

# **LAPORAN PEMBUATAN APLIKASI TIKET PESAWAT**

Diajukan untuk memenuhi Ujian akhir semester  
Mata kuliah Pemrograman  
Dosen pengampu : Kamarudin, M.Kom., S.Kom.



Nama : Frand Odi Anggoro

NIM :21.11.4443

**PRORAM STUDI INFORMATIKA  
UNIVERSITAS AMIKOM YOGYAKARTA  
TAHUN 2024**



## Laporan pengerjaan aplikasi tiket pesawat

Aplikasi ini merupakan sistem manajemen tiket berbasis konsol yang ditulis dalam C#<sup>1</sup>. yang memungkinkan user untuk membuat, melihat dan mengelola tiket (kelas bisnis dan ekonomi) serta menyimpan informasi tiket dalam database MySQL.

### Fungsi yang tersedia di dalam aplikasi:

1. Pembuatan tiket  
Ada 2 jenis tiket yang tersedia, yaitu :
  - Tiket Kelas Bisnis: User dapat membuat tiket kelas bisnis dengan memasukkan nama dan alamatnya.
  - Tiket Kelas Ekonomi: User dapat membuat tiket kelas ekonomi dengan pilihan diskon dengan memasukkan nama, alamat dan konfirmasi apakah ingin menerima diskon.
2. Lihat detail tiket  
User dapat melihat semua tiket yang telah di buat, termasuk tiket bisnis dan ekonomi, dengan rincian informasinya.
3. Simpan data ke basis data  
Aplikasi menyimpan data tiket ke database MySQL<sup>2</sup>. Dengan regulasi tiket yang sudah ada di database tidak disimpan lagi untuk menghindari duplikasi.
4. Lihat data yang disimpan  
User dapat melihat seluruh data tiket pesawat yang tersimpan di database, termasuk tiket pesawat bisnis dan tiket pesawat ekonomi.
5. Hapus semua data dalam database  
User dapat menghapus semua data tiket dari database setelah konfirmasi.
6. Keluar dari aplikasi  
User dapat keluar dari aplikasi kapan saja dengan memilih opsi yang sesuai.

### Alur Program

- Tampilan Menu Utama:  
Aplikasi menampilkan menu utama dengan pilihan untuk membuat tiket bisnis, ekonomi, melihat rincian tiket, menyimpan ke database, melihat data yang tersimpan, menghapus semua data, dan keluar.
- Input User:  
User memilih opsi yang diinginkan. Input yang tidak valid akan meminta user untuk memasukkan pilihan yang benar.
- Operasi Sesuai Pilihan User:

---

<sup>1</sup> “C# OOP (Object-Oriented Programming).”

<sup>2</sup> Kamarudin, “Mengakses Database MySQL Menggunakan C# Bagian 1”; Kamarudin, “Mengakses Database MySQL Menggunakan C# Bagian 2.”

Berdasarkan pilihan, aplikasi akan melakukan operasi yang sesuai seperti membuat tiket, menampilkan rincian, menyimpan ke database, atau menghapus data.

### ● Basis Data

Aplikasi ini menggunakan MySQL<sup>3</sup> untuk menyimpan data tiket. Ada dua tabel dalam database: tiket\_bisnis dan tiket\_ekonomi, masing-masing menyimpan data tiket sesuai dengan jenisnya.

### Source Kode :

### ● Basis Data

Aplikasi ini menggunakan database MySql, Berikut sintaksnya

```
-- phpMyAdmin SQL Dump
-- version 5.2.1
-- https://www.phpmyadmin.net/
--
-- Host: 127.0.0.1
-- Waktu pembuatan: 25 Jul 2024 pada 18.31
-- Versi server: 10.4.32-MariaDB
-- Versi PHP: 8.2.12

SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
START TRANSACTION;
SET time_zone = "+00:00";

/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8mb4 */;

--
-- Database: `dbticket`
--

--
-- Struktur dari tabel `tiket_bisnis`
--

CREATE TABLE `tiket_bisnis` (
  `id` varchar(15) NOT NULL,
  `nama` varchar(30) NOT NULL,
  `harga` int(7) NOT NULL,
  `alamat` varchar(50) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;

--
-- Struktur dari tabel `tiket_ekonomi`
--

CREATE TABLE `tiket_ekonomi` (
  `nama` varchar(30) NOT NULL,
```

