JOIN (Jurnal Online Informatika)
p-ISSN: 2528-1682, e-ISSN: 2527-9165
Volume 8 Number 1 | June 2023: 1-3
DOI: 10.15575/join.xxxxx.xx

Pembuatan Program Sederhana Pemesanan Kamar Kos Menggunakan Java

First Author[1], Second Author[2], Third Author[3] (10 pt)
[1,3]Department of Informatics, UIN Sunan Gunung Djati Bandung, Indonesia (9 pt)
[2]Department of Mathematics, UIN Sunan Gunung Djati Bandung, Indonesia (9 pt)

Article Info

Article history:

Received, 2023
Revised, 2023
Accepted , 2023

Keywords:

Pemesanan kamar kos
Pemerograman berorientasi objek java
Program sede
OOP

ABSTRACT (10 PT)

Penggunaan teknologi dalam memfasilitasi pemesanan kamar kos menjadi penting dalam mengoptimalkan proses tersebut. Penelitian ini bertujuan untuk mengembangkan sebuah program komputer sederhana menggunakan bahasa pemrograman Java yang memungkinkan pengguna untuk melakukan pemesanan kamar kos secara efisien. Metode pengembangan perangkat lunak yang digunakan adalah pemrograman berorientasi objek (OOP) dan desain antarmuka yang intuitif. Program ini memungkinkan pengguna untuk melihat ketersediaan kamar, melakukan reservasi, dan mengelola informasi pemesanan. Hasil pengujian menunjukkan bahwa program yang dibangun dapat menjalankan fungsi-fungsi dasar pemesanan kamar kos dengan baik dan memberikan pengalaman pengguna yang memuaskan. Implementasi program ini diharapkan dapat meningkatkan efisiensi dalam proses pemesanan kamar kos serta memberikan landasan bagi pengembangan sistem serupa di masa mendatang.

Corresponding Author:

Rifqi Ilhami Fauzi,
Informatics Department, Faculty of Science & Technology, UIN Sunan Gunung Djati Bandung
Jl. A. H. Nasution No. 105, Cibiru, Bandung, Indonesia. 40614
Email: join@uinsgd.ac.id

1. PENDAHULUAN

Penggunaan teknologi dalam berbagai aspek kehidupan sehari-hari telah mengubah cara kita melakukan berbagai aktivitas, termasuk dalam hal pemesanan akomodasi seperti kamar kos. Ketersediaan teknologi dan kemudahan akses terhadap komputer serta perangkat mobile telah mendorong perkembangan sistem pemesanan yang lebih efisien dan efektif.

Dalam konteks ini, pembuatan sebuah program komputer sederhana untuk pemesanan kamar kos memiliki nilai yang signifikan. Program ini diharapkan dapat memberikan solusi dalam memudahkan proses pemesanan kamar kos, yang seringkali melibatkan sejumlah proses manual yang rentan terhadap kesalahan dan memakan waktu.

Penelitian ini bertujuan untuk mengembangkan sebuah program sederhana menggunakan bahasa pemrograman Java yang fokus pada pemesanan kamar kos. Program ini diharapkan mampu menyajikan informasi tentang ketersediaan kamar, memfasilitasi proses reservasi, serta menyediakan fungsi manajemen data pemesanan bagi pengguna dengan antarmuka yang mudah dipahami.
Melalui pemrograman berorientasi objek dan penggunaan bahasa Java yang populer dan fleksibel, program ini diharapkan dapat memberikan kontribusi dalam mempermudah dan meningkatkan

efisiensi proses pemesanan kamar kos. Dengan adanya sistem ini, diharapkan dapat mengurangi waktu yang dibutuhkan dalam melakukan pemesanan, meningkatkan akurasi data, serta memberikan pengalaman yang lebih baik bagi pengguna dalam mendapatkan kamar kos yang diinginkan.

Penelitian ini menjadi langkah awal dalam pengembangan teknologi yang dapat diimplementasikan dalam skala yang lebih luas, memungkinkan adaptasi sistem serupa dalam manajemen pemesanan akomodasi lainnya di masa depan.

2. METODE

Metode yang di gunakan terdapat beberapa tahapan diantaranya :

2.1 Study literatur.

Metode penelitian ini dimulai dengan tahap desain penelitian yang mencakup analisis kebutuhan sistem pemesanan kamar kos serta perancangan konseptual program. Desain ini mengintegrasikan struktur data untuk informasi kamar kos, harga kamar, dan proses reservasi. Proses ini melibatkan implementasi logika bisnis seperti pengecekan ketersediaan kamar, proses reservasi, dan manajemen data pengguna.

2.2 Pengembangan mini project.

Tahap ini dilakukan untuk memulai sebuah mini project. Mini project yang di buat adalah program sederhana yang bisa menjalankan proses transaksi penyewaan kamar kos dan pencetakan struk atas data yang di inputkan. Mini project ini di buat dengan menggunakan bahasa pemrograman Java. Algoritma yang digunakan didasarkan pada pengaturan ketersediaan kamar, proses reservasi, dan pengelolaan basis data sederhana.

2.3 Pengujian mini project

Pengujian program dilakukan secara bertahap dengan pengujian fungsional untuk memastikan setiap fitur berjalan sesuai dengan yang diharapkan. Selain itu, pengujian ketersediaan kamar juga dilakukan untuk memverifikasi keakuratan informasi yang ditampilkan kepada pengguna. Data yang diperoleh dari pengujian ini mencakup hasil dari setiap langkah pengujian fungsionalitas dan ketersediaan kamar.

2.4 Evaluasi dan analisis

Proses pengumpulan data juga melibatkan penggunaan program secara langsung oleh kelompok pengguna yang telah ditentukan. Data ini mencakup pengalaman pengguna dalam menggunakan program, umpan balik terkait antarmuka pengguna, dan kelancaran proses pemesanan. Dengan memadukan pengujian fungsional dan data pengguna, evaluasi program dilakukan untuk mengevaluasi kehandalan, keefektifan, dan kepuasan pengguna terhadap program yang dibangun.

