**NAMA : Gita Pita Loka**

**NPM : G1F022062**

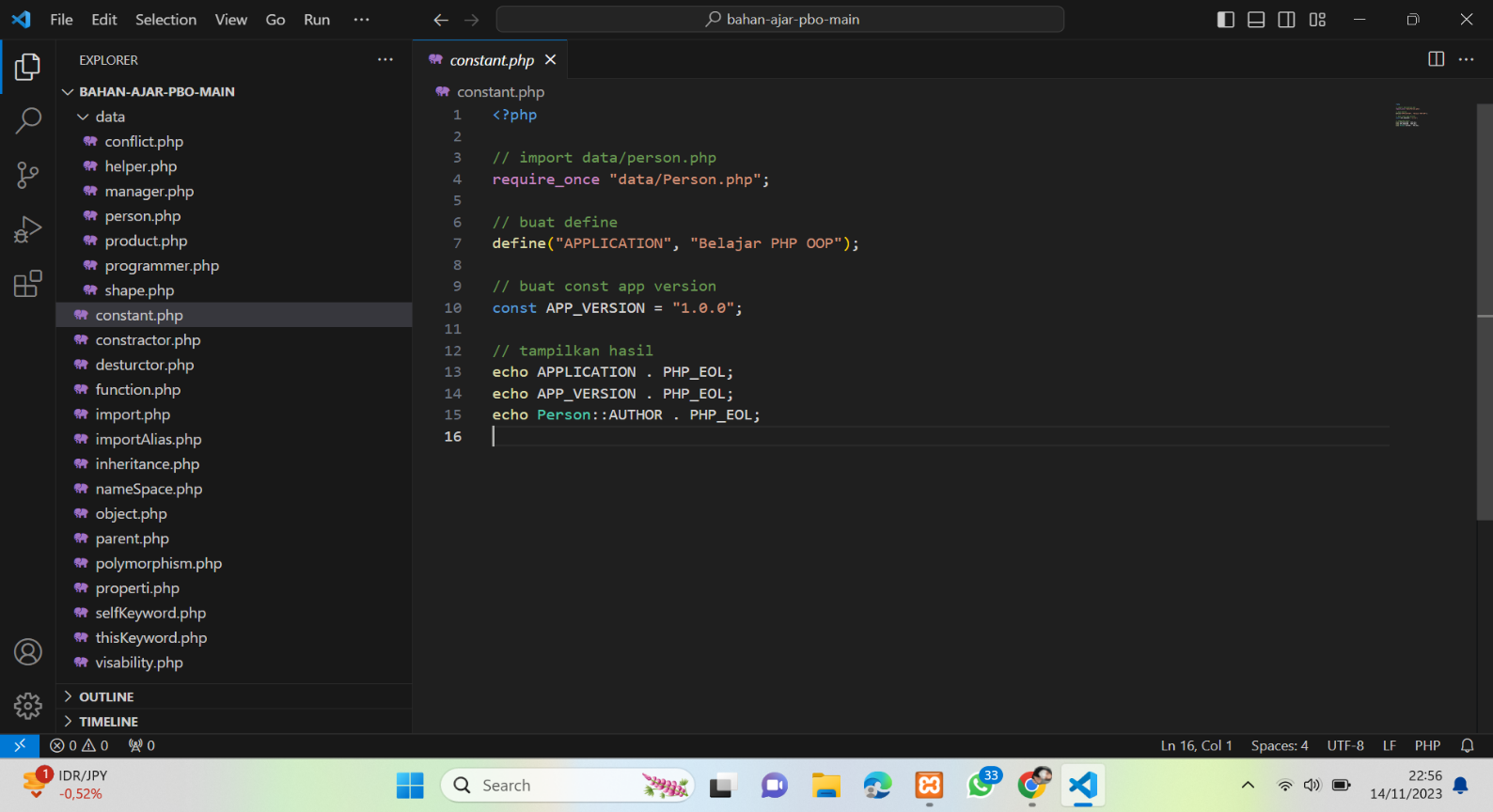
**Responsi PBO**

**Soal !**

* Silahkan lakukan git clone repositori dari <https://github.com/alzahfariski/bahan-ajar-pbo> (silahkan liat di youtube caranya) !
* lengkapi code php yang belum lengkap sehingga setiap file dapat di run dan tidak memunculkan error !
* upload atau lakukakan git push ke akun git kalian masing-masing !
* salin url lalu kumpulkan dengan berikan penjelasan mengenai pemahaman kalian secara descriptive (contoh penjelasan mengenai file object.php menggunakan code apa saja dan berfungsi untuk apa) penjelelasan kalian akan mempengaruhi penilaian!

**pembahasan !**

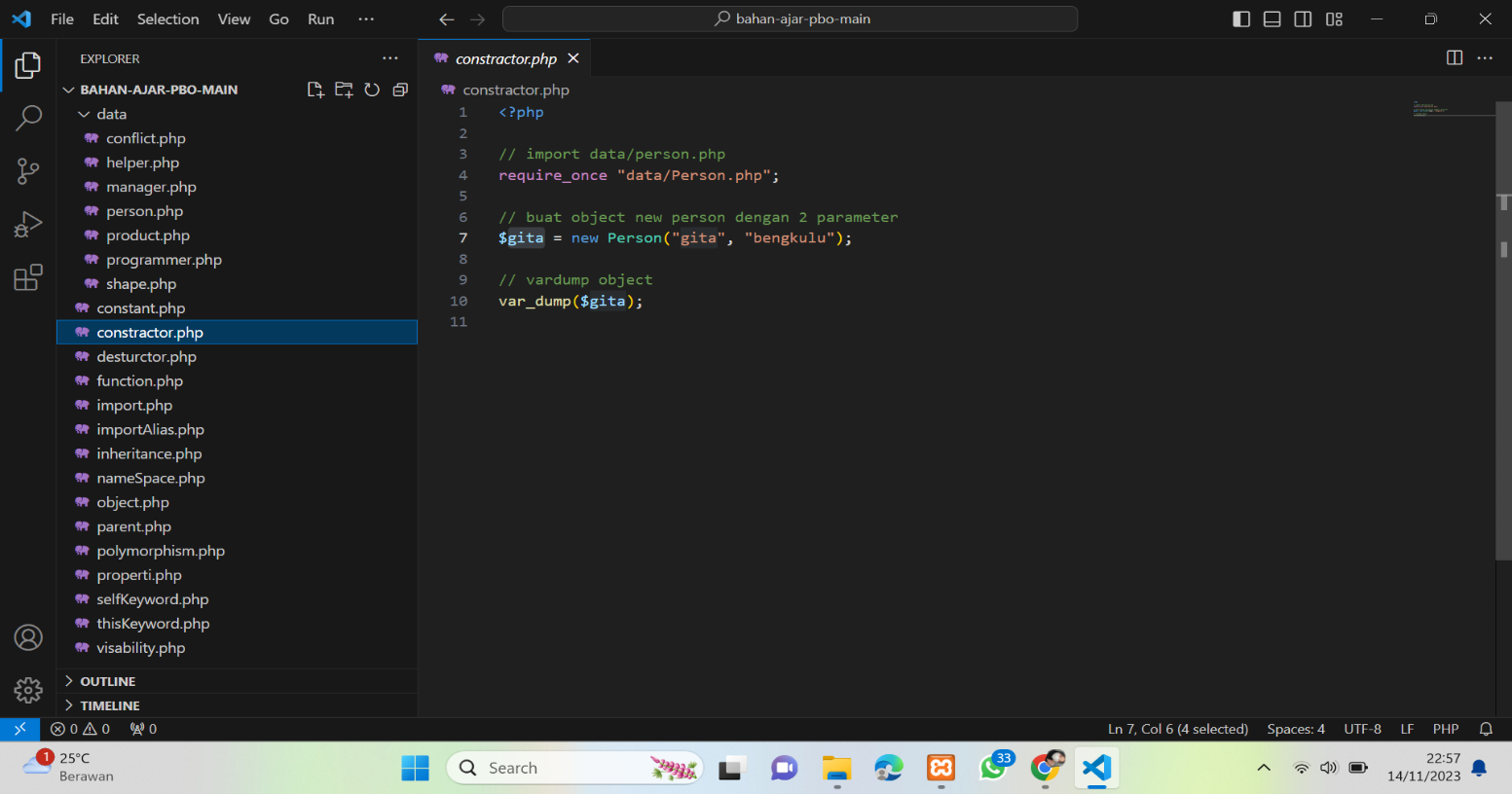
**1.** File constant.php



**Penjelasan :**

1. **require\_once "data/Person.php";**: Mengimpor (include) file "Person.php" yang terletak di direktori "data". Menggunakan require\_once untuk memastikan file hanya diimpor satu kali.
2. **define("APPLICATION", "Belajar PHP OOP");**: Mendefinisikan konstanta dengan nama "APPLICATION" dan nilai "Belajar PHP OOP". Konstanta dapat diakses di seluruh skrip dan nilainya tidak dapat diubah.
3. **const APP\_VERSION = "1.0.0";**: Mendefinisikan konstanta dengan nama "APP\_VERSION" dan nilai "1.0.0". Konstanta menggunakan kata kunci const dan juga dapat diakses di seluruh skrip.
4. **echo APPLICATION . PHP\_EOL;**: Mencetak nilai konstanta "APPLICATION" ke layar, diikuti dengan karakter baris baru (PHP\_EOL).
5. **echo APP\_VERSION . PHP\_EOL;**: Mencetak nilai konstanta "APP\_VERSION" ke layar, diikuti dengan karakter baris baru.
6. **echo Person::AUTHOR . PHP\_EOL**;: Mencetak nilai konstanta "AUTHOR" dari kelas "Person" ke layar. Menggunakan :: untuk mengakses konstanta atau properti statis dari kelas tanpa membuat instansi kelas.