<sup>3</sup> Rached, "Connect C# to MySQL."

```

`alamat` varchar(50) NOT NULL,
`harga` int(7) NOT NULL,
`id` varchar(15) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;

--
-- Indexes for dumped tables
--

--
-- Indeks untuk tabel `tiket_bisnis`
--
ALTER TABLE `tiket_bisnis`
  ADD PRIMARY KEY (`id`);

--
-- Indeks untuk tabel `tiket_ekonomi`
--
ALTER TABLE `tiket_ekonomi`
  ADD PRIMARY KEY (`id`);
COMMIT;

/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;
/*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;
/*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;

```

## ● Class pendukung

### ■ Class Tiket

Sebagai kelas yang menangani tiket yang menyediakan method2 yang diperlukan oleh kelas kelas turunannya, yaitu child class Tiket Ekonomi dan Tiket Bisnis

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace TicketRev1
{
    public abstract class Tiket : ITicket
    {
        private readonly string name;
        private string alamat;
        protected string id;

        public string Name { get { return name; } }
        public int Price { get; protected set; }
        public string Alamat { get; set; }
        public string Id { get { return id; } set { id = value; } }

        public Tiket(string name, string alamat)
        {
            this.name = name;
            this.alamat = alamat;
            id = Convert.ToString(Utils.GenerateRandomId()); //dapatkan id
            //dari randomize generator

```

```

    }

    //deklarasi fungsi yang dapat di override
    public virtual void PrintInfo()
    {
        Console.WriteLine("\n");
        Console.WriteLine("----- Data Tiket -----");
        Console.WriteLine($"{Name}\t: {Name}\nAlamat\t: {Alamat}");
        Console.WriteLine("-----");
    }

    //deklarasi fungsi abstract
    public abstract int Discount();
}
}

```

- Class ITicket : sebagai kelas interface yang berkontrak dengan kelas parent Tiket, berisi method method yang harus dilengkapi oleh kelas tiket

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

//interface iticket berfiliasi dengan parentclass Tiket
namespace TicketRev1
{
    internal interface ITicket
    {
        void PrintInfo();
    }
}

```

- Class Utils : sebagai class yang digunakan untuk membuat method method tambahan pada program

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading;
using System.Threading.Tasks;

namespace TicketRev1
{
    public class Utils
    {
        //buat ada titik2 berjalan
        public static void PrintDots(int CountOfDots, string typeMsg)
        {
            if (typeMsg == "save")

```

```

        {
            Console.WriteLine("Saving Data, Please Wait ");
        }
        else if (typeMsg == "load")
        {
            Console.WriteLine("Loading Data, Please Wait ");
        }
        else if (typeMsg == "create")
        {
            Console.WriteLine("Creating Data, Please Wait ");
        }
        else if (typeMsg == null)
        {
            Console.Write("");
        }
        else if (typeMsg == "read")
        {
            Console.Write("Reading Data, Please Wait");
        }

        for (int i = 0; i < CountOfDots; i++)
        {
            Console.Write(". ");
            Thread.Sleep(100); //delay 300 ms per titik
        }
        Console.Clear();
    }

    //buat readonly id random dan generate
    private static readonly Random Random = new Random();
    public static int GenerateRandomId()
    {
        return Random.Next(1000000, 9999999); // ID acak antara 100000
dan 999999
    }