3. HASIL DAN PEMBAHASAN

3.1 Pengertian OOP (Object Oriented Programing)

OOP adalah merupakan kepanjangan dari *Object Oriented Programming*. OOP merupakan suatu metode pemrograman yang berorientasi kepada objek. Dalam bahasa Indonesia OOP dikenal dengan PBO (Pemrograman Berorientasi Objek). OOP bertujuan untuk mempermudah pengembangan sebuah program. Ia memiliki variabel dan fungsi yang dibungkus ke dalam objek ataupun *class*. Keduanya dapat saling berinteraksi sehingga membentuk sebuah program.

Seorang programmer harus mampu meminimalisir program dan membuatnya tertata rapi. Selain itu juga untuk mempercepat pembuatan aplikasi. Jikalau programnya semakin besar maupun kompleks maka semua akan membuat kode sulit di-*maintenance*. Teknik prosedural identik dengan menggabungkan seluruh kode. Bayangkan jika kita membuat program sebesar GOJEK dengan teknik prosedural, kita akan kesulitan untuk memodifikasi kode program tersebut. Malah akan dibuat pusing jika seluruh program disatukan tanpa mengorganisasikan kode program. Itulah alasan mengapa harus menggunakan teknik OOP. Berikut merupakan komponen OOP yang penting untuk dipelajari:

1. Class Dan Object

   Class bertugas untuk mengumpulkan prosedur/fungsi dan variabel dalam satu tempat. Class merupakan *blueprint* dari sebuah objek atau cetakan untuk membuat objek. Contoh *class* sebagai berikut ini:

```
1 public class Car{
2
3     // Body Class
4
5 }
```

Gambar 3.1: Contoh Class dan Object

Class akan merepresentasikan objek yang mau dibuat. Jadi dalam membuat nama kelas harus disesuaikan dengan objek yang akan dibuat. Penulisan nama *class* memiliki aturan. Yakni dengan format **PascalCase** yaitu penulisannya diiawali dengan huruf kapital. Jika nama variabel tersusun dari dua kata atau lebih maka tidak perlu diberi spasi di antaranya dan diawali dengan huruf kapital pula.

Misal: *class* **MakananKucing**, *class* **Senjata**, dan *class* **SignIn**.

Sedangkan object adalah sebuah variabel instance yang merupakan wujud dari class. Istance merupakan wujud dari sebuah kelas. Sebuah objek di gambarkan dengan variable dan method.

Class berisi dari beberapa kumpulan definisi variabel dan fungsi yang menggambarkan sebuah objek. Tidak hanya terdiri dari class, OOP juga terdiri dari beberapa elemen penyusun lainnya seperti *attribute*, *method*, *inheritance* dan sebagainya.

2. Atribut

Atribut merupakan bagian dari sebuah kelas yang masih berhubungan erat dari kelas tersebut. Atribut bisa juga disebut sebagai properti atau *properties* dari sebuah *class*. Contohnya ketika kamu punya sebuah *class* Motor, maka kamu dapat menambahkan atribut seperti kecepatan motor, umur motor, ukuran, ban, warna dsb. Untuk lebih detailnya, kita contohkan pada program berikut:

```
class Car {

    // Attribute
    int speed;
    int tire;

}
```

Gambar 3.2 : Contoh Atribut

Penggunaan atribut berlaku dari kurung kurawal awal (**{**) sampai dengan sebelum kurung kurawal akhir (**}**). Ini dinamakan *scope*. Untuk penulisannya menggunakan format **lowerCamelCase**. Jadi untuk kata pertama diawali dengan huruf kecil, sedangkan kata selanjutnya diawali dengan huruf kapital. Sama seperti PascalCase, penggunaan spasi tidak diperkenankan ketika menghubungkan dua kata atau lebih dari sebuah nama properti.

Contohnya **length**, **width**, **apple**, **speed**, **listMovies**, dll.

3. Method

Method berperan menjelaskan bagaimana suatu atribut beraksi. Peran yang dimaksud berupa tingkah laku (*behavior*) yang dapat digambarkan oleh suatu *method*. Misal *class* Manusia. Manusia tentu memiliki *method* berupa tingkah laku, seperti berpikir, berjalan, berbicara, makan dll. Maka tentunya *method* dapat disesuaikan dengan program yang dibuat.

Contoh, Pertama buat kelas hewan

```
public class Hewan{

    // Method dari kelas Hewan
    void lari() {
        System.out.println("Berlari dengan sangat cepat.");
    }

    void jalan() {
        System.out.println("Berjalan sambil melompat.");
    }

    void makan() {
        System.out.println("Makan wortel dengan menggunakan mulutnya");
    }

}
```

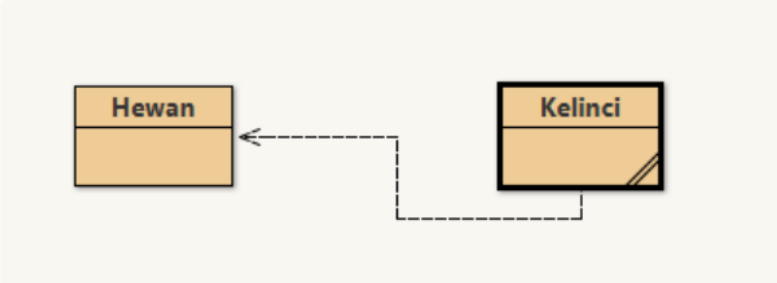
Gambar 3.3: Contoh Method

Kemudian buat kelas kelinci

```
public class Kelinci {

    public static void main(String[] args) {

        // Memanggil kelas Hewan
        Hewan kelinci = new Hewan();
        kelinci.makan(); // Method atau tingkah laku hewan kelinci
        kelinci.jalan();
        kelinci.lari();

    }

}
```

Gambar 3.4: Contoh method

Tabel *class* Hewan dan *class* Kelinci saling berkaitan. Artinya kelas Kelinci dapat memanggil method dari *class* Hewan.