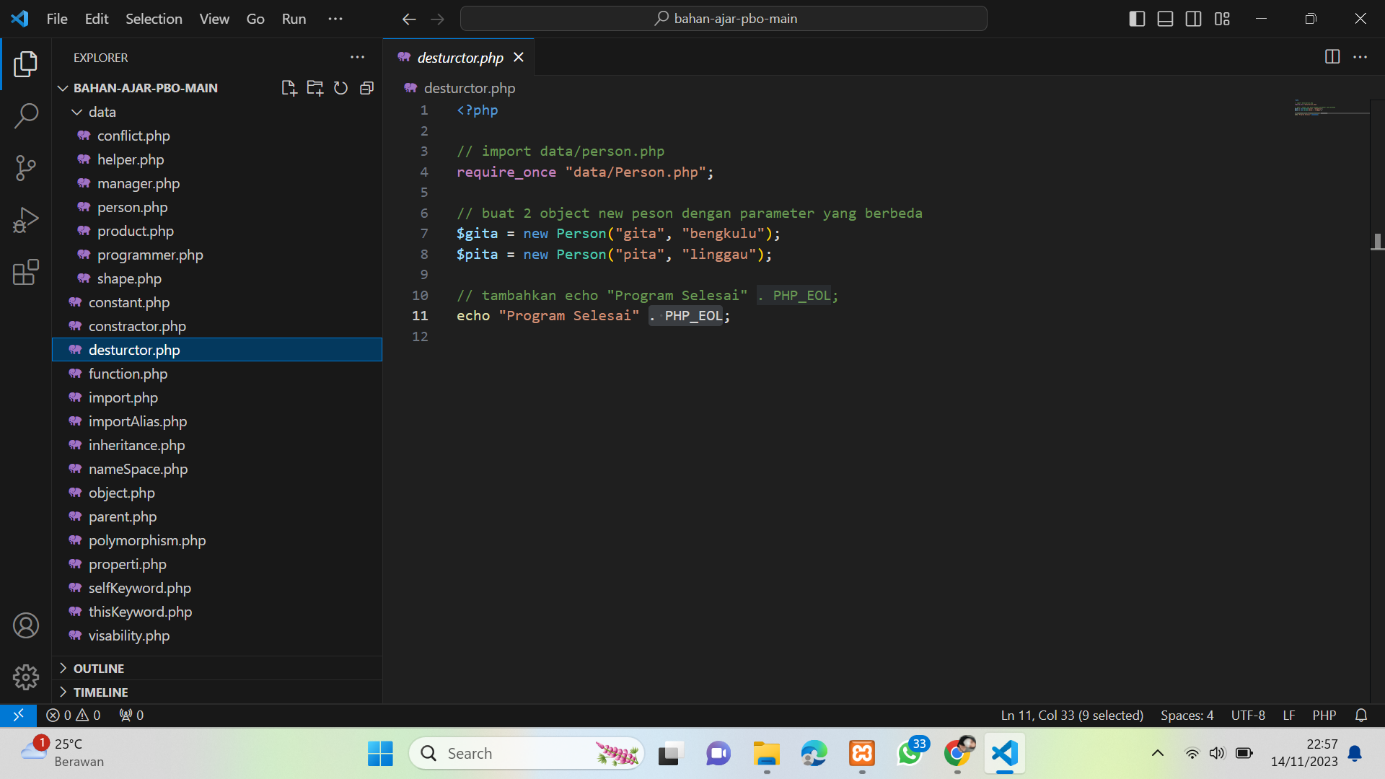
**2.** File constactor.php



**Penjelasan :**

1. **require\_once "data/Person.php";**: Menyertakan (import) file "Person.php" yang terletak di direktori "data".
2. **$gita = new Person("gita", "bengkulu");**: Membuat objek baru dari kelas "Person" dengan memberikan dua parameter ("gita" dan "bengkulu") ke konstruktor kelas.
3. **var\_dump($gita);**: Menggunakan fungsi var\_dump untuk menampilkan informasi terstruktur tentang objek yang baru dibuat, dalam hal ini, objek dengan nama "$gita".

**3.** File desturctor.php



**Penjelasan :**

1. **require\_once "data/Person.php"**;: Menyertakan (import) file "Person.php" yang berisi definisi kelas "Person" dari direktori "data".
2. **$gita = new Person("gita", "bengkulu");**: Membuat objek baru dari kelas "Person" dengan nama "$gita" dan memberikan dua parameter ("gita" dan "bengkulu") ke konstruktor kelas.
3. **$pita = new Person("pita", "linggau");**: Membuat objek baru lagi dari kelas "Person" dengan nama "$pita" dan parameter yang berbeda ("pita" dan "linggau").
4. **echo "Program Selesai" . PHP\_EOL;**: Mencetak teks "Program Selesai" ke layar, diikuti dengan karakter baris baru (PHP\_EOL untuk memastikan hasil cetakan terformat dengan baik).

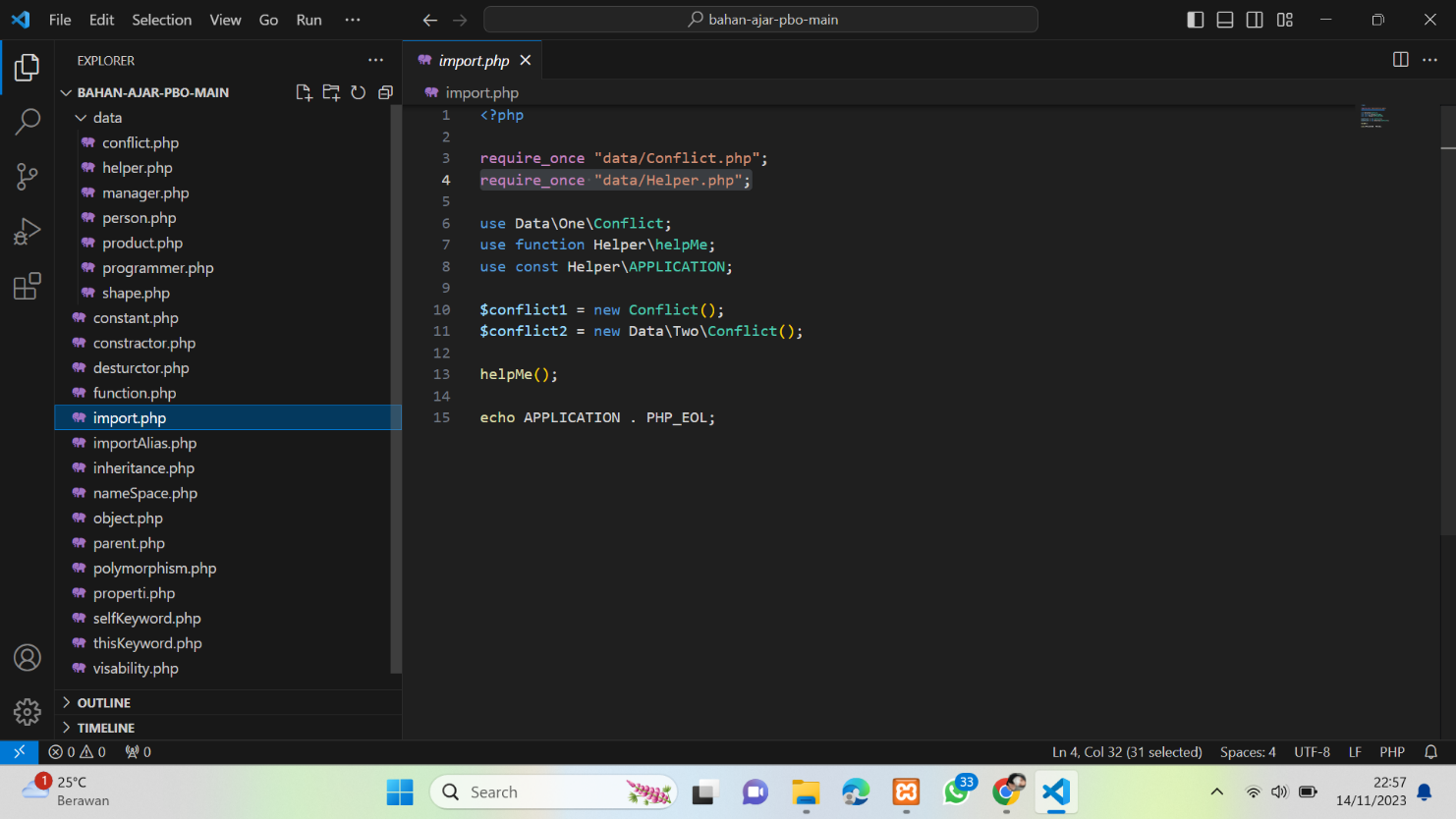
**4.** File function.php



**Penjelasan :**

1. **require\_once "data/person.php";**: Mengimpor (include) file "person.php" yang berisi definisi kelas "Person". Menggunakan **require\_once** untuk memastikan file hanya diimpor satu kali.
2. **$person1 = new Person("gita", "bengkulu");**: Membuat objek baru dari kelas "Person" dengan nama "$person1" dan memberikan dua argumen, yaitu "gita" sebagai nama dan "bengkulu" sebagai lokasi, kepada konstruktor kelas "Person".
3. **$person1->sayHello("gita");**: Memanggil metode "sayHello" dari objek "$person1" dengan memberikan argumen "gita".

