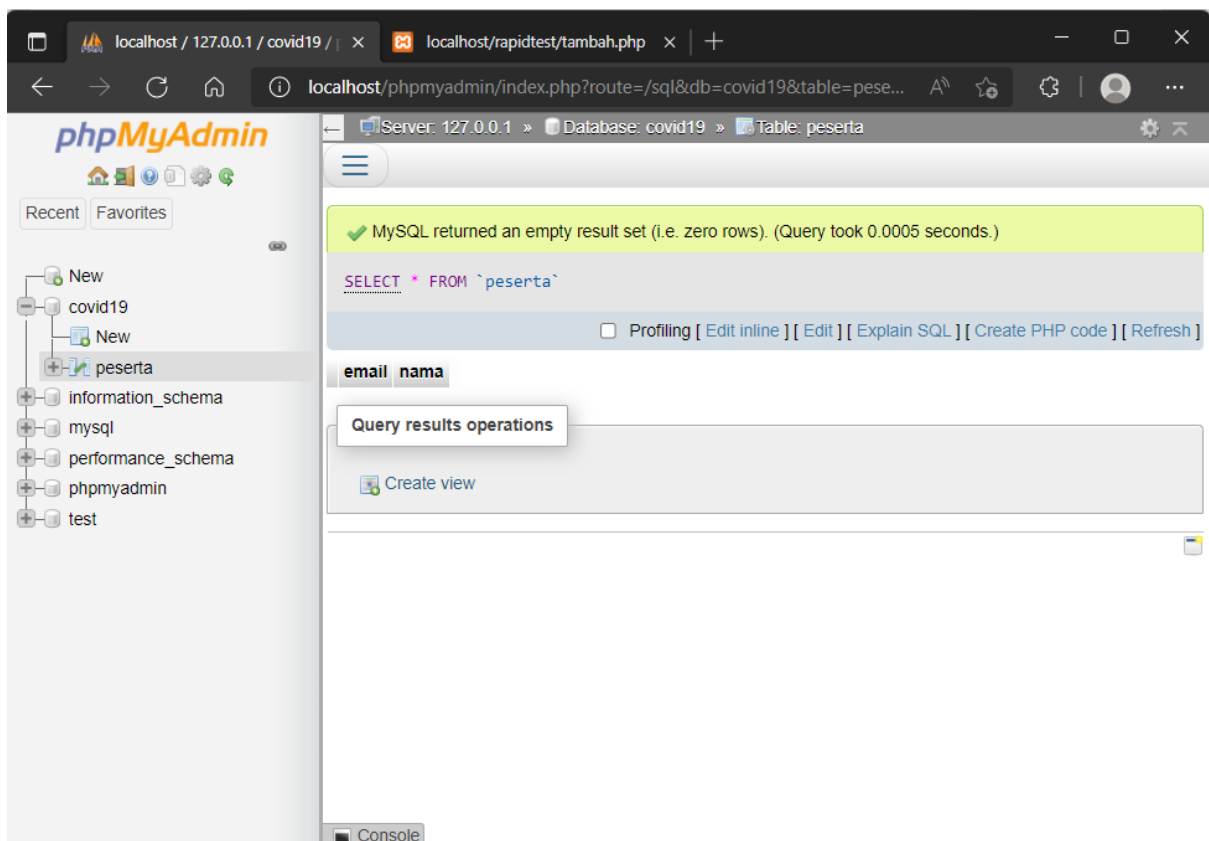
SQL Injection



Formulir Pendaftaran Rapid Test

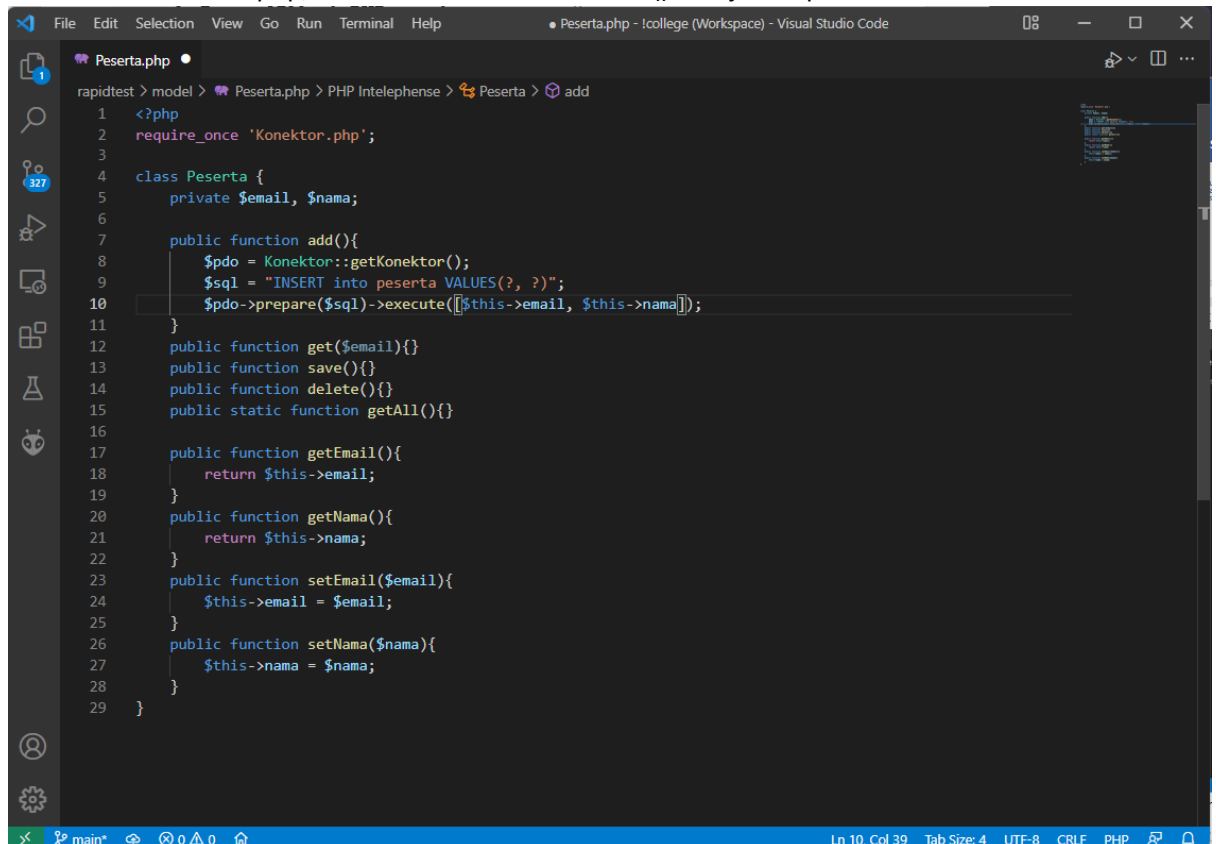
Email:

Nama:



Prepared statement untuk akses data yang lebih aman

1. Buka berkas Peserta.php, kemudian ubah *method* add() menjadi seperti kode di bawah ini:



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

2. Dari potongan skip PHP di atas (poin no. 1) terlihat ada dua tahap, yaitu prepare() dan execute(). Skrip SQL yang telah melalui fase prepare() tidak dapat diubah lagi walaupun dipaksa dengan *inject*. Setelah persiapan (prepare) selesai, maka berikutnya adalah melakukan *binding* terhadap data. Kita dapat menggunakan larik (*array*) untuk melakukan *binding* dengan urusan data sama dengan urutan tanda tanya (?) di bagian skrip SQL.
3. Sekarang coba lakukan pendaftaran lagi dengan menggunakan data yang sama dengannya untuk melakukan SQL *injection* tadi. Saudara akan mendapat bahwa SQL *injection* berhasil digagalkan dan data tersimpan seperti gambar 5.