Gambar 3.5: Tabel Antara Dua Kelas Saling Berhubungan

kemudian di tambahkan property tambahan :

```
public class Hewan {

  // Properti
   double tinggi = 20;
  double berat = 4;

  // Inisialisasi properti melalui konstruktor
  int umur;

  // Konstruktor dengan parameter
  Hewan(int umur) {
    this.umur = umur;
  }

  void lari() {
    System.out.println("Berlari dengan sangat cepat..");
  }

     void jalan() {
            System.out.println("Berjalan sambil melompat.");
      }

      void makan() {
            System.out.println("Makan wortel dengan menggunakan mulutnya");
      }

      int getUmur() {
            return umur;
      }

      double getBerat() {
            return berat;
      }

      double getTinggi() {
            return tinggi;
      }

}
```

Gambar 3.6: Property Yang di Tambahkan ke Method

Kemudian tambahkan beberapa kode program pada *class* Kelinci.:

```
public class Kelinci {

    public static void main(String[] args) {

        // Memanggil kelas Hewan
        Hewan kelinci = new Hewan(4);
        kelinci.makan(); //Method atau tingkah laku hewan kelinci
        kelinci.jalan();
        kelinci.lari();

        // Penggunaan fungsi getUmur dari class Hewan
        System.out.println("Umur Kelinci adalah" + kelinci.getUmur() + "tahun");

        // Perhitungan indeks massa tubuh (BMI)
        // Rumus: berat(kg) / ( tinggi(m) * tinggi(m) )
        double bmi = kelinci.getBerat() / ((kelinci.getTinggi() * 0.01) *
(kelinci.getTinggi() * 0.01));

        // Hasilnya
        System.out.println("Indeks massa tubuhnya adalah " + bmi);

    }

}
```

Gambar 3.7 : Kode Program Pada Kelas Kelinci

4. *Inheritance*

Inheritance adalah hubungan antara dua objek atau lebih. Di mana terdapat sebuah objek utama yang mewariskan attribute maupun *method* yang dimilikinya kepada objek lainnya, baik itu sebagian maupun keseluruhan.

Contohnya seekor anak kucing berjenis mamalia, mewarisi sifat dan juga bentuk fisik orang tuanya, seperti bulu, mata, telinga, bahkan suaranya.

Suatu objek diwariskan dengan menggunakan keyword *extends*. Baik, pertama buatlah *class* **Hewan**. *Class* Hewan merupakan *parent class* atau orang tua dari semua kelas.[1] Contoh source kode :

```
public class Kucing extends Hewan {

    public Kucing() {

        super(); // Akan tetap memanggil constructor "hewan mamalia" dari parent Class
        System.out.println("Kucing Bersuara Meong");

    }

}
```

[1] Dicoding. *Apa aitu OOP pada java beserta contohnya.* Diakses pada 22 Desember 2023 dari https://www.dicoding.com/blog/apa-itu-oop-pada-java-beserta-contohnya/

Gambar 3.8 : Contoh Source Inheritance

Kemudian Buat *class* Kucing, yang artinya merupakan turunan dari *class* Hewan. Gunakanlah keyword *extends*.

```
public class Kucing extends Hewan {

    public Kucing() {

        super(); // Akan tetap memanggil constructor "hewan mamalia" dari parent Class
        System.out.println("Kucing Bersuara Meong");

    }

}
```

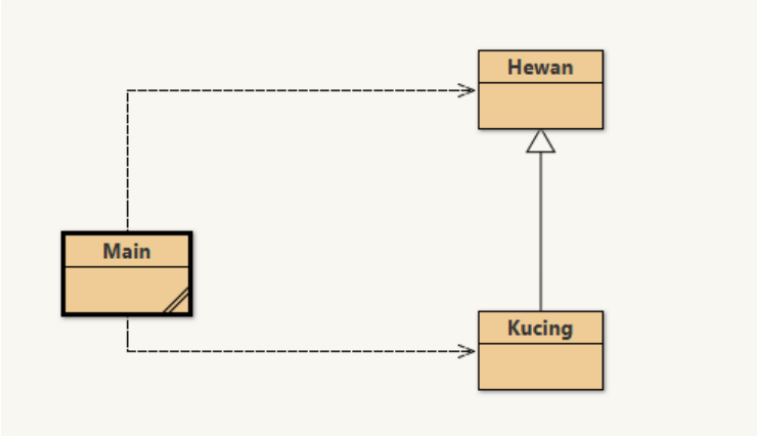
Gambar 3.9 : Contoh Pembuatan Class Kucing di Inheritance

Dan yang terakhir buatlah *class* dengan nama Main. Kelas ini berguna untuk menjalankan atau mengeksekusi sebuah program.

```
public class Main {

   public static void main(String[] args) {

        Hewan hewan = new Hewan(); // Memanggil class Hewan
        System.out.println("Apakah hewan IS-A Objek -> " + (hewan instanceof Object));
        System.out.println("Apakah hewan IS-A Hewan -> " + (hewan instanceof Hewan));
        System.out.println("Apakah hewan IS-A Kucing -> " + (hewan instanceof Kucing));
        System.out.println("------------------------"); // spasi

        Kucing kucing = new Kucing(); // Memanggil class Kucing "Kucing bersuara meong"
        System.out.println("Apakah hewan IS-A Objek -> " + (kucing instanceof Object));
        System.out.println("Apakah kucing IS-A Hewan -> " + (kucing instanceof Hewan));
        System.out.println("Apakah kucing IS-A Kucing -> " + (kucing instanceof Kucing));

    }

}
```