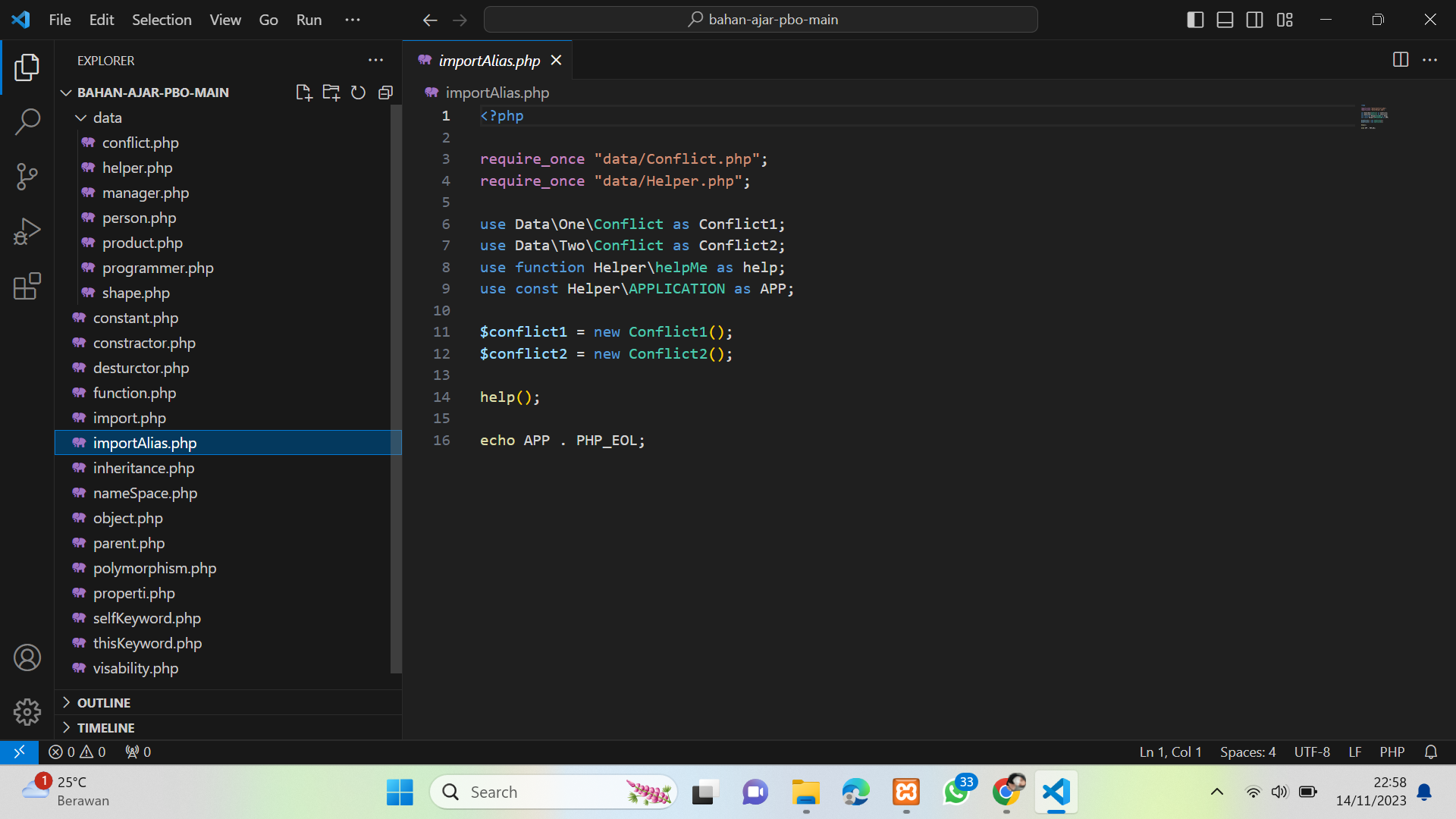
**5.** File import.php



**Penjelasan :**

1. **require\_once "data/Conflict.php";**: Mengimpor (include) file "Conflict.php" dari direktori "data".
2. **require\_once "data/Helper.php";**: Mengimpor (include) file "Helper.php" dari direktori "data".
3. **use Data\One\Conflict;**: Menggunakan namespace "Data\One" dan mengimpor kelas "Conflict" ke dalam skrip.
4. **use function Helper\helpMe;**: Menggunakan namespace "Helper" dan mengimpor fungsi "helpMe" ke dalam skrip.
5. **use const Helper\APPLICATION;**: Menggunakan namespace "Helper" dan mengimpor konstanta "APPLICATION" ke dalam skrip.
6. **$conflict1 = new Conflict();**: Membuat objek baru dari kelas "Conflict" yang sudah diimpor menggunakan **use**. Objek ini disimpan dalam variabel **$conflict1**.
7. **$conflict2 = new Data\Two\Conflict();**: Membuat objek baru dari kelas "Conflict" yang berada dalam namespace "Data\Two".
8. **helpMe();**: Memanggil fungsi "helpMe()" yang sudah diimpor menggunakan **use**. Fungsi ini mungkin melakukan tugas tertentu.
9. **echo APPLICATION . PHP\_EOL;**: Mencetak nilai konstanta "APPLICATION" ke layar, diikuti dengan karakter baris baru.

