**BAB VI**

**CLASS DAN CONSTRUCTOR**

# **Tujuan**

1. Praktikan mampu membedakan konsep dasar *Class* dan *Constructor* pada bahasa pemrograman Python, PHP, Java, dan C#.
2. Praktikan mampu mengimplementasikan konsep pemrograman berbasis *object* dalam bahasa pemrograman Python, PHP, Java, dan C#.
3. Praktikan dapat memahami dasar penggunaan *Class* dan *Constructor* pada bahasa pemrograman Python, PHP, Java, dan C#.
4. Praktikan mampu mengimplementasikan *Class* dan *Constructor* menggunakan bahasa pemrograman Python, PHP, Java, dan C#.
5. Mampu membedakan *Class* dan *Constructor* dari empat bahasa pemrograman yang berbeda yaitu Python, PHP, Java, dan C#.

# **Dasar Teori**

# **Pemrograman Berbasis Objek**

Pemrograman berorientasi *object* (*Object Oriented Programming* atau disingkat OOP) adalah paradigma pemrograman yang berorientasikan kepada *object* yang merupakan suatu metode dalam pembuatan program, dengan tujuan untuk menyelesaikan kompleksnya berbagai masalah program yang terus meningkat. *Object* adalah entitas yang memiliki atribut, dan perilaku (*behaviour*).

Ide dasar pada OOP atau *Object Oriented Programming* adalah menyelesaikan suatu masalah bukan menggunakan perintah program tetapi dengan membuat *object-object* yang saling berinteraksi. OOP atau *Object Oriented Programming* memiliki konsep dasar yang lain yaitu *Class, Constructor, Object, Inheritance, Encapsulaton, Polymorphism* dan *Abstraction.*

Keuntungan OOP atau *Object Oriented Programming* diantaranya :

1. OOP atau *Object Oriented Programming* menyediakan struktur modular yang jelas untuk program sehingga OOP atau *Object Oriented Programming* sangat bagus digunakan untuk mendefinisikan tipe data abstrak di mana *detail* implementasinya tersembunyi.
2. OOP atau *Object Oriented Programming* akan mempermudah dalam memaintain dan memodifikasi kode yang sudah ada. *Object* yang baru dapat dibuat tanpa mengubah kode yang sudah ada.
3. OOP atau *Object Oriented Programming* menyediakan *framework* untuk *library* kode di mana komponen *software* yang tersedia dapat dengan mudah diadaptasi dan dimodifikasi oleh *programmer*. Hal ini sangat berguna untuk mengembangkan GUI (*Graphical User Interfaces*).
4. *Improved Software Development Productivity*. Adalah sistem program dapat dimodifikasi tanpa melibatkan banyak modul dimana hanya objek saja yang terlibat. Selain itu, sistem program dapat dikembangkan sampai skala paling kompleks.
5. *Improved Software Maintainability*. Adalah bagian dari *software* dapat dengan mudah di *maintenance* jika ada perubahan meskipun dalam skala yang cukup besar.
6. *Faster Development*. Adalah metode ini didukung oleh banyak *library* objek, sehingga mempercepat penyelesaian program dan juga projek berikutnya.
7. *Lower Cost of Development*. Adalah *faster development* tentu akan mengurangi biaya pengembangan pembuatan program.
8. *Higher Quality Software*. Adalah *faster development* pun akan memberikan lebih banyak waktu dan sumberdaya untuk proses verifikasi *software.*

Kerugian OOP atau *Object Oriented Programming* diantaranya :

1. Tidak memperbolehkan implementasi yang kuat pada *reuse.*
2. Properti *software* tidak terikat dalam satu unit fungsional, sehingga harus *crosscut* di antara komponennya.
3. *Crosscut* tersebut mengakibatkan sulitnya pengembangan dan pemeliharaan.
4. Untuk beberapa *programmer* butuh waktu untuk terbiasa dengan OOP atau *Object Oriented Programming*.
5. Ukuran program yang dibuat dengan metode ini cukup besar.
6. *Runtime* program lebih lambat.
7. Tidak semua masalah program bisa diselesaikan dengan OOP atau *Object Oriented Programming.*

(Sumber : *Riadi,Muchlisin.http://www.kajianpustaka.com/2016/10/pemrograman-berorientasi-object-oop.html dan* [*https://defablogblog.wordpress.com/2017/01/26/kelebihan-dan-kekurangan-oop-dan-pdo/*](https://defablogblog.wordpress.com/2017/01/26/kelebihan-dan-kekurangan-oop-dan-pdo/))

# **Class**

*Class* adalah *prototype* atau *blueprint* atau rancangan yang mendefinisikan *variable* dan *method*-*method* pada seluruh *object* tertentu. *Class* berfungsi untuk menampung isi dari program yang akan di jalankan, di dalamnya berisi atribut atau tipedata dan *method* untuk menjalankan suatu program.

*Class* merupakan suatu *blueprint* atau cetakan untuk menciptakan suatu *instant* dari *object*. *Class* juga merupakan grup suatu *object* dengan kemiripan *attributes* atau *properties*, *behaviour,* dan relasi ke-*object* lain. Contohnya seperti *Class Person, Vehicle, Tree, Fruit,* dan lain-lain. *Class* menentukan apa yang dimiliki sebuah *object* (atribut) dan apa yang dapat dilakukan *object* (metode). Umumnya *class* mewakili sebuah kata benda yang dapat merupakan seseorang, hewan, buah, ataupun benda mati.