Gambar 3.10: Contoh Pembuatan Kelas Main

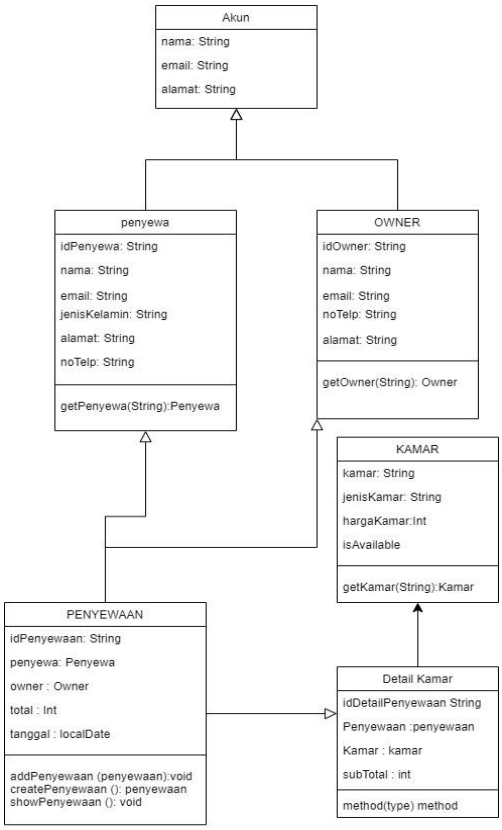
Jika *coding* benar, maka tabel akan berbentuk seperti gambar di bawah ini:



Gambar 3.11 : Contoh Tabel Yang Terbentuk Jika Codingan Benar

3.2 pengimplementasian source code pemesanan kamar kos

1. Gambar Class Diagram



Gambar 3.12: Class Diagram

2. Source code

a. Class akun

```
package model;

public class Akun {
    private String nama;
    private String email;
    private String alamat;

    //Set Methods
    public Akun setNama(String nama) {
        this.nama = nama;
        return this;
    }

    public Akun setEmail(String email) {
        this.email = email;
        return this;
    }

    public Akun setAlamat(String alamat) {
        this.alamat = alamat;
        return this;
    }

    //Get Methods
    public String getNama() {
        return nama;
    }

    public String getEmail() {
        return email;
    }

    public String getAlamat() {
        return alamat;
    }

}
```

b. Class penyewa

```
package model;

public class Penyewa extends Akun {
    private String idPenyewa;
    private String jenisKelamin;
    private String noTelp;
```

```
    public String getIdPenyewa() {
        return idPenyewa;
    }

    public Penyewa setIdPenyewa(String idPenyewa) {
        this.idPenyewa = idPenyewa;
        return this;
    }

    public String getJenisKelamin() {
        return jenisKelamin;
    }

    public Penyewa setJenisKelamin(String jenisKelamin) {
        this.jenisKelamin = jenisKelamin;
        return this;
    }

    public String getNoTelp() {
        return noTelp;
    }

    public Penyewa setNoTelp(String noTelp) {
        this.noTelp = noTelp;
        return this;
    }

}
```

c. Class owner

```
package model;

public class Owner extends Akun {
    private String idOwner;
    private String noTelp;

    public String getIdOwner() {
        return idOwner;
    }

    public String getNoTelp() {
        return noTelp;
    }

    public Owner setIdOwner(String idOwner) {
        this.idOwner = idOwner;
        return this;
    }

    public Owner setNoTelp(String noTelp) {
                this.noTelp = noTelp;
                return this;
            }
        }
```

```
104.
105.
```

d. Class kamar

```
106.  package model;
107.
108.  public class Kamar {
109.      private String noKamar;
110.      private String jenisKamar;
111.      private int hargaKamar;
112.      private boolean isAvailable;
113.
114.      //Set Method
115.      public Kamar setIdKamar(String noKamar) {
116.          this.noKamar = noKamar;
117.          return this;
118.      }
119.
120.      public Kamar setJenisKamar(String jenisKamar) {
121.          this.jenisKamar = jenisKamar;
122.          return this;
123.      }
124.
125.      public Kamar setHargaKamar(int hargaKamar) {
126.          this.hargaKamar = hargaKamar;
127.          return this;
128.      }
129.
130.      public Kamar setIsAvailable(boolean isAvailable) {
131.          this.isAvailable = isAvailable;
132.          return this;
133.      }
134.
135.      //Get Method
136.      public String getIdKamar() {
137.          return noKamar;
138.      }
139.
140.      public String getJenisKamar() {
141.          return jenisKamar;
142.      }
143.
144.      public int getHargaKamar() {
145.          return hargaKamar;
146.      }
147.
148.      public boolean isIsAvailable() {
149.          return isAvailable;
150.      }
151.
152.  }
153.
154.
```

e. Calss detai kamar

```
package model;

public class DetailKamar {
    private String idDetailKamar;
    private Penyewaan penyewaan;
    private Kamar Kamar;
    private int hargaJual;
    private int subTotal;

    public String getIdDetailKamar() {
        return this.idDetailKamar;
    }

    public void setIdDetailKamar(String idDetailKamar) {
        this.idDetailKamar = idDetailKamar;
    }

    public Penyewaan getPenyewaan() {
        return this.penyewaan;
    }

    public void setPenyewaan(Penyewaan penyewaan) {
        this.penyewaan = penyewaan;
    }

    public Kamar getKamar() {
        return this.Kamar;
    }

    public void setKamar(Kamar Kamar) {
        this.Kamar = Kamar;
    }

    public int getHargaJual() {
        return this.hargaJual;
    }

    public void setHargaJual(int hargaJual) {
        this.hargaJual = hargaJual;
    }


    public int getSubTotal() {
        return this.subTotal;
    }

    public void setSubTotal(int subTotal) {
        this.subTotal = subTotal;
    }

}
```