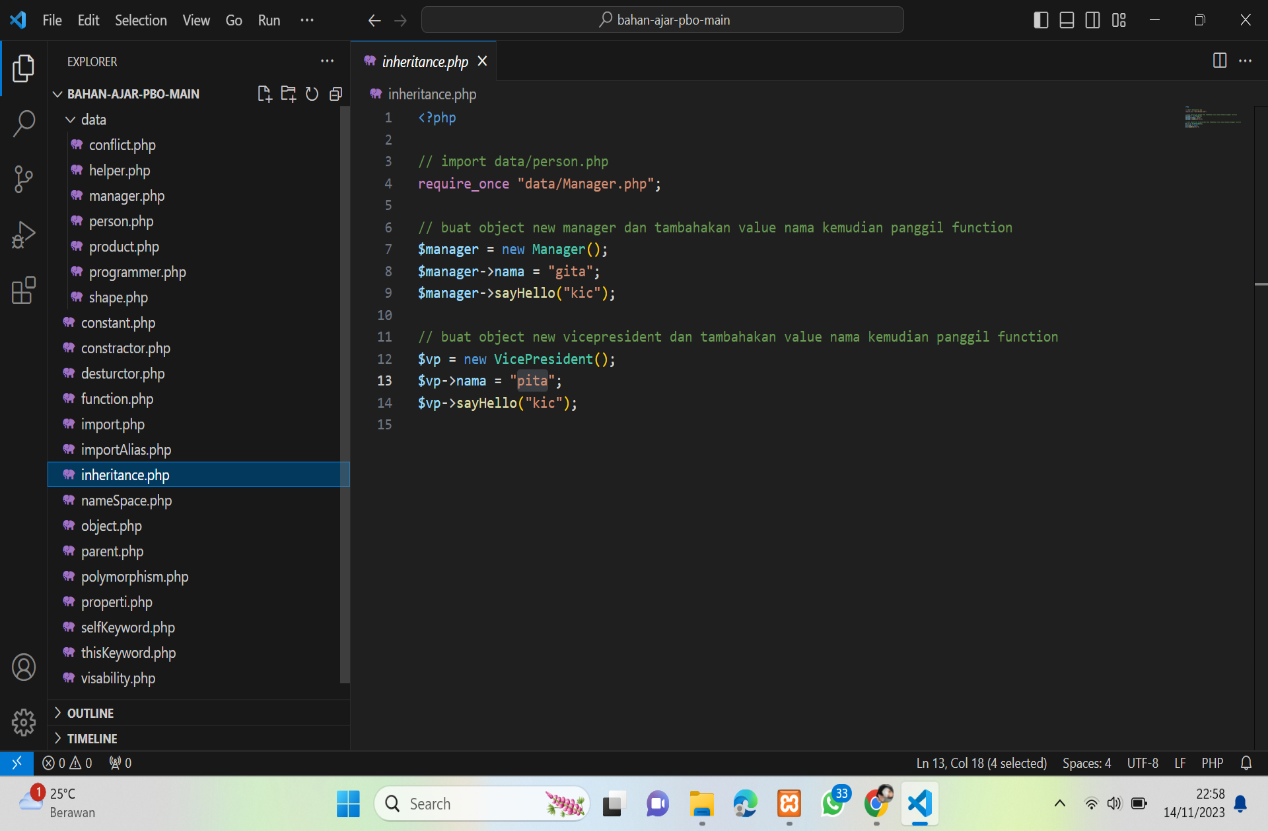
**6.** File importAlias.php



**Penjelasan :**

1. **require\_once "data/Conflict.php";**: Mengimpor (include) file "Conflict.php" dari direktori "data".
2. **require\_once "data/Helper.php";**: Mengimpor (include) file "Helper.php" dari direktori "data".
3. **use Data\One\Conflict as Conflict1;**: Menggunakan namespace "Data\One" dan mengimpor kelas "Conflict" ke dalam skrip dengan alias "Conflict1".
4. **use Data\Two\Conflict as Conflict2;**: Menggunakan namespace "Data\Two" dan mengimpor kelas "Conflict" ke dalam skrip dengan alias "Conflict2".
5. **use function Helper\helpMe as help;**: Menggunakan namespace "Helper" dan mengimpor fungsi "helpMe" ke dalam skrip dengan alias "help".
6. **use const Helper\APPLICATION as APP;**: Menggunakan namespace "Helper" dan mengimpor konstanta "APPLICATION" ke dalam skrip dengan alias "APP".
7. **$conflict1 = new Conflict1();**: Membuat objek baru dari kelas "Conflict" yang sudah diimpor menggunakan **use** dengan alias "Conflict1".
8. **$conflict2 = new Conflict2();**: Membuat objek baru dari kelas "Conflict" yang sudah diimpor menggunakan **use** dengan alias "Conflict2".
9. **help();**: Memanggil fungsi "help" yang sudah diimpor menggunakan **use**.
10. **echo APP . PHP\_EOL;**: Mencetak nilai konstanta "APPLICATION" yang sudah diimpor menggunakan **use** dengan alias "APP".

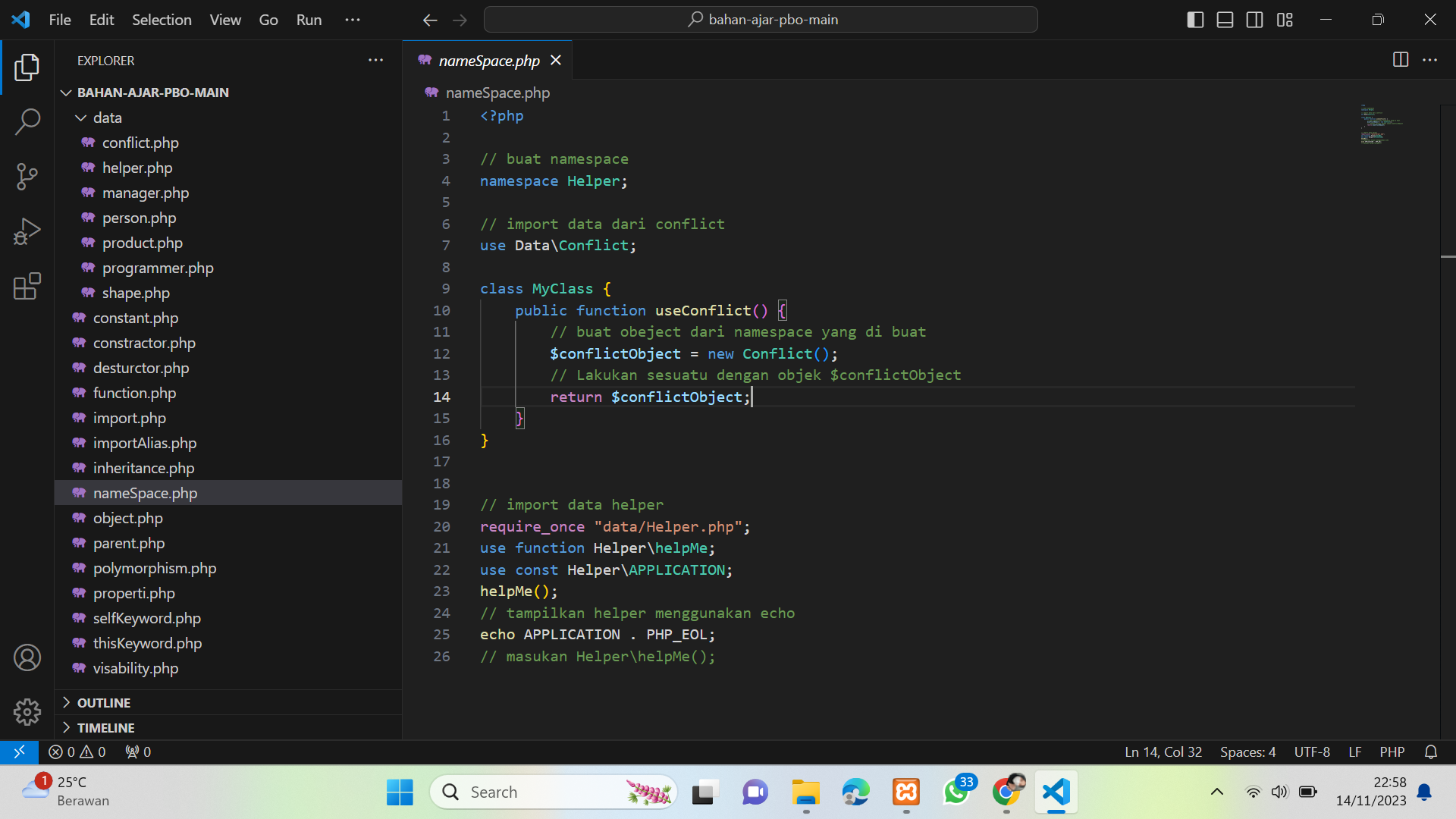
**7.** File inherintance.php



**Penjelasan :**

1. **require\_once "data/Manager.php";**: Mengimpor (include) file "Manager.php" yang berisi definisi kelas "Manager" dan "VicePresident".
2. **$manager = new Manager();**: Membuat objek baru dari kelas "Manager" dan menyimpannya dalam variabel **$manager**.
3. **$manager->nama = "gita";**: Menetapkan nilai properti "nama" dari objek "$manager" menjadi "gita".
4. **$manager->sayHello("kic");**: Memanggil metode "sayHello" dari objek "$manager" dengan memberikan argumen "kic".
5. **$vp = new VicePresident();**: Membuat objek baru dari kelas "VicePresident" dan menyimpannya dalam variabel "$vp".
6. **$vp->nama = "pita";**: Menetapkan nilai properti "nama" dari objek "$vp" menjadi "pita".
7. **$vp->sayHello("kic");**: Memanggil metode "sayHello" dari objek "$vp" dengan memberikan argumen "kic".