(Sumber : [*http://scdc.binus.ac.id/himsisfo/2016/07/pengertian-methode-class-dan-object-dalam-oop/*](http://scdc.binus.ac.id/himsisfo/2016/07/pengertian-methode-class-dan-object-dalam-oop/) *dan* [*https://id.wikipedia.org/wiki/Kelas\_(ilmu\_komputer)*](https://id.wikipedia.org/wiki/Kelas_(ilmu_komputer)))

# **Constructor**

***Constructor***(dalam bahasa Indonesia : konstruktor) adalah method khusus yang akan dijalankan secara otomatis pada saat sebuah object dibuat (instansiasi), yakni ketika perintah **“*new*”** dijalankan. Constructor biasa digunakan untuk membuat proses awal dalam mempersiapkan *object*, seperti memberi nilai awal kepada property, memanggil method internal dan beberapa proses lain yang digunakan untuk mempersiapkan *object*.

Konstruktor memiliki ciri sebagai berikut :

1. Namanya sama dengan nama kelas.

2. Tidak memiliki nilai balik (dan juga tidak boleh ada kata *void* didepannya).

class Buku {

private String pengarang;

private String judul;

public Buku(String p,String j){

this.pengarang = p;

this.judul = j;

}

}

(Sumber : [*https://www.slideshare.net/WahyuWidodo12/modul-pemrograman-berorientasi-objek-90030375*](https://www.slideshare.net/WahyuWidodo12/modul-pemrograman-berorientasi-objek-90030375))

# **Hasil Percobaan dan Analisis**

# **Class pada PHP**

Penggunaan *class* dalam bahasa pemrograman PHP cukup mudah untuk dipelajari. *Source code* dalam penggunaan *class* di bahasa pemrograman PHP pun dapat dikatakan sangat sederhana. Dalam penggunaan *class* di bahasa pemrogaman PHP ini, kita perlu fokus pada *file* server.php. Selain itu, dalam penggunaan *class* di bahasa pemrograman PHP ini juga mengenal istilah *file* index.php. Sebenarnya, *file* index.php ini digunakan atau dipakai sebagai tampilan *input* oleh PHP. Tujuannya adalah supaya dapat memberikan datanya.

Berikut *source code* server.php :

<?php

include("userService.php");

$user = new userService($\_POST['email'], $\_POST['password']);

if($get\_user = $user->login()) {

echo 'Welcome '.$user->getRole();

echo ', Logged it as user email:'.$get\_user;

} else {

echo 'Invalid Login';

}

?>

Berikut *source code* index.php :

<!DOCTYPE html>

<html lang="en">

<head>

<title>Class and Constructor</title>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1">

</head>

<body>

<form action="./server.php" method="POST">

<label >System Login Kelompok 4</label><br>

<input type="email" name="email" placeholder="email">

<input type="password" name="password" placeholder="password">

<button type="submit">Submit</button>

</form>

</body>

</html>

Pada percobaan tersebut, yaitu membuat *Class* dengan bahasa pemrograman PHP. Perlu kita perhatikan pada beberapa bagian dalam program (*syntax)* tersebut. Yang pertama adalah penggunaan include("userService.php");. Perlu kita ketahui bahwa include("userService.php"); digunakan sebagai perintah untuk memanggil suatu berkas, bukan suatu *class*.

Kemudian yang kedua adalah penggunaan $\_POST, digunakan untuk mengirimkan data atau mengirimkan nilai langsung ke *action* untuk ditampung tanpa menampilkan pada URL. Informasi yang dikirim dari formulir dengan metode POST tidak terlihat oleh orang lain dan tidak memiliki batasan pada jumlah informasi untuk diprogram. Lalu selanjutnya yang ketiga yaitu penggunaan $user, digunakan dalam bahasa pemrograman PHP sebagai pemanggil *class* dari userService(). Perlu kita ketahui bahwa *parameter* tersebut dapat dilihat atau diambil dari *input*-an yang terdapat di dalam *file* index.php.

Bagian selanjutnya yang perlu kita ketahui adalah mengenai penggunaan $user->login(). Penggunaan $user->login() ini sangat penting, karena merupakan suatu cara akses *function* atau *method* yang terdapat di dalam *class* userService. Yang keempat yaitu penggunaan button yang umumnya berfungsi sebagai :

1. Tag <button> sebagai “tombol” (“button”) biasa.
2. Tag <button> untuk “submit”.
3. Tag <button> untuk “reset”.

Dalam program di atas, menggunakan fungsi untuk *reset.* Sehingga apabila ingin memasukkan *input*-an baru, tidak perlu *refresh page.*

Dan bagian terakhir yang perlu kita perhatikan adalah penggunaan echo yang berfungsi untuk mengeluarkan *output* sesuai dengan yang dihaapkan oleh pembuat program tersebut.

# **Constructor pada PHP**

Penggunaan *construction* dalam bahasa pemrograman PHP cukup mudah untuk dipelajari. *Source code* dalam penggunaan *construction* di bahasa pemrograman PHP pun dapat dikatakan sangat sederhana. Dalam penggunaan *construction* di bahasa pemrogaman PHP ini, kita perlu fokus pada *file* userService.php. Selain itu, dalam penggunaan *construction* di bahasa pemrograman PHP ini juga mengenal istilah *file* index.php. Sebenarnya, *file* index.php ini digunakan atau dipakai sebagai tampilan *input* oleh PHP. Tujuannya adalah supaya dapat memberikan datanya.