f. Class penyewaan

```
208.  package model;
209.  import java.time.LocalDate;
210.  import java.util.ArrayList;
211.
212.  public class Penyewaan {
213.      private String IdPenyewaan;
214.      private Penyewa penyewa;
215.      private Owner owner;
216.      private int total;
217.      private LocalDate tanggal;
218.      private ArrayList<DetailKamar> detailKamars;
219.
220.      public String getIdPenyewaan() {
221.          return this.IdPenyewaan;
222.      }
223.
224.      public Penyewaan setIdPenyewaan(String IdPenyewaan) {
225.          this.IdPenyewaan = IdPenyewaan;
226.          return this;
227.      }
228.
229.      public Penyewa getPenyewa() {
230.          return penyewa;
231.      }
232.
233.      public Penyewaan setPenyewa(Penyewa penyewa) {
234.          this.penyewa = penyewa;
235.          return this;
236.      }
237.
238.      public Owner getOwner() {
239.          return owner;
240.      }
241.
242.      public Penyewaan setOwner(Owner owner) {
243.          this.owner = owner;
244.          return this;
245.      }
246.
247.      public int getTotal() {
248.          return total;
249.      }
250.
251.      public Penyewaan setTotal(int total) {
252.          this.total = total;
253.          return this;
254.      }
255.
256.      public LocalDate getTanggal() {
```

```
257.                return tanggal;
258.            }
259.
260.            public Penyewaan setTanggal(LocalDate tanggal) {
261.                this.tanggal = tanggal;
262.                return this;
263.            }
264.
265.            public ArrayList<DetailKamar> getDetailKamars() {
266.                return detailKamars;
267.            }
268.
269.             public  Penyewaan  setDetailKamars(ArrayList<DetailKamar>
    detailKamars) {
270.                this.detailKamars = detailKamars;
271.                return this;
272.            }
273.
274.
275.
276.        }
277.
278.
```