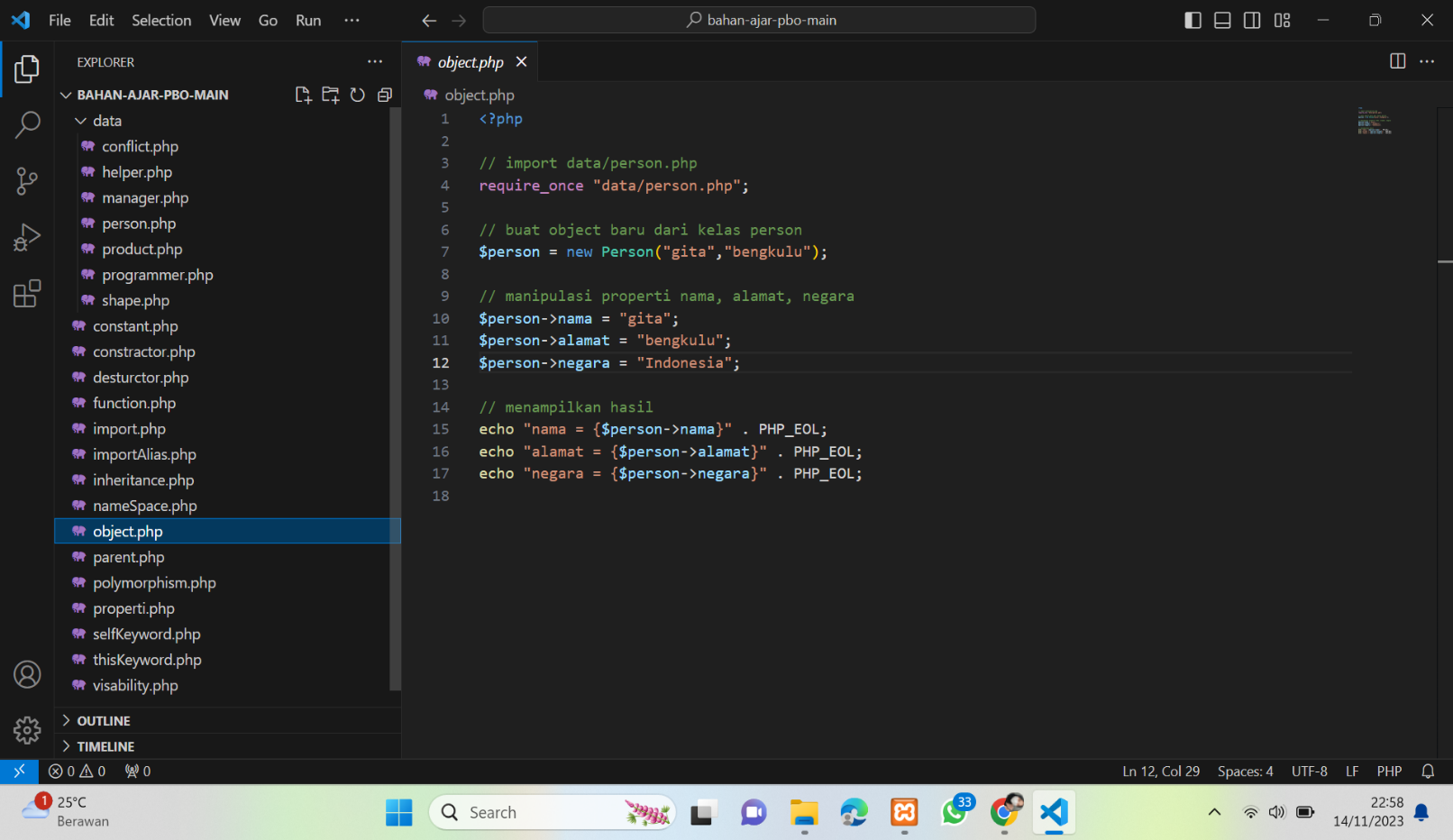
**8.** File nameSpace.php



**Penjelasan :**

1. **namespace Helper;**: Mendefinisikan namespace dengan nama "Helper".
2. **use Data\Conflict;**: Menggunakan namespace "Data" dan mengimpor kelas "Conflict" ke dalam namespace "Helper".
3. **class MyClass { ... }**: Mendefinisikan kelas "MyClass" di dalam namespace "Helper".
4. **require\_once "data/Helper.php";**: Mengimpor (include) file "Helper.php" yang mungkin berisi definisi fungsi atau konstanta.
5. **use function Helper\helpMe;**: Menggunakan namespace "Helper" dan mengimpor fungsi "helpMe" ke dalam skrip.
6. **use const Helper\APPLICATION;**: Menggunakan namespace "Helper" dan mengimpor konstanta "APPLICATION" ke dalam skrip.
7. **helpMe();**: Memanggil fungsi "helpMe()" yang sudah diimpor. Fungsi ini mungkin melakukan tugas tertentu.
8. **echo APPLICATION . PHP\_EOL;**: Mencetak nilai konstanta "APPLICATION" ke layar, diikuti dengan karakter baris baru (**PHP\_EOL**).
9. **Helper\helpMe();**: Pernyataan ini seharusnya tidak diperlukan karena fungsi "helpMe()" sudah diimpor sebelumnya dengan pernyataan **use function**.

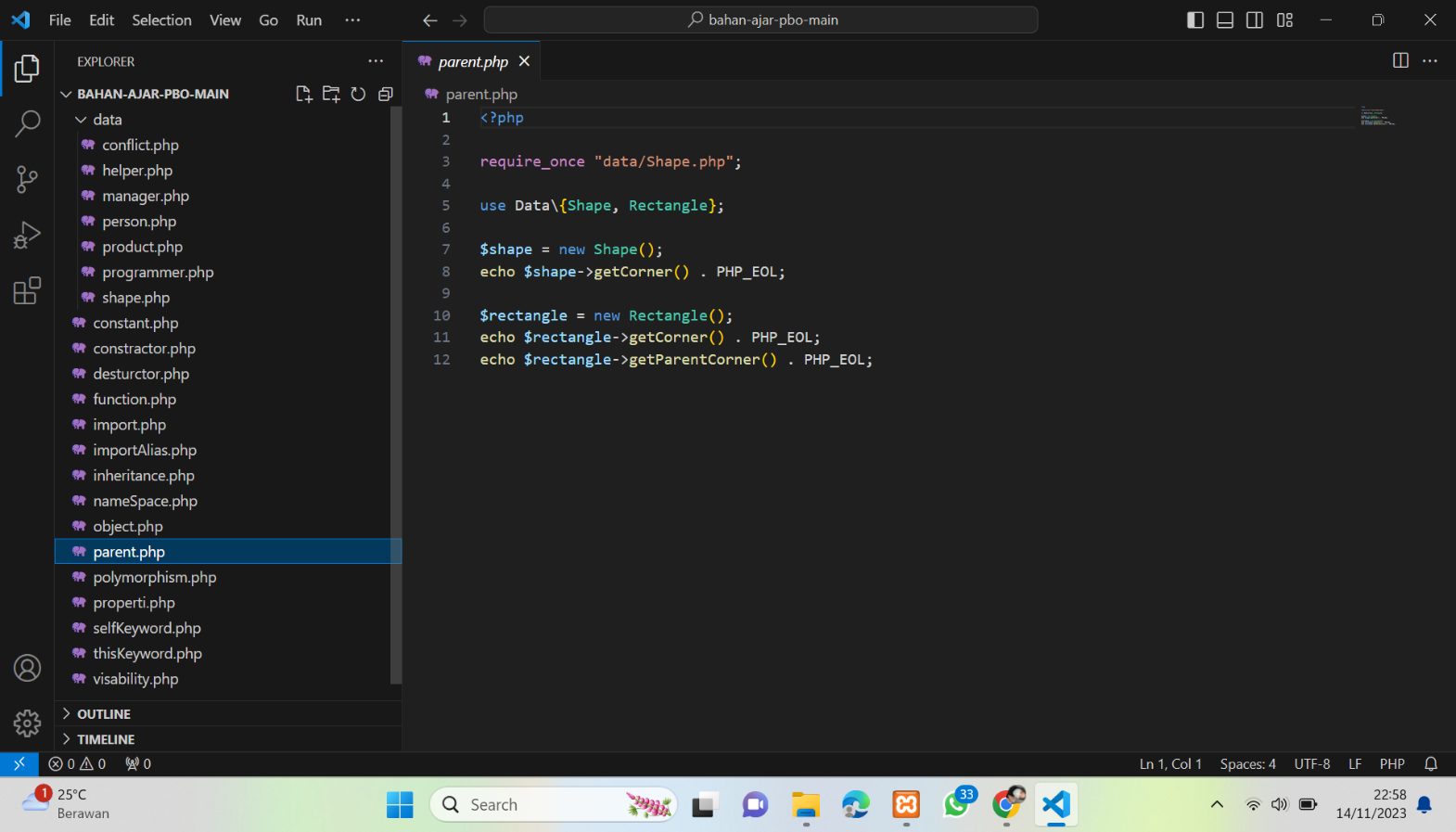
**9.** File object.php



**Penjelasan :**

1. **require\_once "data/person.php";**: Mengimpor (include) file "person.php" yang berisi definisi kelas "Person".
2. **$person = new Person("gita","bengkulu");**: Membuat objek baru dari kelas "Person" dengan nama "$person" dan memberikan dua argumen, yaitu "gita" sebagai nama dan "bengkulu" sebagai alamat, kepada konstruktor kelas "Person".
3. **$person->nama = "gita";**: Mengubah nilai properti "nama" dari objek "$person" menjadi "gita".
4. **$person->alamat = "bengkulu";**: Mengubah nilai properti "alamat" dari objek "$person" menjadi "bengkulu".
5. **$person->negara = "Indonesia";**: Mengubah nilai properti "negara" dari objek "$person" menjadi "Indonesia".
6. **echo "nama = {$person->nama}" . PHP\_EOL;**: Menampilkan nilai properti "nama" dari objek "$person" ke layar, diikuti dengan karakter baris baru.
7. **echo "alamat = {$person->alamat}" . PHP\_EOL;**: Menampilkan nilai properti "alamat" dari objek "$person" ke layar, diikuti dengan karakter baris baru.
8. **echo "negara = {$person->negara}" . PHP\_EOL;**: Menampilkan nilai properti "negara" dari objek "$person" ke layar, diikuti dengan karakter baris baru.