Berikut *source code* userService.php :

<?php

class userService

{

protected $email; // using protected so they can be accessed

protected $password; // and overidden if necessary

protected $dataUser; // dummy data

protected $getRole; // stores the role data

public function \_\_construct($email, $password)

{

$this->\_dataUser = [

(object) [

'email' => "nadiakelompok4@gmail.com",

'password' => "12345",

'role' => "superadmin"

],

(object) [

'email' => "riankelompok4@gmail.com",

'password' => "12345",

'role' => "user"

]

];

$this->email = $email;

$this->password = $password;

}

public function login()

{

$user = $this->checkCredentials();

if($user) {

$this->getRole = $user->role;

return $get\_data = $user->email;

} else {

return false;

}

}

protected function checkCredentials()

{

foreach($this->\_dataUser as $key => $value) {

if($value->email == $this->email && $value->password == $this->password) {

return $value;

}

}

return false;

}

public function getRole()

{

return $this->getRole;

}

}

?>

A screenshot of a cell phone

Description automatically generated

Gambar 6.1 output dalam PHP

A screenshot of a cell phone

Description automatically generated

Gambar 6.2 output dalam PHP

Dalam program di atas menggunakan bahasa pemrograman PHP. Dalam *output* program di atas, langkah pertama adalah adanya perintah untuk mengisikan atau untuk meng-*input* kan *email* yang sesuai atau *email* yang benar. Dalam contoh penerapannya, *email* yang dimasukkan adalah nadiakelompok4gmail.com. Kemudian langkah selanjutnya adalah adanya perintah untuk memasukkan atau meng-*input* kan *password* yang sesuai atau yang benar. Dalam contoh penerapannya, *password* yang dimasukkan adalah 12345. Apabila *email* dan *password* tersebut sudah benar, maka akan mengeluarkan atau meng-*output* kan kalimat Welcome superadmin, Logged it as user email : nadiakelompok4@gmail.com.

# **Class pada Python**

Penggunaan *class* dalam bahasa pemrograman python sebenarnya hampir mirip dengan penggunaan *class* dalam bahasa pemrograman PHP.

Berikut *source code* nya :

from userService import userService

print(“System Login Kelompok xx!\n”)

email = input(“Email: “)

password = input(“Password: “)

get\_class = userService(email,password)

get\_class.login()

Dalam *source code* tersebut, dapat kita perhatikan terdapat penggunaan from userService import userService dan penggunaan get\_class. Penggunaan keduanya sangat penting dalam *class* di bahasa pemrograman python. Untuk yang pertama yaitu penggunaan from userService import userService yang berfungsi untuk memanggil berkas dan juga memanggil *class.* Berkas atau *class* tersebut harus berada pada *userService*. Selanjutnya terdapat get\_class. get\_class ini merupakan sebuah objek baru. Fungsi nya untuk menjalankan *constructor* dengan memakai nama *class* tersebut. Selain itu terdapat perintah print(“System Login Kelompok xx!\n”). Perintah tersebut berfungsi untuk mengeluarkan atau meng-*output* kan kalimat yang sesuai dengan apa yang kita inginkan atau yang kita harapkan.

# **Constructor pada Python**

def \_\_init\_\_(self, email, password):

self.email = email

self.password = password

self.data = {

"nadiakelompok4@gmail.com" : {

"email" : "nadiakelompok4@gmail.com",

"password" : "12345",

"role" : "superadmin"

},

"riankelompok4@gmail.com" : {

"email" : "riankelompok4@gmail.com",

"password" : "12345",

"role" : "user"

}

}

def login(self):

get\_data = self.checkCredentials()

if get\_data:

print("\nWelcome ",get\_data['role'])

print("Logged it as user email: ",get\_data['email'])

else:

print("\nInvalid Login!\n")

def checkCredentials(self):

for value in self.data:

if value == self.email:

get\_data\_user = self.data[value]

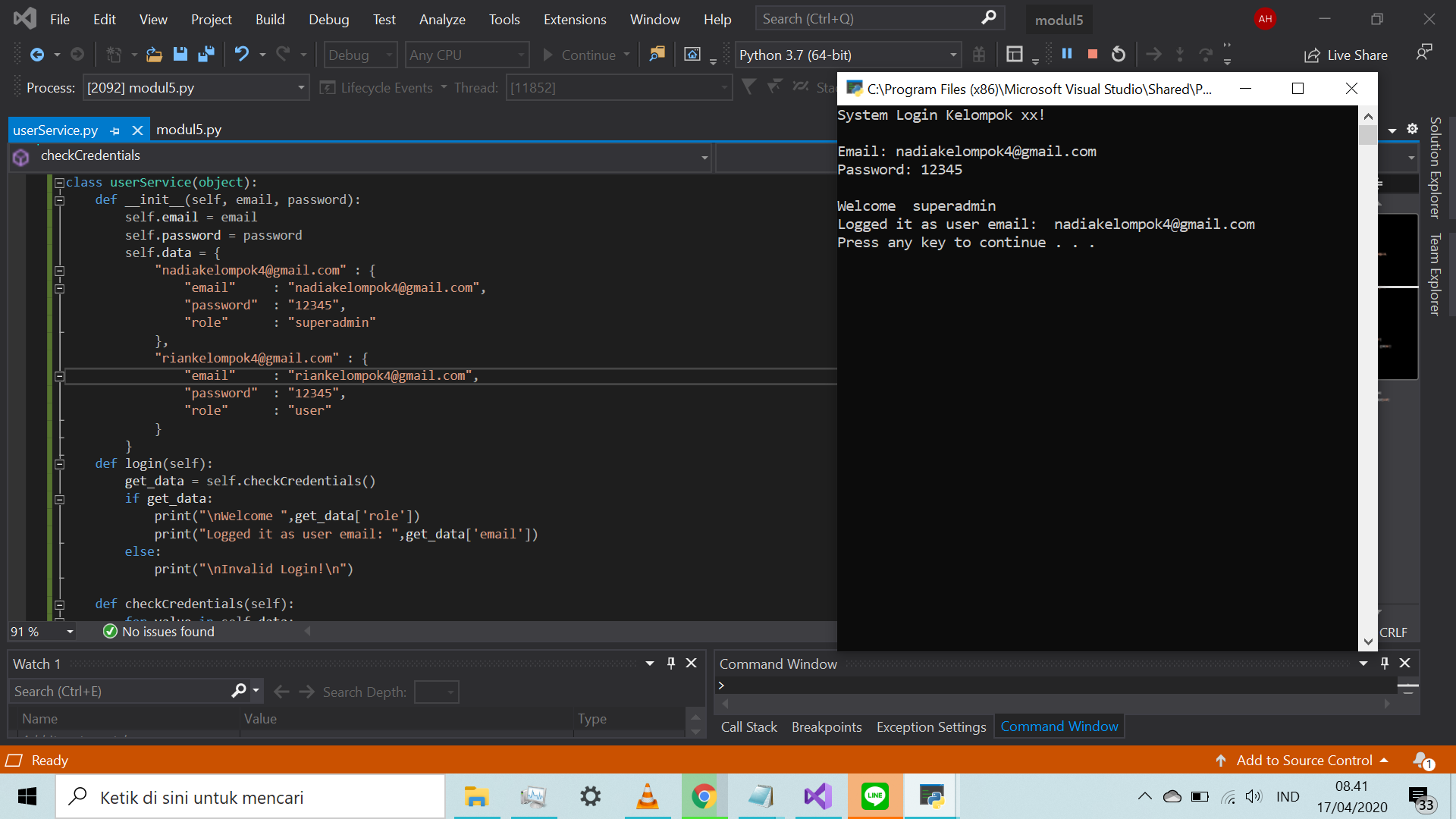
if self.password == get\_data\_user['password']:

return get\_data\_user

else:

return False

Pada *syntax* tersebut ada beberapa hal yang perlu untuk diperhatikan. Dalam bahasa pemrograman python, bisa dikatakan sangat berbeda dan unik dalam pembuatan *construction.* Keunikan nya terletak pada pemberian nama fungsinya. Dalam bahasa pemrograman python, nama fungsi nya diberi \_init\_ dengan *parameter* yang harus ada *self*. \_init\_ pada bahasa pemrograman python ini digunakan untuk konfigurasi *object* yang telah di *construct*. Pada \_init\_ *function* umumnya dilakukan inisialisasi properti atau melakukan operasi yang harus dilakukan sebelum *object* dibuat. Perlu kita perhatikan bahwa *self* ini harus diberikan pada *parameter method* atau *function* yang akses ke *construction* di dalam *class* tersebut.



Gambar 6.3 output dalam python

Pada hasil *running* pemrograman di atas, terdapat *email* dan *password* yang sudah terdapat di dalam *syntax* di atas. Di dalam *syntax* tersebut, sudah terdapat pernyataan mengenai data *email* dan *password*. Sehingga pada saat program nya di *running,* maka akan mengeluarkan *output* sesuai yang diharapkan atau yang diinginkan.

# **Class pada C#**

string email, password;

Console.Write("Email : ");

email = Console.ReadLine();

Console.Write("Password : ");

password = Console.ReadLine();

userService data = new userService(email, password);

data.login();

Console.ReadKey();

Dalam pemrograman tersebut kita perlu memperhatikan beberapa hal. Yang pertama terdapat *variable email* dan *password* dengan tipe data *string.* Yang kedua adalah penggunaan Console.Write yang berfungsi untuk mengeluarkan atau meng­-*output* kan kalimat yang kita inginkan atau yang kita harapkan. Selanjutnya adalah penggunaan *class* dalam bahasa pemrograman C# yang yaitu pembuatan *object class*. Perlu kita perhatikan dalam pembuatan *object class* di dalam bahasa pemrograman C# dapat dikatakan sangat sederhana. Pada contoh pemrograman menggunakan bahasa C# di atas terdapat userService data = new userService(email, password); . Nilai dari *parameter* tersebut diperoleh dari *input*-an *user*.

# **Construction pada C#**

private string[,] data;

private string email, password, roles = "";

public userService(string emails, string passwords)

{

email = emails;

password = passwords;

data = new string[2, 3] {

{"nadiakelompok4@gmail.com", "12345", "superadmin" },

{"riankelompok4@gmail.com", "12345", "user" }

};

}

public void login()

{

var (status, role) = checkCredentials();

if (status == true)

{

Console.WriteLine("\nWelcome " + role);

Console.WriteLine("Logged it as user email: " + email);

}

else

{

Console.WriteLine("\nInvalid Login");

}

}

private (bool, string) checkCredentials()

{

for (int i = 0; i < data.GetLength(0); i++)

{

if (data[i, 0] == email && data[i, 1] == password)

{

roles = data[i, 2];

return (true, roles);

}

}

return (false, roles);

}

Pada program tersebut menggunakan bahasa pemrograman C#. pada program di atas, menerapkan *construction*. Perlu kita perhatikan dengan seksama, *construction* akan secara otomatis dijalankan atau dieksekusi pada saat *object* pada *class user.Service* dibuat. Lalu untuk deklarasi *email* dan *variable* nilai nya diambil dari *parameter* constructor. Kemudian untuk memanggil *method* login() hanya dengan pernyataan data.login() . Dengan begitu, *method* akan terpanggil dengan baik sesuai yang diharapkan. Dan yang terakhir adalah penggunaan *array* yaitu new string[2, 3]. *Output* dari program tersebut, dapat dilihat pada gambar di bawah ini.

A screenshot of a cell phone

Description automatically generated

Gambar 6.4 output dalam C#

Pada hasil *running* pemrograman di atas, terdapat *email* dan *password* yang sudah terdapat di dalam *syntax* di atas. Di dalam *syntax* tersebut, sudah terdapat pernyataan mengenai data *email* dan *password*. Sehingga pada saat program nya di *running,* maka akan mengeluarkan *output* sesuai yang diharapkan atau yang diinginkan.