    //cek apakah bisa mendapatkan diskon
    public static bool IsDiscountable(char input)
    {
        if (input == 'y')
        {
            return true;
        }
        else
        {
            return false;
        }
    }
}

```

- Class TiketEkonomi: sebagai child class dari class Tiket yang bertanggung jawab untuk Tiket berjenis Ekonomi

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace TicketRev1
{
    internal class TiketEkonomi : Tiket
    {
        private const string jenisTiket = "EKONOMI"; //deklarasi jenis
        tiket
        private readonly int priceEko = 15000;
        private readonly bool discount;

        private readonly string IdEko;

        public TiketEkonomi(string name, string alamat , bool discount) :
        base(name,alamat)
        {
            IdEko = "EKO-" + Id;
            base.Price = priceEko; //set price di class parent dengan
            price di class TiketEko
            this.discount = discount;
            this.Alatamat = alamat;

        }
        //panggil method wajib dari abstract discount
        //karena ekonomi maka bisa pakai diskon
        public override int Discount()
        {
            int disc = priceEko * 90 / 100; // diskon 10%
            return disc;
        }

        public override void PrintInfo()
        {
            Console.ForegroundColor = ConsoleColor.Green;
            base.PrintInfo();//cetak informasi umum dari parent Tiket

            //cetak jenis tiket : Ekonomi

            Console.WriteLine("Id\t: {0}", IdEko);
            Console.WriteLine("Jenis\t: {0}", jenisTiket);

            if (discount)
            {
                Console.WriteLine("Harga\t: {0}", Discount());
            }
        }
    }
}
```



```

        else
        {
            Console.WriteLine("Harga\t: {0}", priceEko);
        }
        Console.WriteLine("=====");
        Console.ResetColor();
    }
    public void PrintPrice()
    {
        Console.ForegroundColor = ConsoleColor.Green;
        Console.WriteLine("Harga Tiket Ekonomi: {0}", priceEko);
        Console.ResetColor();
    }
}
}

```

- Class Tiket Bisnis : sebagai child class dari class Tiket yang bertanggung jawab untuk Tiket berjenis Bisnis

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace TicketRev1
{
    internal class TiketBisnis : Tiket
    {
        private const string jenisTiket = "BISNIS"; //deklarasi jenis
        tiket
        private const int priceBis = 20000;

        private readonly string IdBisnis;

        public TiketBisnis(string name, string alamat) : base(name,
        alamat)
        {
            IdBisnis = "BIS-" + Id;
            this.Alatmat = alamat;
            base.Price = priceBis; //set price di class parent dengan
            price di class TiketBisnis

        }
        //panggil method wajib dari abstract discount
        //karena bisnis maka tidak menerapkan diskon
        public override int Discount()
        {
            //karena tiket bisnis tidak menerapkan diskon
            throw new NotImplementedException();
        }
        public void PrintPrice()
        {
            Console.ForegroundColor = ConsoleColor.Yellow;

```

```

        Console.WriteLine("Harga Tiket Bisnis: {0}", priceBis);
        Console.ResetColor();
    }

    public override void PrintInfo()
    {
        Console.ForegroundColor = ConsoleColor.Yellow;
        base.PrintInfo();//cetak informasi umum dari parent Tiket

        //cetak jenis tiket : BISNIS
        Console.WriteLine("Id\t: {0}", IdBisnis);
        Console.WriteLine("Jenis\t: {0}", jenisTiket);
        Console.WriteLine("Harga\t: {0}", priceBis);

        Console.WriteLine("=====");
        Console.ResetColor();
    }
}

```

- **Class Program:** sebagai Main program sistem, sebagai tempat utama semua program akan di eksekusi

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Runtime.InteropServices;
using System.Text;
using System.Threading.Tasks;
using MySql.Data.MySqlClient;
using Mysqlx;
using Mysqlx.Crud;

namespace TicketRev1
{
    internal class Program
    {
        public static void Main()
        {
            // Deklarasi collection untuk menampung tiket
            List<TiketBisnis> TicBisnis = new List<TiketBisnis>();
            List<TiketEkonomi> TicEkonomi = new List<TiketEkonomi>();

            // deklarasi database untuk penyimpanan data
            string connStr =
"server=localhost;user=root;database=dbTicket;password=";
            MySqlConnection conn = new MySqlConnection(connStr);

            while (true)
            {
                Console.Clear(); // Bersihkan layar sebelum menampilkan
                Console.WriteLine("----- ++ Aplikasi Tiket Pesawat ++ --
                menu
                ----");

                Console.WriteLine("=====");
            }
        }
    }
}