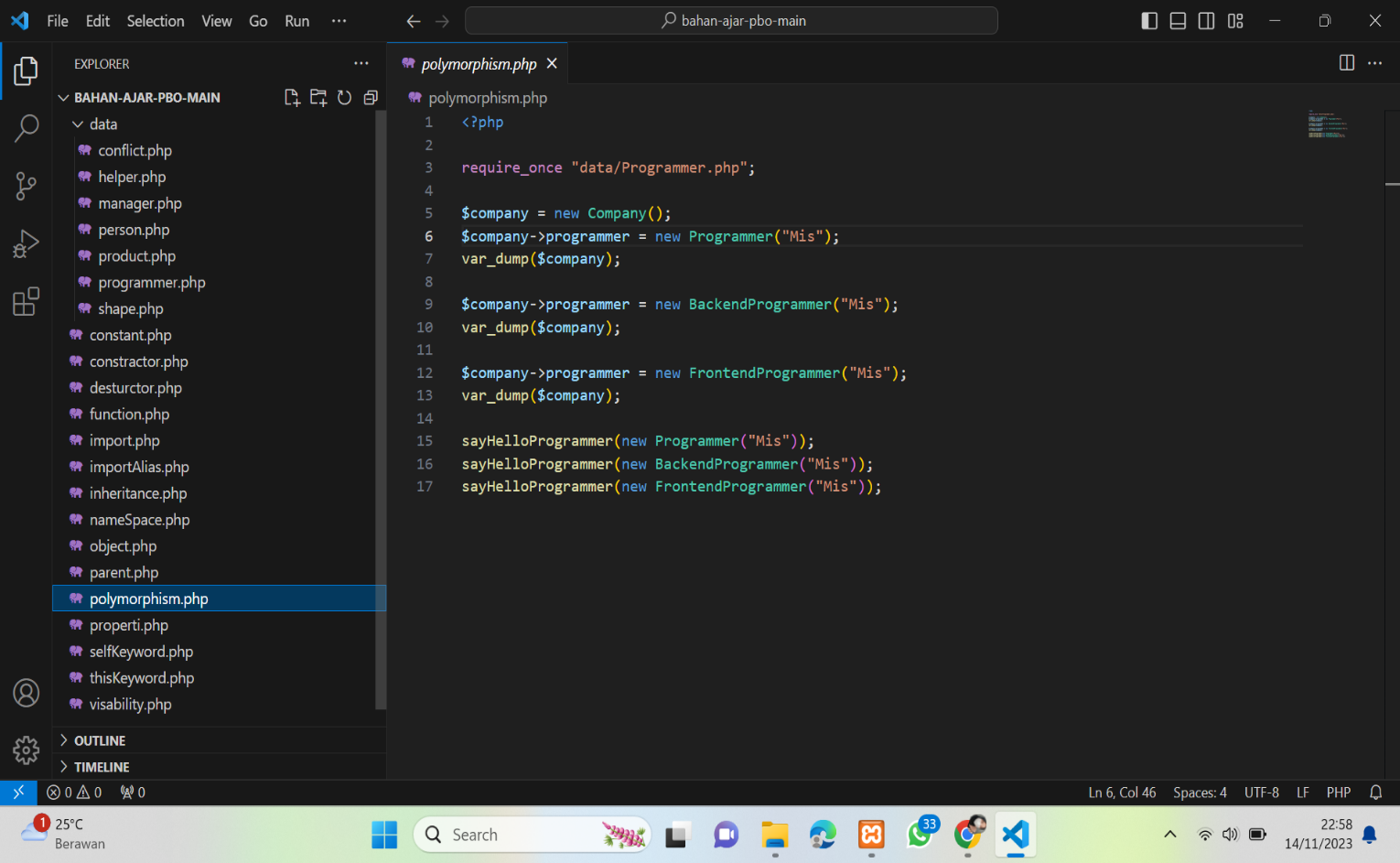
**10.** File parent.php



**Penjelasan :**

1. **require\_once "data/Shape.php";**: Mengimpor (include) file "Shape.php" yang berisi definisi kelas "Shape" dan "Rectangle".
2. **use Data\{Shape, Rectangle};**: Menggunakan pernyataan **use** untuk mengimpor namespace "Data" dan kelas "Shape" serta "Rectangle" ke dalam skrip.
3. **$shape = new Shape();**: Membuat objek baru dari kelas "Shape" dan menyimpannya dalam variabel "$shape".
4. **echo $shape->getCorner() . PHP\_EOL;**: Memanggil metode "getCorner()" dari objek "$shape" dan menampilkan hasilnya ke layar, diikuti dengan karakter baris baru (**PHP\_EOL**).
5. **$rectangle = new Rectangle();**: Membuat objek baru dari kelas "Rectangle" dan menyimpannya dalam variabel "$rectangle".
6. **echo $rectangle->getCorner() . PHP\_EOL;**: Memanggil metode "getCorner()" dari objek "$rectangle" (kelas "Rectangle") dan menampilkan hasilnya ke layar, diikuti dengan karakter baris baru.
7. **echo $rectangle->getParentCorner() . PHP\_EOL;**: Memanggil metode "getParentCorner()" dari objek "$rectangle" (kelas "Rectangle") .