# **Class pada Java**

import java.util.Scanner;

String email, password;

Scanner input = new Scanner(System.in);

System.out.print("Email : ");

email = input.nextLine();

System.out.print("Password : ");

password = input.nextLine();

userService satu = new userService( email, password );

satu.login();

Pada program di atas menggunakan bahasa pemrograman Java. Dalam *syntax* di atas ada beberapa hal yang perlu diperhatikan. Yang pertama yaitu penggunaan import java.util.Scanner; yang berfungsi untuk menggunakan *Scanner*. *Scanner* merupakan sebuah *class* yang menyediakan fungsi-fungsi untuk mengambil *input* dari *keyword*. Lalu yang kedua ada pernyataan String email, password; . Pernyataan tersebut berarti terdapat *variable email* dan *password* dengan tipe data *string*. Kemudian terdapat pernyataan System.out.print . Pernyataan tersebut berfungsi untuk mengeluarkan atau meng-*output* kan kalimat atau teks yang kita inginkan.

Untuk membuat *object class* dalam bahasa pemrograman Java, tidak jauh berbeda dengan bahasa pemrograman C#, yaitu dengan userService satu = new userService( email, password ); . Selanjutnya untuk nilai dari *email* dan *password* tersebut, didapatkan dari *input*-an *user*.

# **Construction pada Java**

private String[][] data = new String[2][3];

private String email, password, roles = "";

//ini namanya constructor yang akan dijalankan setiap class diinisialisasikan

public userService(String emails, String passwords)

{

email = emails;

password = passwords;

String[][] data =

{

{"nadiakelompok4@gmail.com", "12345", "supedAdmin"},

{"riankelompok4@gmail.com", "12345", "user"}

};

this.data = data;

}

private boolean checkCredential()

{

for(int i = 0; i < data.length; i++ )

{

if(data[i][0].equals(email))

{

if(data[i][1].equals(password))

{

roles = data[i][2];

return true;

}

}

}

return false;

}

public void login()

{

boolean status = checkCredential();

if(status == true)

{

System.out.println("\nWelcome " + roles);

System.out.println("Logged it as user email " + email);

}

else

{

System.out.println("\nInvalid Login ");

}

}

Pada program tersebut, perlu diperhatikan beberapa hal. Selain penggunaan *class* dan *construction* yang paling utama atau paling penting, pada program tersebut pula terdapat pernyataan perulangan atau *looping* dan pernyataan pengkondisian. Pernyataan perulangan atau *looping* ditandai dengan for(int i = 0; i < data.length; i++ ). Sedangkan pernyataan pengkondisian di dalam program tersebut ditandai dengan if(data[i][0].equals(email)), if(data[i][1].equals(password)), dan if(status == true).

Selanjutnya hal yang perlu diperhatikan adalah pada *class userService*.*java* terdapat checkCredential dan *method* login. Perlu kita perhatikan, bahwa kedua nya termasuk ke dalam *method.* Fungsi dari checkCredential adalah untuk melihat (lebih kepada mengecek) apakah *input*-an yang diberikan pada *user* sama dengan data yang terdapat pada *construction*. Apabila data-data tersebut berbeda, maka program tidak akan berjalan atau tidak akan tereksekusi dengan baik dan maksimal. Kemudian fungsi dari *method* login adalah akan memberikan *output* atau keluaran berupa teks apabila *method* checkCredential() bernilai *true* atau benar.

Perlu kita perhatikan dengan seksama, *construction* akan secara otomatis dijalankan atau dieksekusi pada saat *object* pada *class user.Service* dibuat. Kemudian untuk memanggil *method* login() pada *class userService* yaitu dengan satu.login().

A screenshot of a cell phone

Description automatically generated

Gambar 6.5 output dalam Java

Pada hasil *running* pemrograman di atas, terdapat *email* dan *password* yang sudah terdapat di dalam *syntax* di atas. Di dalam *syntax* tersebut, sudah terdapat pernyataan mengenai data *email* dan *password*. Sehingga pada saat program nya di *running,* maka akan mengeluarkan *output* sesuai yang diharapkan atau yang diinginkan.

# **Tugas**

package tugasmodul\_5;

import java.util.Scanner;

public class Tugasmodul\_5 {

public static void main(String[] args) {

String email, password;

Scanner input = new Scanner(System.in);

System.out.print("Email : ");

email = input.nextLine();

System.out.print("Password : ");

password = input.nextLine();

userService satu = new userService( email, password );

satu.login();

}

}

package tugasmodul\_5;

public class userService {

private String[][] data = new String[2][3];

private String[][] histori = new String [2][5];

private String email, password, roles = "";

int buku;

//ini namanya constructor yang akan dijalankan setiap class diinisialisasikan

public userService(String emails, String passwords)

{

email = emails;

password = passwords;

String[][] data =

{

{"nadiakelompok4@gmail.com", "12345", "supedAdmin"},

{"riankelompok4@gmail.com", "12345", "user"}

};

String[][] histori =

{

{"nadiakelompok4@gmail.com","Fisika Dasar II","Dasar Komputer dan Pemrograman","Sistem Digital","21-04-2020"},

{"riankelompok4@gmail.com","Konsep Jaringan Komputer","Dasar Komputer dan Pemrograman","Kalkulus","21-04-2020"}

};

this.data = data;

this.histori = histori;

}

private boolean checkCredential()

{

for(int i = 0; i < data.length; i++ )

{

if(data[i][0].equals(email))

{

if(data[i][1].equals(password))

{

roles = data[i][2];

return true;

}

}

}

return false;

}

public void login()

{

boolean status = checkCredential();

if(status == true)

{

System.out.println("\nWelcome " + roles);

System.out.println("Logged it as user email " + email);

System.out.println(email + "Telah Meminjam Buku, yaitu : " );

for (int i=1;i<4;i++) {

System.out.println(histori[buku][i]);