g. AppController.java

```
279.        package controller;
280.            import java.io.BufferedReader;
281.            import java.io.InputStreamReader;
282.            import java.time.LocalDate;
283.            import java.util.ArrayList;
284.            import java.util.HashMap;
285.            import java.util.Map;
286.            import java.util.Scanner;
287.
288.            import model.Kamar;
289.            import model.Penyewa;
290.            import model.Owner;
291.            import model.Penyewaan;
292.            import model.DetailKamar;
293.
294.        public class AppController {
295.            //Fauzii
296.            HashMap<String, Kamar> kamars = new HashMap();
297.            HashMap<String, Penyewa> penyewas = new HashMap();
298.            HashMap<String, Owner> owners = new HashMap();
299.            ArrayList<Penyewaan>penyewaans = new ArrayList<>();
300.
301.            public void setUp() {
302.                Kamar kamarA1 = new Kamar();
303.                Kamar kamarA2 = new Kamar();
304.                Kamar kamarA3= new Kamar();
305.                Kamar kamarB1= new Kamar();
306.                Kamar kamarB2= new Kamar();
307.                Kamar kamarB3= new Kamar();
308.
```

```
309.                kamarA1.setJenisKamar("Reguler(Kasur, Lemari, Wc diluar)")
310.                    .setIdKamar("A1")
311.                    .setHargaKamar(500000)
312.                    .setIsAvailable(true);
313.
314.                kamarA2.setJenisKamar("Reguler(Kasur, Lemari, Wc diluar)")
315.                    .setIdKamar("A2")
316.                    .setHargaKamar(500000)
317.                    .setIsAvailable(true);
318.
319.                kamarA3.setJenisKamar("Reguler(Kasur, Lemari, Wc diluar)")
320.                    .setIdKamar("A3")
321.                    .setHargaKamar(500000)
322.                    .setIsAvailable(true);
323.
324.            kamarB1.setJenisKamar("VIP(Kasur, Lemari, Wc didalam, AC)")
325.                    .setIdKamar("B1")
326.                    .setHargaKamar(1000000)
327.                    .setIsAvailable(true);
328.
329.            kamarB2.setJenisKamar("VIP(Kasur, Lemari, Wc didalam, AC)")
330.                    .setIdKamar("B2")
331.                    .setHargaKamar(1000000)
332.                    .setIsAvailable(true);
333.
334.            kamarB3.setJenisKamar("VIP(Kasur, Lemari, Wc didalam, AC)")
335.                    .setIdKamar("B3")
336.                    .setHargaKamar(1000000)
337.                    .setIsAvailable(true);
338.
339.            kamars.put(kamarA1.getIdKamar(), kamarA1);
340.            kamars.put(kamarA2.getIdKamar(), kamarA2);
341.            kamars.put(kamarA3.getIdKamar(), kamarA3);
342.            kamars.put(kamarB1.getIdKamar(), kamarB1);
343.            kamars.put(kamarB2.getIdKamar(), kamarB2);
344.            kamars.put(kamarB3.getIdKamar(), kamarB3);
345.
346.            Owner Owner1 = new Owner();
347.
348.            Owner1.setIdOwner("P1")
349.                    .setNoTelp("085797035322")
350.                    .setNama("Rifqi")
351.                    .setEmail("rifqi@gmail.com")
352.                    .setAlamat("Bandung");
353.
354.            owners.put(Owner1.getIdOwner(), Owner1);
355.        }
356.        // selesai
```

```
357.
358.            //Vey
359.            public void dataPenyewa(){
360.                Scanner input = new Scanner(System.in);
361.                    System.out.println("=======SELAMAT DATANG DI
    MAMIKOS=======");
362.                System.out.println("Silahkan Masukan Data Anda:");
363.                System.out.print("Nama\t\t:");
364.                String nama = input.nextLine();
365.                System.out.print("ID\t\t:");
366.                String idPenyewa = input.nextLine();
367.                System.out.print("Jenis Kelamin\t:");
368.                String jenisKelamin = input.nextLine();
369.                System.out.print("Alamat\t\t:");
370.                String alamat = input.nextLine();
371.                System.out.print("Email\t\t:");
372.                String email = input.nextLine();
373.                System.out.print("No Telp\t\t:");
374.                String noTelp = input.nextLine();
375.
376.                Penyewa penyewaBaru = new Penyewa();
377.
378.                penyewaBaru.setIdPenyewa(idPenyewa);
379.                penyewaBaru.setNama(nama);
380.                penyewaBaru.setJenisKelamin(jenisKelamin);
381.                penyewaBaru.setAlamat(alamat);
382.                penyewaBaru.setEmail(email);
383.                penyewaBaru.setNoTelp(noTelp);
384.
385.                penyewas.put(idPenyewa, penyewaBaru);
386.                // penyewas.put(, penyewaBaru);
387.                lihatDaftarKamar();
388.            }
389.
390.            public void lihatDaftarKamar() {
391.                System.out.println();
392.                System.out.println("==== Daftar Kamar ====");
393.                for (Map.Entry<String, Kamar> Kamar : kamars.entrySet())
    {
394.                    String key = Kamar.getKey();
395.                    Kamar tKamar = Kamar.getValue();
396.                    if (tKamar.isIsAvailable()) {
397.                        System.out.println("No Kamar \t :" + key);
398.                                System.out.println("Jenis Kamar \t\t :" +
    tKamar.getJenisKamar());
399.                                        System.out.println("Harga \t\t :" +
    tKamar.getHargaKamar() + "/bulan");
400.                        System.out.println("");
401.                    }
402.
403.                }
404.            }
405.            //Selesai
406.
407.            //rehan
408.            public Kamar getKamar(String idKamar) {
409.                return kamars.get(idKamar);
```

```
410.            }
411.
412.            public Penyewa getPenyewa(String idPenyewa) {
413.                return penyewas.get(idPenyewa);
414.            }
415.
416.            public Owner getOwner(String idOwner) {
417.                return owners.get(idOwner);
418.            }
419.
420.            public void sewa() {
421.                    BufferedReader input = new BufferedReader(new
    InputStreamReader(System.in));
422.                ArrayList<DetailKamar> tDetailKamars = new ArrayList<>();
423.                int increment = 0;
424.                String lanjut = "";
425.                try {
426.                    do {
427.                        increment++;
428.                        System.out.println();
429.                        System.out.print("Silahkan masukkan No Kamar\t: ");
430.                        String NoKamar = input.readLine();
431.                        System.out.print("Berapa bulan?\t: ");
432.                        int jumlah = Integer.valueOf(input.readLine());
433.
434.                        DetailKamar dk = new DetailKamar();
435.                        Kamar tKamar = getKamar(NoKamar);
436.                        dk.setIdDetailKamar("dk" + increment);
437.                        dk.setKamar(tKamar);
438.                        dk.setHargaJual(tKamar.getHargaKamar());
439.                        dk.setSubTotal(tKamar.getHargaKamar() * jumlah);
440.                        tDetailKamars.add(dk);
441.
442.                    } while (lanjut.equalsIgnoreCase("Y"));
443.                    System.out.println("");
444.                    System.out.println("===Berikut Daftar Kamar Anda===");
445.                    tampilkanDaftarKamar(tDetailKamars);
446.                                        System.out.println("TOTAL:      " +
    hitungTotalKamar(tDetailKamars));
447.                                BufferedReader inputKonfirmasi = new
    BufferedReader(new InputStreamReader(System.in));
448.                    System.out.println("Konfirmasi Penyewaan (Y/N)?");
449.                    String konfirm = "n";
450.                    konfirm = inputKonfirmasi.readLine();
451.                    if (konfirm.equalsIgnoreCase("y")) {
452.                        Penyewaan penyewaan = new Penyewaan();
453.                        penyewaan.setDetailKamars(tDetailKamars);
454.                        penyewaan.setIdPenyewaan("P" + penyewaans.size()
    + 1);
455.                        penyewaan.setPenyewa(getPenyewa("P1"));
456.                        penyewaan.setOwner(getOwner("P1"));
457.                        penyewaan.setTanggal(LocalDate.now());
458.                        penyewaan.setTotal(hitungTotalKamar(tDetailKamars
    ));
```

```
459.                     penyewaans.add(penyewaan);
460.                 }
461.
462.             } catch (Exception e) {
463.                 System.out.println(e.getMessage());
464.             }
465.         }
466.         // han selesai
467.         // veyy
468.           public  void  tampilkanDaftarKamar(ArrayList<DetailKamar>
    dk) {
469.             for (DetailKamar detailKamar : dk) {
470.                                 System.out.println("No  Kamar  \t  :"  +
    detailKamar.getKamar().getIdKamar());
471.                                 System.out.println("Jenis  Kamar  \t  :"  +
    detailKamar.getKamar().getJenisKamar());
472.                                     System.out.println("Harga  \t  :"  +
    detailKamar.getHargaJual()+ "/bulan");
473.                 System.out.println("");
474.             }
475.         }
476.
477.         public int hitungTotalKamar(ArrayList<DetailKamar> dk) {
478.             int total = 0;
479.             for (DetailKamar detailKamar : dk) {
480.                 total += detailKamar.getSubTotal();
481.             }
482.             return total;
483.         }
484.         //selesai
485.
486.         //han
487.         public void tampilkanLaporanPenyewaan() {
488.         for (Penyewaan penyewaan : penyewaans) {
489.             System.err.println();
490.                 System.out.println("==================SEWA  KOS
    MAMIKOS==================");
491.                             System.out.println("No  Penyewaan  \t  :  "  +
    penyewaan.getIdPenyewaan());
492.                                 System.out.println("Tanggal  transaksi:  "  +
    penyewaan.getTanggal());
493.                                 System.out.println("Owner  \t  \t  :  "  +
    penyewaan.getOwner().getNama());
494.
495.             // Tampilkan nama penyewa dari objek penyewaan
496.                                 System.out.println("Penyewa  \t  :  "  +
    penyewaan.getPenyewa().getNama());
497.                                 System.out.println("Alamat  \t  \t  :  "  +
    penyewaan.getPenyewa().getAlamat());
498.                             System.out.println("Jenis  kelamin  \t  :  "  +
    penyewaan.getPenyewa().getJenisKelamin());
499.
500.                 System.out.println("============================
    ========================");
501.                                 ArrayList<DetailKamar>  detailKamars  =
    penyewaan.getDetailKamars();
502.                 for (DetailKamar dk : detailKamars) {
```

```
503.                                 System.out.println("No  Kamar  \t  :"  +
    dk.getKamar().getIdKamar());
504.                                 System.out.println("Jenis  Kamar  \t  :"  +
    dk.getKamar().getJenisKamar());
505.                     System.out.println("Harga \t\t :" + dk.getHargaJual()+
    "/bulan");
506.                 }
507.                  System.out.println("============================
    ========================");
508.                 System.out.println("Total \t \t : " + penyewaan.getTotal());
509.             }
510.         }
511.
512.         }
513.
514.
```