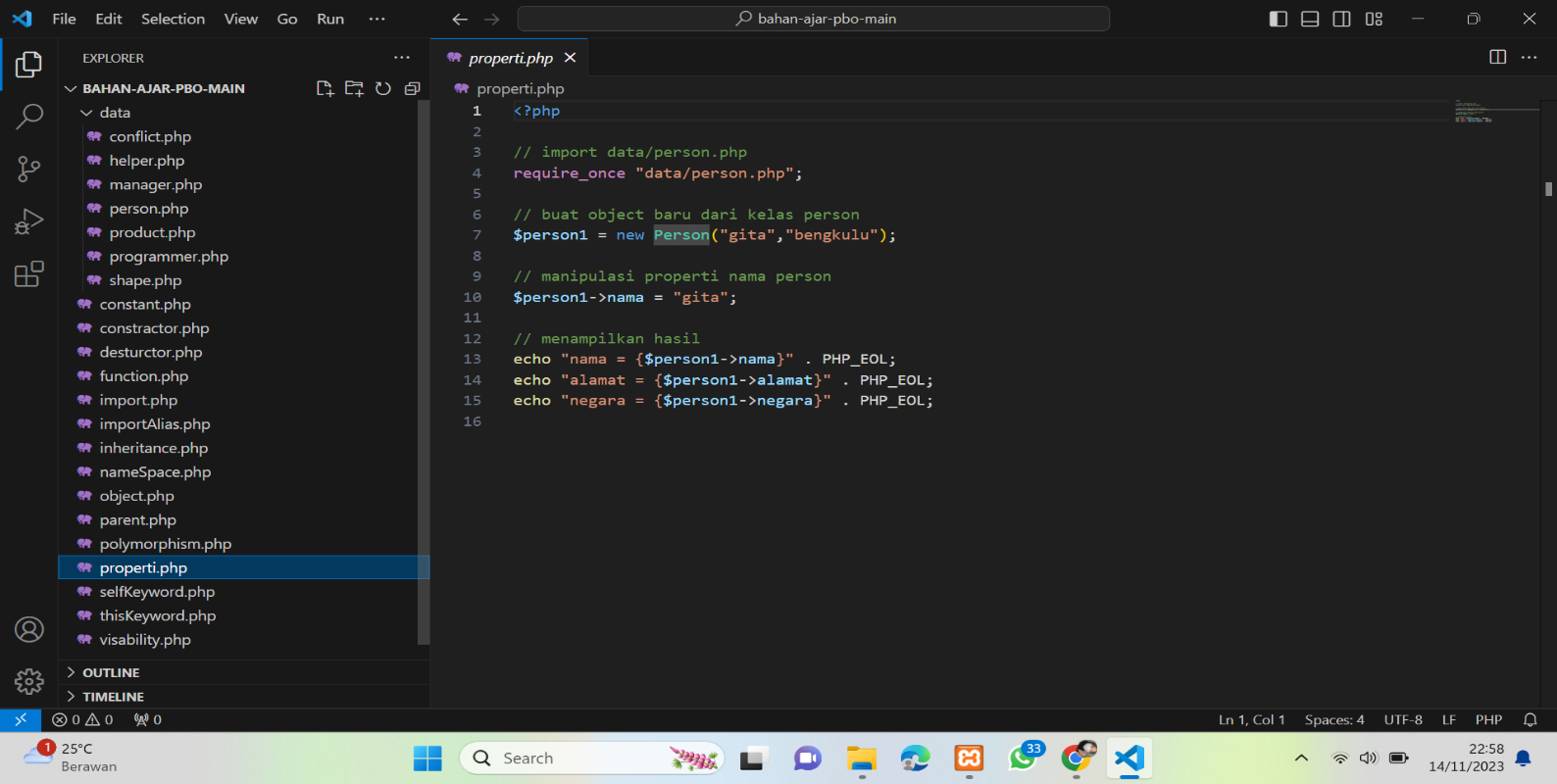
**11.** File polymorphism.php



**Penjelasan :**

1. **require\_once "data/Programmer.php";**: Mengimpor (include) file "Programmer.php" yang berisi definisi kelas "Programmer", "BackendProgrammer", "FrontendProgrammer", dan fungsi "sayHelloProgrammer".
2. **$company = new Company();**: Membuat objek baru dari kelas "Company" dan menyimpannya dalam variabel "$company".
3. **$company->programmer = new Programmer("Mis");**: Mengisi properti "programmer" dari objek "$company" dengan objek baru dari kelas "Programmer" yang memiliki nama "Mis".
4. **$company->programmer = new BackendProgrammer("Mis");**: Mengisi properti "programmer" dari objek "$company" dengan objek baru dari kelas "BackendProgrammer" yang memiliki nama "Mis".
5. **$company->programmer = new FrontendProgrammer("Mis");**: Mengisi properti "programmer" dari objek "$company" dengan objek baru dari kelas "FrontendProgrammer" yang memiliki nama "Mis".
6. **sayHelloProgrammer(new Programmer("Mis"));**: Memanggil fungsi "sayHelloProgrammer" dengan memberikan objek baru dari kelas "Programmer" yang memiliki nama "Mis".
7. **sayHelloProgrammer(new BackendProgrammer("Mis"));**: Memanggil fungsi "sayHelloProgrammer" dengan memberikan objek baru dari kelas "BackendProgrammer" yang memiliki nama "Mis".

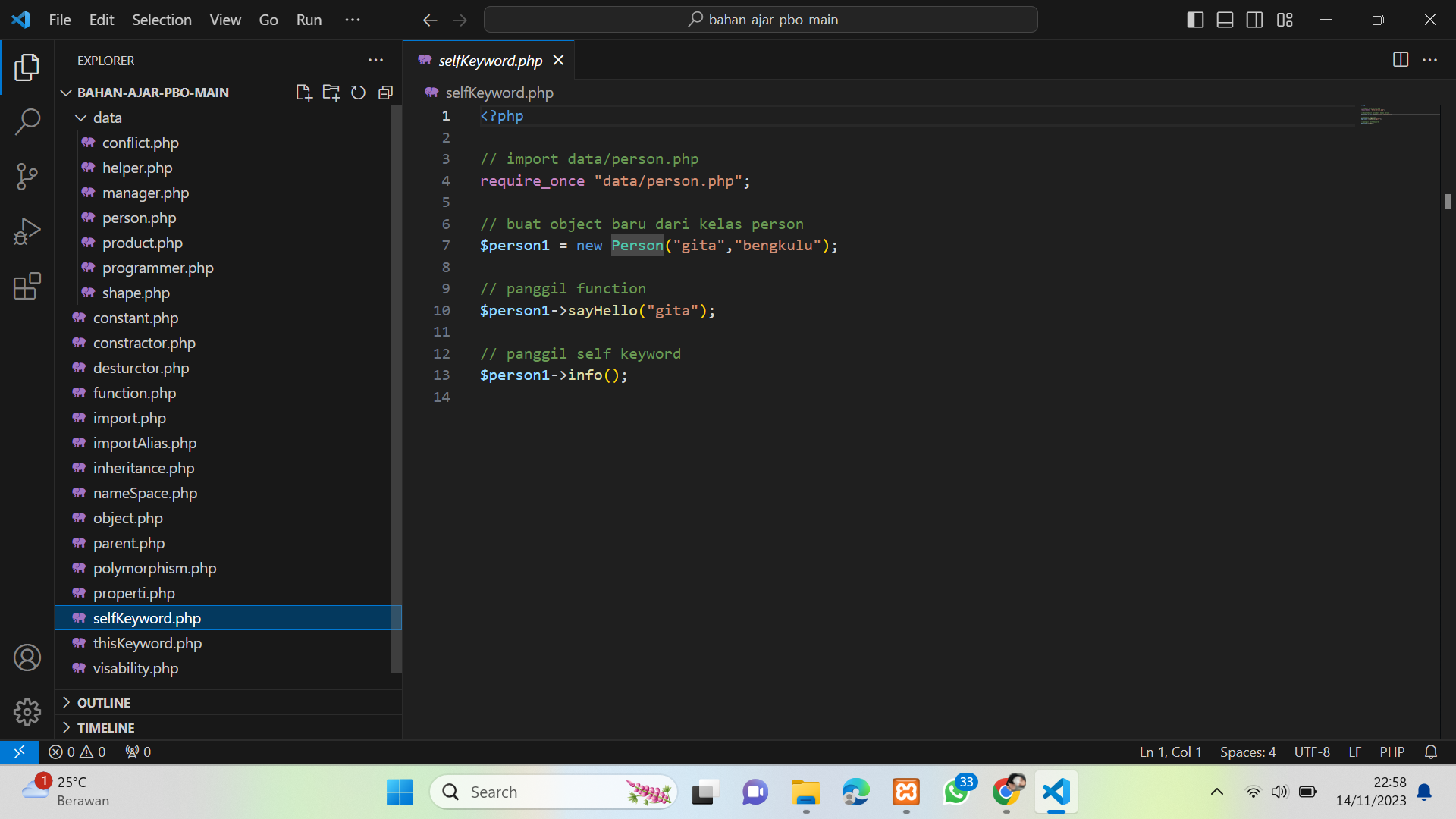
**12.** File properti.php



**Penjelasan :**

1. **require\_once "data/person.php";**: Mengimpor (include) file "person.php" yang berisi definisi kelas "Person".
2. **$person1 = new Person("gita","bengkulu");**: Membuat objek baru dari kelas "Person" dengan nama "$person1" dan memberikan dua argumen, yaitu "gita" sebagai nama dan "bengkulu" sebagai alamat, kepada konstruktor kelas "Person".
3. **$person1->nama = "gita";**: Memanipulasi (mengganti) nilai properti "nama" dari objek "$person1" menjadi "gita".
4. **echo "nama = {$person1->nama}" . PHP\_EOL;**: Menampilkan nilai properti "nama" dari objek "$person1" ke layar, diikuti dengan karakter baris baru (**PHP\_EOL**).
5. **echo "alamat = {$person1->alamat}" . PHP\_EOL;**: Menampilkan nilai properti "alamat" dari objek "$person1" ke layar, diikuti dengan karakter baris baru.
6. **echo "negara = {$person1->negara}" . PHP\_EOL;**: Menampilkan nilai properti "negara" dari objek "$person1" ke layar, diikuti dengan karakter baris baru.