```

```

//print harga terkini

TiketBisnis Tiket1 = new TiketBisnis(null,null);
Tiket1.PrintPrice();
TiketEkonomi Tiket2 = new TiketEkonomi(null,null,false);
Tiket2.PrintPrice();

Console.WriteLine("-----
----");

Console.WriteLine("Pilih Ingin Membuat Jenis Tiket Apa :
");

Console.WriteLine("1. Bisnis \n2. Ekonomi \n3. Lihat
Rincian Tiket \n4. Simpan Ke database\n5. Lihat Data yang tersimpan\n6.
Hapus Semua data di database\n\n99. Keluar");
Console.WriteLine("\n-----
-----");

Console.Write("Pilihan Anda (1 .. 5) : ");
//validasi hanya inputan 1 sd 4 yang diizinkan
bool validInput = int.TryParse(Console.ReadLine(), out int
pil);

if (!validInput || pil < 1 || pil > 6 && pil!= 99 )
{
    Console.WriteLine("Masukkan inputan yang benar antara
1 hingga 6.");
    Console.ReadKey();
    continue; // Kembali ke awal loop jika input tidak
valid
}
else
{
    switch (pil)
    {
        case 1:
            Utils.PrintDots(5, "load");
            Console.WriteLine("Tiket
Bisnis\n=====
\n\n+ Masukkan Data Anda \n");
            Console.Write("Nama : ");
            string valName = Console.ReadLine();

            Console.Write("Alamat : ");
            string valAlamat = Console.ReadLine();

            // Masukkan data ke constructor TiketBisnis
            TiketBisnis TiketBisnis = new
TiketBisnis(valName, valAlamat);

            // Creating data animation
            Utils.PrintDots(5, "create");

            // Masukkan ke list object bertipe
'TiketBisnis'
            TicBisnis.Add(TiketBisnis);

            // Tampilkan informasi dari tiket bisnis
            Console.WriteLine("Data Sukses terInput
..\nBerikut data Tiket Bisnis Anda : \n");
            TiketBisnis.PrintInfo();
            Console.ReadKey();

            break;

```

```

case 2:

    Utils.PrintDots(5, "load");
    Console.WriteLine("Tiket
Ekonomi\n===== \n\n+ Masukkan Data Anda \n");
    Console.WriteLine("Nama : ");
    string valNameEko = Console.ReadLine();

    Console.WriteLine("Alamat : ");
    string valAlamatEko = Console.ReadLine();

    //konfirmasi apakah ingin di diskon?
    Console.WriteLine("Ingin Didiskon ? (y/n) :");
    char phil =
Convert.ToChar(Console.ReadLine());

    // Masukkan data ke constructor TiketEkonomi
    dengan diskon/tidak
    TiketEkonomi TiketEkonomi = new
TiketEkonomi(valNameEko, valAlamatEko, Utils.IsDiscountable(phil)); ;

    // Creating data animation
    Utils.PrintDots(5, "create");

    // Masukkan ke list object bertipe
'TiketEkonomi'
    TicEkonomi.Add(TiketEkonomi);

    // Tampilkan informasi dari TiketEkonomi
    Console.WriteLine("Data Sukses terInput
..\nBerikut data Tiket Ekonomi Anda : \n");
    TiketEkonomi.PrintInfo();
    Console.ReadKey();
    break;
case 3:
    Utils.PrintDots(5, "load");
    Console.Clear();

    if (TicBisnis.Count > 0)
    {
        Console.WriteLine("Berikut Rincian Tiket
Yang telah terinput");
        Console.WriteLine(" === Tiket Bisnis
===");
        foreach (var tiket in TicBisnis)
        {
            tiket.PrintInfo();
        }
    }

    Console.WriteLine("\n----- +++++ -----
\n");

    if (TicEkonomi.Count > 0)
    {
        Console.WriteLine(" === Tiket Ekonomi ===
");
        foreach (var tiket in TicEkonomi)
        {
            tiket.PrintInfo();
        }
    }
    else