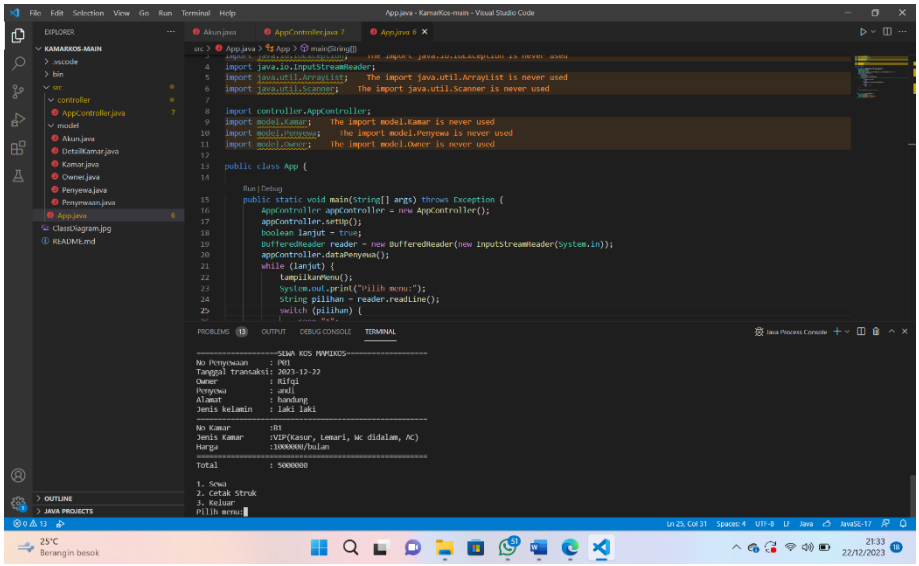
h. App.java

```
515.
516.         import java.io.BufferedReader;
517.         import java.io.IOException;
518.         import java.io.InputStreamReader;
519.         import java.util.ArrayList;
520.         import java.util.Scanner;
521.
522.         import controller.AppController;
523.         import model.Kamar;
524.         import model.Penyewa;
525.         import model.Owner;
526.
527.         public class App {
528.
529.             public static void main(String[] args) throws Exception {
530.                 AppController appController = new AppController();
531.                 appController.setUp();
532.                 boolean lanjut = true;
533.                     BufferedReader  reader  =  new  BufferedReader(new
    InputStreamReader(System.in));
534.                 appController.dataPenyewa();
535.                 while (lanjut) {
536.                     tampilkanMenu();
537.                     System.out.print("Pilih menu:");
538.                     String pilihan = reader.readLine();
539.                     switch (pilihan) {
540.                         case "1":
541.                             appController.sewa();
542.                             break;
543.                         case "2":
544.                             appController.tampilkanLaporanPenyewaan();
545.                             break;
546.                         default:
547.                             lanjut = false;
548.                             break;
```

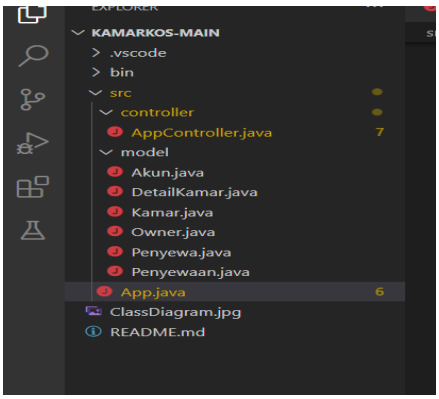
```
549.                 }
550.             }
551.
552.                 System.out.println("===SAMPAI JUMPA===");
553.
554.         }
555.
556.         public static void tampilkanMenu() {
557.             System.out.println("");
558.             System.out.println("1. Sewa");
559.             System.out.println("2. Cetak Struk");
560.             System.out.println("3. Keluar");
561.         }
562.
563.     }
564.
565.
```