}

System.out.println("Tanggal Peminjamannya, yaitu pada : " + histori[buku][4]);

}

else

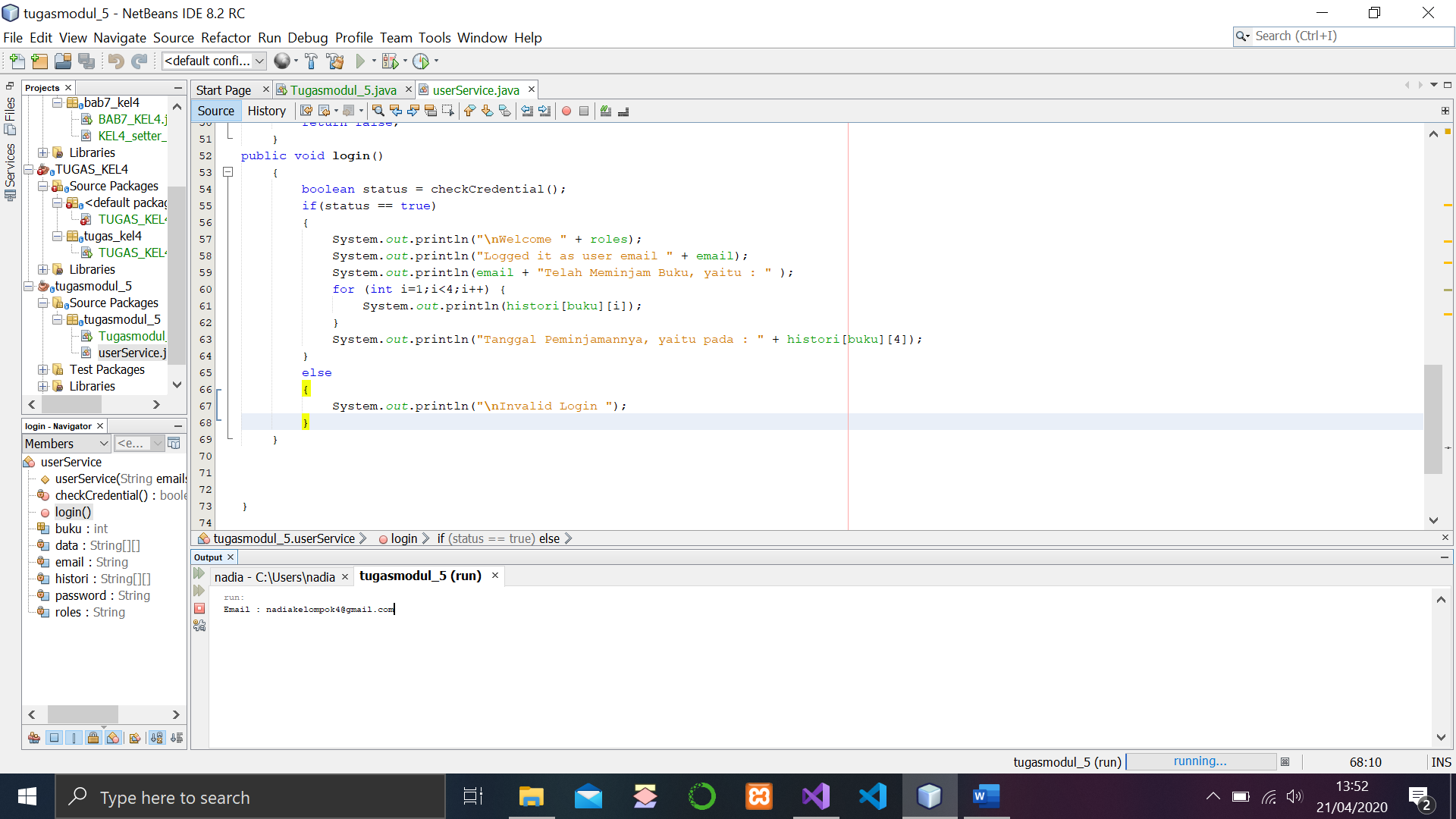
{

System.out.println("\nInvalid Login ");