```

```

        {
            Console.WriteLine("Tidak ada tiket yang
terdaftar.");
        }
        //kembali ke menu
        Console.WriteLine("Enter untuk kembali ke menu
utama ..");

        Console.ReadKey();
        break;
    case 4:
        Console.WriteLine("Simpan ke Database");
        Utils.PrintDots(6, "load");

        try
        {
            conn.Open();
            Console.WriteLine("Terhubung ke
database");

            if (TicEkonomi.Count == 0 &&
TicBisnis.Count == 0)
            {
                Console.WriteLine("Tidak ada tiket
untuk disimpan.");
            }
            else
            {
                Utils.PrintDots(5, "save");
                foreach (var tiket in TicEkonomi)
                {
                    string checkQuery = "select
count(*) from tiket_ekonomi where id = @id";
                    MySqlCommand checkCmd = new
MySqlCommand(checkQuery, conn);
                    checkCmd.Parameters.AddWithValue("@id", "Eko-" + tiket.Id);

                    int count =
Convert.ToInt32(checkCmd.ExecuteScalar());

                    if (count == 0)
                    {
                        string sqlInsert = "insert
tiket_ekonomi (id, nama, harga, alamat) values (@id, @nama, @harga,
@alamat)";
                        MySqlCommand cmd = new
MySqlCommand(sqlInsert, conn);

                        cmd.Parameters.AddWithValue("@id", "Eko-" + tiket.Id);
                        cmd.Parameters.AddWithValue("@nama", tiket.Name);
                        cmd.Parameters.AddWithValue("@harga", tiket.Price);
                        cmd.Parameters.AddWithValue("@alamat", tiket.Alat);

                        int rowCount =
cmd.ExecuteNonQuery();

                        if (rowCount > 0)
                        {

```

```

Ekonomi, {0} tersimpan", tiket.Name);
        Console.WriteLine("Tiket
    }
    else
    {
        Console.WriteLine("Tidak
    }
    }
    else
    {
        Console.WriteLine("Tiket
Ekonomi dengan ID {0} dan Nama {1} sudah ada.", "Eko-" + tiket.Id,
    tiket.Name);
    }
    }

    foreach (var tiket in TicBisnis)
    {
        string checkQuery = "select
count(*) from tiket_bisnis where id = @id";
        MySqlCommand checkCmd = new
        MySqlCommand(checkQuery, conn);
        checkCmd.Parameters.AddWithValue("@id", "Bis-" + tiket.Id);

        int count =
        Convert.ToInt32(checkCmd.ExecuteScalar());

        if (count == 0)
        {
            string sqlInsert = "insert
tiket_bisnis (id, nama, harga, alamat) values (@id, @nama, @harga,
@alamat)";
            MySqlCommand cmd = new
            MySqlCommand(sqlInsert, conn);

            cmd.Parameters.AddWithValue("@id", "Bis-" + tiket.Id);
            cmd.Parameters.AddWithValue("@nama", tiket.Name);
            cmd.Parameters.AddWithValue("@harga", tiket.Price);
            cmd.Parameters.AddWithValue("@alamat", tiket.Alat);

            int rowCount =
            cmd.ExecuteNonQuery();

            if (rowCount > 0)
            {
                Console.WriteLine("Tiket
Bisnis, {0} tersimpan", tiket.Name);
            }
            else
            {
                Console.WriteLine("Tidak
            }
            }
            else
            {
                Console.WriteLine("Tiket

```

```

Bisnis dengan ID {0} dan Nama {1} sudah ada.", "Bis-" + tiket.Id,
tiket.Name);
        }
    }
}
catch (Exception err)
{
    Console.WriteLine(err.ToString());
}
finally
{
    conn.Close();
}

Console.ReadKey();
break;

case 5:
    Utils.PrintDots(5, "read");
    Console.WriteLine("List Data yg Tersimpan
di database");

    conn.Open();//buka akses ke database

    try
    {
        //mulai pembacaan tiket bisnis
        //query untuk membaca data
        string sqlReadBis = "select * from
tiket_bisnis";

        //deklarasi method comaandSql
        MySqlCommand cmdBis = new
        MySqlCommand(sqlReadBis, conn);

        //deklarasi method readerSql
        MySqlDataReader readerBis =
        cmdBis.ExecuteReader();

        Console.WriteLine("\nData          Tiket
Bisnis\n");

        int noBis = 1; // untuk nomer secara urut
        while(readerBis.Read())
        {
            //tampilkan sebaris demi baris
            Console.WriteLine(noBis + ".   Id
tiket\t:      {0},\tNama\t:      {1},\tAlamat\t:      {2},\tHarga\t:      {3}",
readerBis["id"],      readerBis["nama"],      readerBis["alamat"],      readerBis
["harga"]);

            noBis++;
        }
        readerBis.Close();

        //Mulai pembacaan dari db tiketekonomi
        string sqlReadEko = "select * from
tiket_ekonomi";

        MySqlCommand cmdEko = new
        MySqlCommand(sqlReadEko, conn);
        MySqlDataReader readerEko =
        cmdEko.ExecuteReader();