3. Hasil

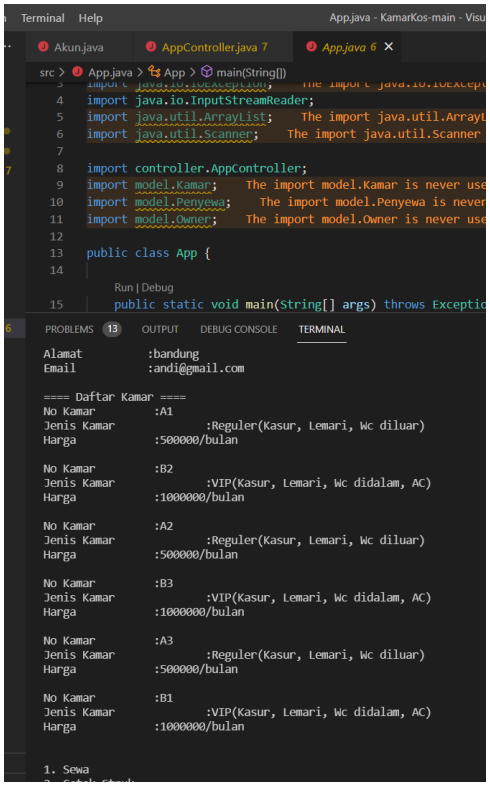
- Tampilan pemerograman



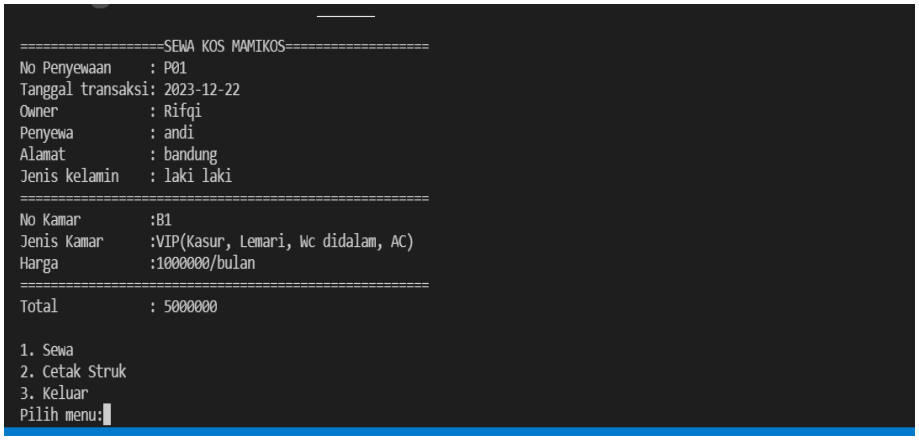
- Tampilan penggabungan class yang di eksekusi oleh AppController.java dan di tampilkan oleh App.java



- Tampilan kamar yang tersedia



- Tampilan setelah di running



Hasil dari pengembangan program pemesanan kamar kos menggunakan Java menunjukkan keberhasilan dalam menciptakan sebuah sistem yang memungkinkan pengguna untuk melakukan pemesanan secara efisien. Program ini berhasil mengintegrasikan fitur-fitur penting seperti manajemen ketersediaan kamar, proses reservasi, dan manajemen data pengguna dalam sebuah antarmuka yang mudah dipahami.

Data dari pengujian fungsionalitas menunjukkan bahwa program berjalan sesuai dengan yang diharapkan, dengan fitur-fitur seperti pengecekan ketersediaan kamar, proses reservasi, dan manajemen data pengguna berjalan dengan baik. Pengujian ketersediaan kamar juga menghasilkan informasi yang akurat dan real-time bagi pengguna.

Keberhasilan implementasi sistem pemesanan kamar kos ini terletak pada kemudahan pengguna dalam melakukan pemesanan, informasi yang cukup akurat dan tersedia secara real-time, serta kesesuaian fitur dengan kebutuhan pengguna. Sistem ini memberikan kontribusi signifikan dalam meningkatkan efisiensi dalam proses pemesanan kamar kos, mengurangi waktu yang dibutuhkan dalam reservasi, dan akan memberikan pengalaman yang memuaskan bagi pengguna.

Penggunaan teknologi dalam memfasilitasi pemesanan kamar kos memiliki implikasi besar terhadap efisiensi dan kepraktisan. Rekomendasi untuk pengembangan lebih lanjut adalah menambahkan fitur-fitur tambahan seperti notifikasi reservasi, integrasi dengan sistem pembayaran online, atau peningkatan keamanan data pengguna untuk meningkatkan kualitas dan keamanan system

4. KESIMPULAN

Pengembangan program sederhana pemesanan kamar kos menggunakan Java telah membuktikan bahwa penerapan teknologi dalam sistem pemesanan akomodasi dapat memberikan solusi efisien dan praktis bagi pengguna. Melalui implementasi yang teliti, program ini berhasil memfasilitasi proses pemesanan kamar kos dengan menghadirkan fitur-fitur penting seperti manajemen ketersediaan kamar, proses reservasi, dan manajemen data pengguna.

Kesuksesan program ini terletak pada antarmuka pengguna yang intuitif, memudahkan pengguna dalam mendapatkan informasi ketersediaan kamar dan melakukan reservasi dengan cepat dan efisien. Pengujian fungsionalitas juga mengkonfirmasi bahwa sistem beroperasi sesuai yang diharapkan, memberikan informasi yang akurat tentang ketersediaan kamar secara real-time.

Umpan balik dari pengguna menggambarkan kepuasan terhadap pengalaman menggunakan program ini. Ketersediaan sistem dalam memberikan layanan yang handal dan efisien telah memberikan kontribusi besar dalam meningkatkan efisiensi dalam proses pemesanan kamar kos.

Kesimpulan ini menekankan bahwa penggunaan teknologi dalam menyederhanakan proses pemesanan kamar kos memiliki dampak yang signifikan dalam memberikan kemudahan, kecepatan, dan kepraktisan bagi pengguna. Rekomendasi untuk pengembangan lebih lanjut adalah mempertimbangkan integrasi dengan sistem pembayaran online, notifikasi reservasi, dan peningkatan keamanan data pengguna untuk meningkatkan kualitas dan keamanan sistem secara keseluruhan.

Dengan demikian, program ini berhasil menunjukkan potensi dalam meningkatkan efisiensi dan pengalaman pengguna dalam proses pemesanan kamar kos, sementara menjadi landasan untuk pengembangan sistem serupa dengan tingkat fungsionalitas yang lebih luas dan efisien di masa mendatang.

REFERENSI

[1] W3school. *Java OOP.* Diakses Pada 22 Desamber 2023 dari https://www.w3schools.com/java/java_oop.asp

[2] Programiz. *Apa aitu pemrograman Berorietasi objek? Panduan pemula.* Diakses pada 22 Desember 2023 dari https://programiz.pro/resources/what-is-object-oriented-programming/

[3] Dicoding. *Apa aitu OOP pada java beserta contohnya.* Diakses pada 22 Desember 2023 dari https://www.dicoding.com/blog/apa-itu-oop-pada-java-beserta-contohnya/

[4] Abdul M, Alvin Y, Renny S. (2020) *Software Development Website Inventaris Pada Pusat Perbelanjaan XYZ.* JURIKOM (Jurnal Riset Komputer), Vol. 7 No. 1. Hal 28-33

[5] Simarmata, J. 2010. Rekayasa Perangkat Lunak. Yogyakarta: C.V Andi Offset.