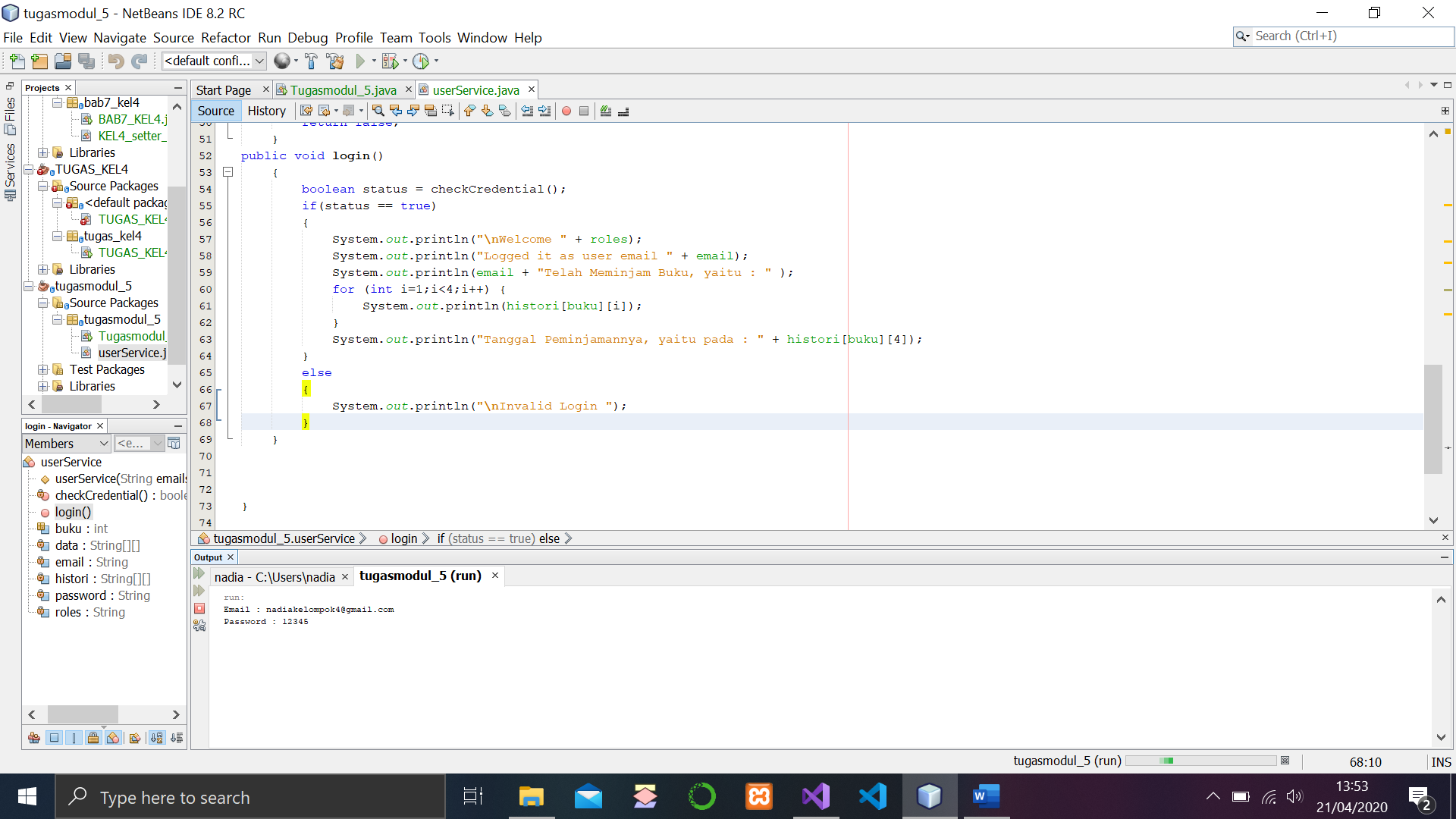
}

}

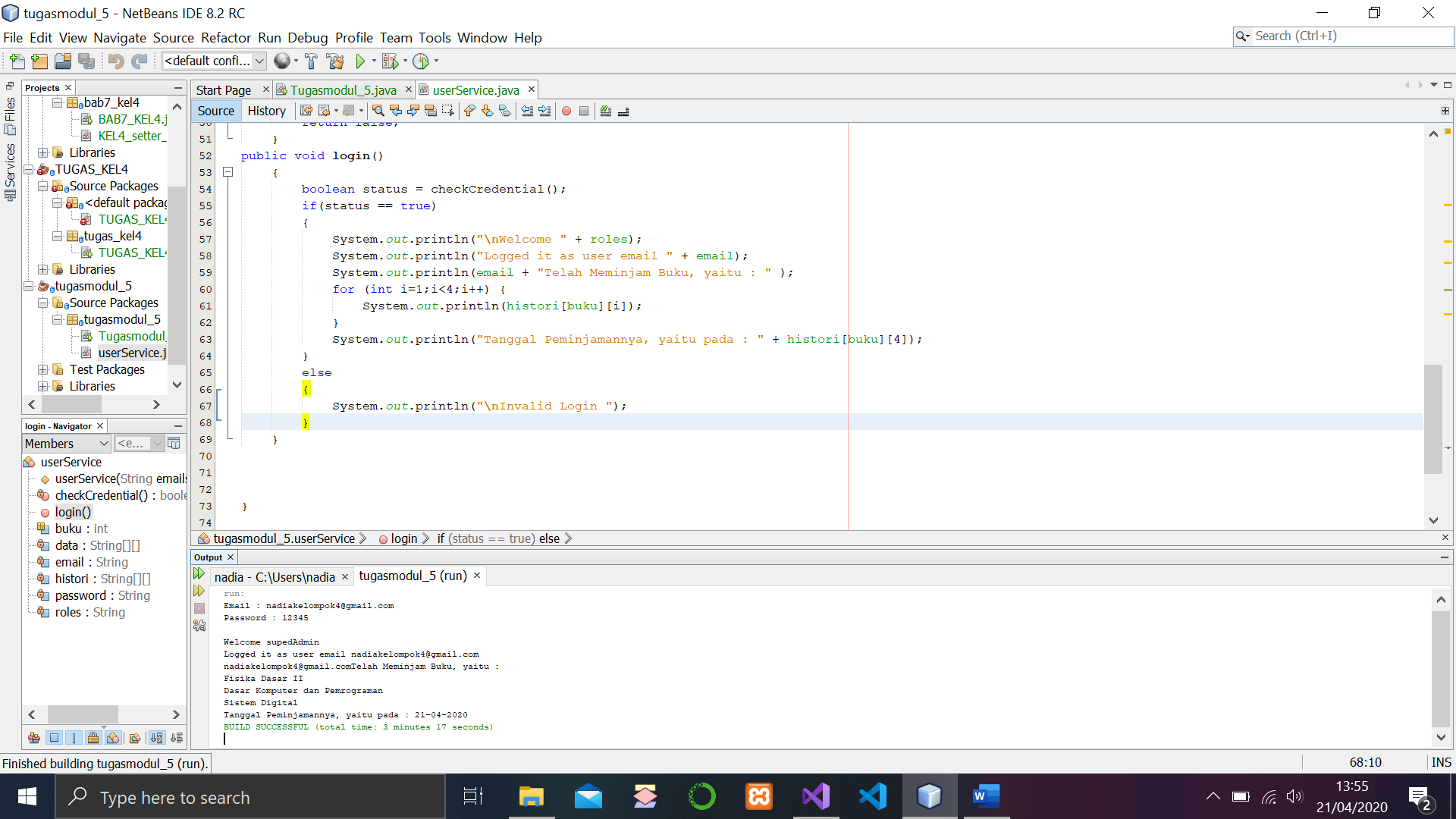
}



Gambar 6.6 output 1 tugas



Gambar 6.7 output 2 tugas



Gambar 6.8 output 3 tugas

Pada program tersebutmenggunakan bahasa pemrograman Java. Dalam *syntax* di atas ada beberapa hal yang perlu diperhatikan. Yang pertama yaitu penggunaan import java.util.Scanner; yang berfungsi untuk menggunakan *Scanner*. *Scanner* merupakan sebuah *class* yang menyediakan fungsi-fungsi untuk mengambil *input* dari *keyword*. Lalu yang kedua ada pernyataan String email, password; . Pernyataan tersebut berarti terdapat *variable email* dan *password* dengan tipe data *string*. Kemudian terdapat pernyataan System.out.print . Pernyataan tersebut berfungsi untuk mengeluarkan atau meng-*output* kan kalimat atau teks yang kita inginkan.

Untuk membuat *object class* dalam bahasa pemrograman Java, tidak jauh berbeda dengan bahasa pemrograman C#, yaitu dengan userService satu = new userService( email, password ); . Selanjutnya untuk nilai dari *email* dan *password* tersebut, didapatkan dari *input*-an *user*. Sedangkan untuk keterangan peminjaman buku dan tanggal peminjamannya sudah ditentukan nilainya.

Pada program tersebut, juga perlu diperhatikan beberapa hal. Selain penggunaan *class* dan *construction* yang paling utama atau paling penting, pada program tersebut pula terdapat pernyataan perulangan atau *looping* dan pernyataan pengkondisian. Pernyataan perulangan atau *looping* ditandai dengan for(int i = 0; i < data.length; i++ ). Sedangkan pernyataan pengkondisian di dalam program tersebut ditandai dengan if(data[i][0].equals(email)), if(data[i][1].equals(password)), dan if(status == true).

Selanjutnya hal yang perlu diperhatikan adalah pada *class userService*.*java* terdapat checkCredential dan *method* login. Perlu kita perhatikan, bahwa kedua nya termasuk ke dalam *method.* Fungsi dari checkCredential adalah untuk melihat (lebih kepada mengecek) apakah *input*-an yang diberikan pada *user* sama dengan data yang terdapat pada *construction*. Apabila data-data tersebut berbeda, maka program tidak akan berjalan atau tidak akan tereksekusi dengan baik dan maksimal. Kemudian fungsi dari *method* login adalah akan memberikan *output* atau keluaran berupa teks apabila *method* checkCredential() bernilai *true* atau benar.

Perlu kita perhatikan dengan seksama, *construction* akan secara otomatis dijalankan atau dieksekusi pada saat *object* pada *class user.Service* dibuat. Kemudian untuk memanggil *method* login() pada *class userService* yaitu dengan satu.login().