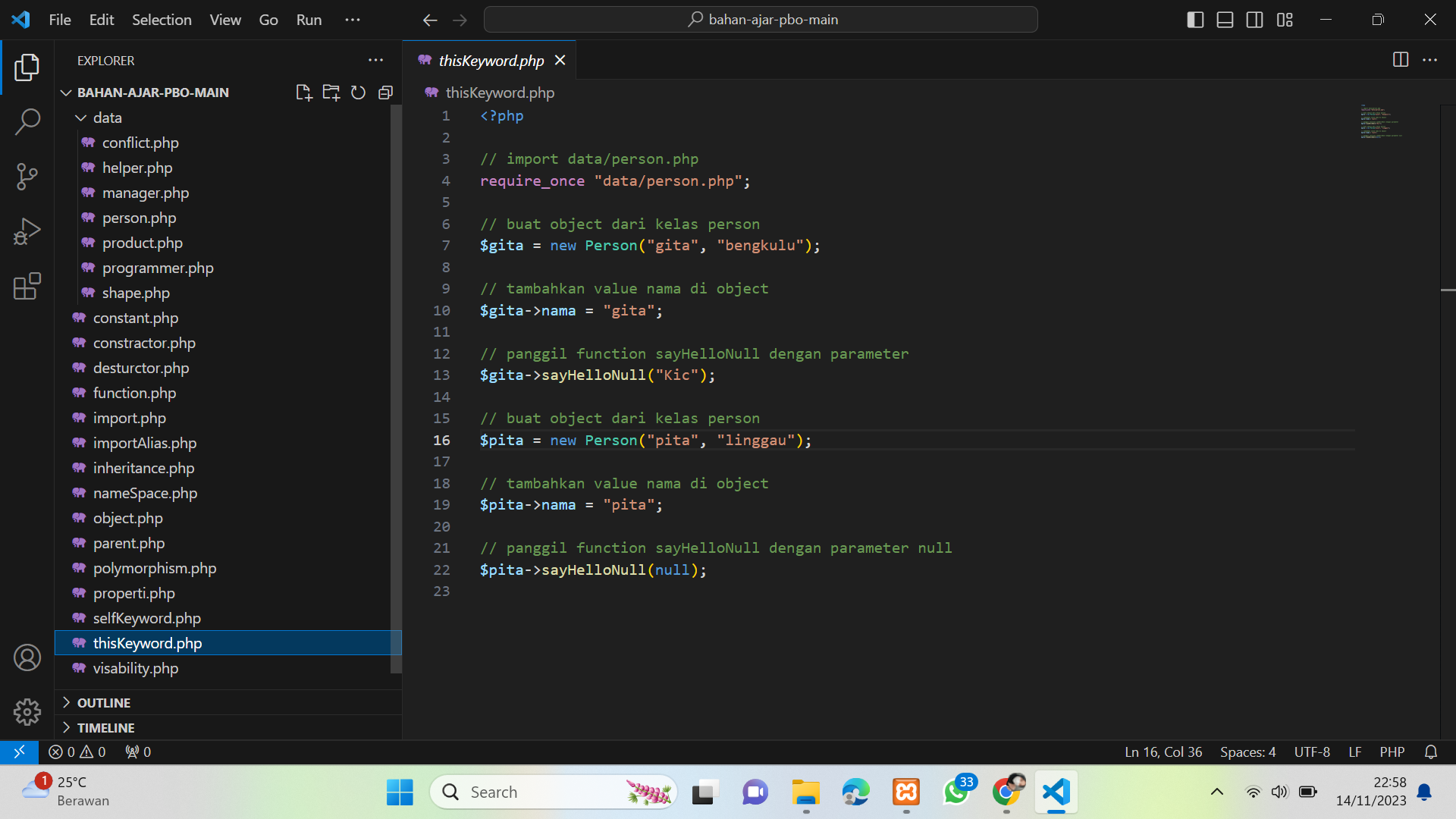
**13.** File selfKeyword.php



**Penjelasan :**

1. **require\_once "data/person.php";**: Mengimpor (include) file "person.php" yang berisi definisi kelas "Person".
2. **$person1 = new Person("gita","bengkulu");**: Membuat objek baru dari kelas "Person" dengan nama "$person1" dan memberikan dua argumen, yaitu "gita" sebagai nama dan "bengkulu" sebagai alamat, kepada konstruktor kelas "Person".
3. **$person1->sayHello("gita");**: Memanggil metode "sayHello" dari objek "$person1" dengan memberikan argumen "gita".
4. **$person1->info();**: Memanggil metode "info" dari objek "$person1". Metode ini mungkin menampilkan informasi tambahan yang terkait dengan objek "Person", dan jika metode "info" menggunakan **self.**

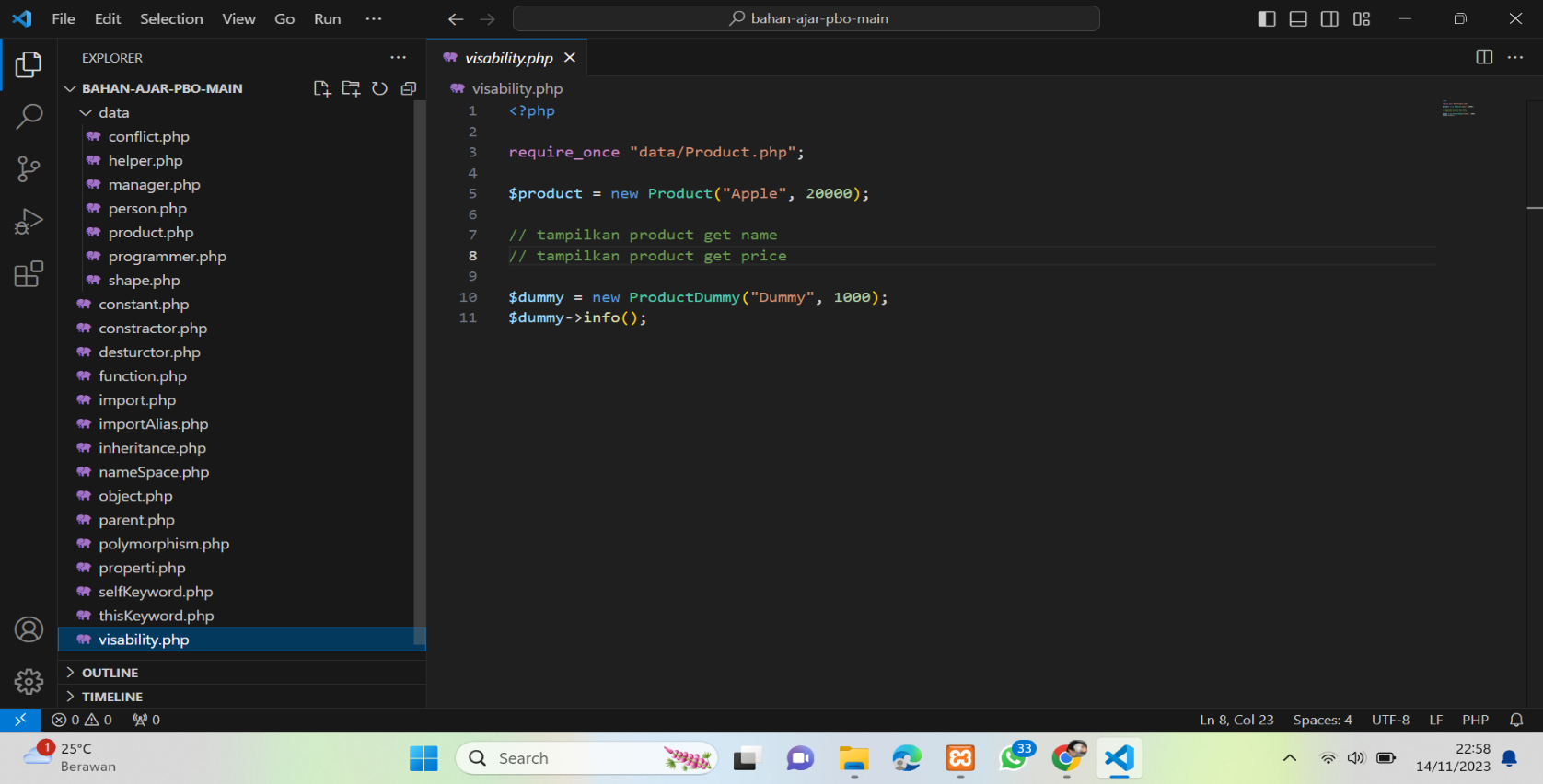
**14.** File thisKeyword.php



**Penjelasan :**

1. **require\_once "data/person.php";**: Mengimpor (include) file "person.php" yang berisi definisi kelas "Person".
2. **$gita = new Person("gita", "bengkulu");**: Membuat objek baru dari kelas "Person" dengan nama "$gita" dan memberikan dua argumen, yaitu "gita" sebagai nama dan "bengkulu" sebagai alamat, kepada konstruktor kelas "Person".
3. **$gita->nama = "gita";**: Menetapkan nilai properti "nama" dari objek "$gita" menjadi "gita".
4. **$gita->sayHelloNull("Kic");**: Memanggil metode "sayHelloNull" dari objek "$gita" dengan memberikan argumen "Kic".
5. **$pita = new Person("pita", "linggau");**: Membuat objek baru dari kelas "Person" dengan nama "$pita" dan memberikan dua argumen, yaitu "pita" sebagai nama dan "linggau" sebagai alamat, kepada konstruktor kelas "Person".
6. **$pita->nama = "pita";**: Menetapkan nilai properti "nama" dari objek "$pita" menjadi "pita".
7. **$pita->sayHelloNull(null);**: Memanggil metode "sayHelloNull" dari objek "$pita" dengan memberikan argumen null.

**15.** File vibability.php



**Penjelasan :**

1. **require\_once "data/Product.php";**: Mengimpor (include) file "Product.php" yang berisi definisi kelas "Product" dan "ProductDummy".
2. **$product = new Product("Apple", 20000);**: Membuat objek baru dari kelas "Product" dengan nama "$product" dan memberikan dua argumen, yaitu "Apple" sebagai nama produk dan 20000 sebagai harga produk, kepada konstruktor kelas "Product".
3. **// tampilkan product get name**: Komentar yang menunjukkan niat untuk menampilkan nilai dari metode "getName()" pada objek "$product". Namun, pernyataan ini sebenarnya tidak ada dalam kode.
4. **// tampilkan product get price**: Komentar yang menunjukkan niat untuk menampilkan nilai dari metode "getPrice()" pada objek "$product". Namun, pernyataan ini sebenarnya tidak ada dalam kode.
5. **$dummy = new ProductDummy("Dummy", 1000);**: Membuat objek baru dari kelas "ProductDummy" dengan nama "$dummy" dan memberikan dua argumen, yaitu "Dummy" sebagai nama produk dan 1000 sebagai harga produk, kepada konstruktor kelas "ProductDummy".
6. **$dummy->info();**: Memanggil metode "info()" dari objek "$dummy". Metode ini mungkin menampilkan informasi tambahan tentang objek "ProductDummy", seperti nama dan harga produk, ke layar.

**Link github : https://github.com/gitapitaloka/Gita-Pita-Loka\_responsipbo.git**

-