Pada hasil *running* pemrograman di atas, terdapat *email* dan *password* yang sudah terdapat di dalam *syntax* di atas. Di dalam *syntax* tersebut, sudah terdapat pernyataan mengenai data *email* dan *password*. Sehingga pada saat program nya di *running,* maka akan mengeluarkan *output* sesuai yang diharapkan atau yang diinginkan.

Pada penerapan program tersebut, *user* meng-*input* kan *email* nadiakelompok4@gmail.com dan meng-*input* kan *password* 12345. Kemudian hasil yang didapatkan adalah informasi berupa status *email* tersebut yaitu sebagai supedAdmin dan juga informasi mengenai peminjaman buku, yang berupa jenis buku yang dipinjam dan juga tanggal peminjamannya.

Link GitHub : https://github.com/Nadia130201/tugas-kuliah.git

# **Kesimpulan**

1. Di dalam bahasa pemrograman PHP, penulisan *class* diawali dengan *keyword* *class* kemudian akan diikuti dengan nama dari *class* tersebut.
2. Penulisan *method construct* pada bahasa pemrograman PHP diawali dengan *syntax function*, lalu di ikuti dengan \_\_*construct. (double underscore dan construct).* Fungsi *construct* ini tidak perlu dipanggil lagi pada saat instansiasi karena *construct* akan di jalankan otomatis jika *class*-nya yang diinstansiasi.
3. *Constructor* adalah *method* khusus yang akan dijalankan secara otomatis pada saat sebuah *object* dibuat (instansiasi), yakni ketika perintah “*new*” dijalankan.
4. Dalam *project* java terdapat tipe *class* yaitu *driver class*, *class driver* adalah *class* yang memiliki *method* *main* di dalamnya,
5. *Object class* adalah sebuah *class* yang menyediakan definisi untuk *object* yang akan di buat di *driver class*. *Class* ini dapat di buat sendiri atau menggunakan *class* yang ada pada *Library Java*.
6. *Keyword* *this* pada *constructor* adalah *keyword* yang menunjukan bahwa *variable* tersebut mereferensikan dengan *variable* level *class*.
7. *Plain Old Java Object*, merupakan *class* yang berdiri sendiri dan tidak memiliki ketergantungan dengan *class* lain baik itu *extends* maupun *implements*.
8. *Default* *Constructor* akan melakukan 3 hal kepada *variable* *class* yaitu:  
   - Mengisi semua *variable* bertipe *numeric(float, int, double,* dan sebagainya) menjadi 0.  
   - Mengisi semua *variable* bertipe *boolean* menjadi *false*.  
   - Mengisi semua *variable* bertipe *string* menjadi *null*.