```

```

Ekonomi\n");
Console.WriteLine("\nData
Tiket

int noEko = 1;
while (readerEko.Read())
{
    Console.WriteLine(noEko + ". Id
tiket\t: {0},\tNama\t: {1},\tAlamat\t: {2},\tHarga\t: {3}", readerEko
["id"], readerEko["nama"], readerEko["alamat"], readerEko["harga"]);
    noEko++;
}
readerEko.Close();
}
catch (Exception err){
    Console.WriteLine(err.ToString());
    Console.WriteLine("\ntekan
sembarang
tombol untuk kembali ke menu utama");

    Console.ReadKey();
}
finally
{
    conn.Close();
    Console.WriteLine("\ntekan
sembarang
tombol untuk kembali ke menu utama");
    Console.ReadKey();
}
break ;
case 6:
    Utils.PrintDots(5, "load");
    Console.WriteLine("Hapus
Semua
data
pada
database\n\n Apakah Anda Yakin? (y/n) : ");
    string pill= Console.ReadLine();

    if ( pill == "y")
    {
        try
        {
            conn.Open();
            //buat query penghapusan
            string sqlDelBis = "delete from
tiket_bisnis";
            string sqlDelEko = "delete from
tiket_ekonomi";

            //Panggil method mysql
            MySqlCommand cmdDelBis= new MySqlCommand(
sqlDelBis, conn);
            MySqlCommand cmdDelEko = new
MySqlCommand(sqlDelEko, conn);

            //jalankan perintah dan ambil jumlah baris
yang terhapus (optionl)
            int
countDelBis=cmdDelBis.ExecuteNonQuery();
            int
countDelEko=cmdDelEko.ExecuteNonQuery();

            //tampilkan informasi penghapusan
            if(countDelBis > 0 || countDelEko > 0)
{

```





```

C:\Users\user\Documents\ki x + v
----- ++ Aplikasi Tiket Pesawat ++ -----
=====
Harga Tiket Bisnis: 20000
Harga Tiket Ekonomi: 15000
=====
Pilih Ingin Membuat Jenis Tiket Apa :
1. Bisnis
2. Ekonomi
3. Lihat Rincian Tiket
4. Simpan Ke database
5. Lihat Data yang tersimpan
6. Hapus Semua data di database

99. Keluar
=====
Pilihan Anda (1 .. 5) : |

```

Tersedia beberapa menu yang bisa di pakai oleh user

● Tampilan saat user memilih 1, membuat tiket bisnis

```

C:\Users\user\Documents\ki x + v
Tiket Bisnis
=====
+ Masukkan Data Anda
Nama : frand odi
Alamat : Rawajitu

```

Berisi field yang bisa di isi oleh user sesuai permintaan field, yang kemudian akan tersimpan di memori sistem

```

C:\Users\user\Documents\ki x + v
Data Sukses terInput ..
Berikut data Tiket Bisnis Anda :

----- Data Tiket -----
Name   : frand odi
Alamat : Rawajitu
-----
Id      : BIS-1364680
Jenis   : BISNIS
Harga  : 20000
=====

```

Setelah selesai maka akan tampil hasil penginputan yang telah di terima, berikut id yang digenerate secara otomatis oleh sistem serta harga tiket.

- Tampilan saat memilih 2, Ekonomi

```
C:\Users\user\Documents\k1 x + v
Tiket Ekonomi
=====
+ Masukkan Data Anda
Nama : odi
Alamat : simbarwaringin
Ingin Didiskon ? (y/n) :y
```

Sama seperti ketika memilih Bisnis, hanya bedanya pada tiket ekonomi ada pilihan apakah ingin mendapatkan diskon atau tidak, adapun besaran diskon nya sebesar 10%.

```
C:\Users\user\Documents\k1 x + v
Data Sukses terInput ..
Berikut data Tiket Ekonomi Anda :

----- Data Tiket -----
Name      : odi
Alamat    : simbarwaringin
-----
Id        : EKO-5018712
Jenis     : EKONOMI
Harga     : 13500
=====
```

Sama seperti ketika memilih bisnis. Bedanya id nya menjadi EKO-xxxxxxx,

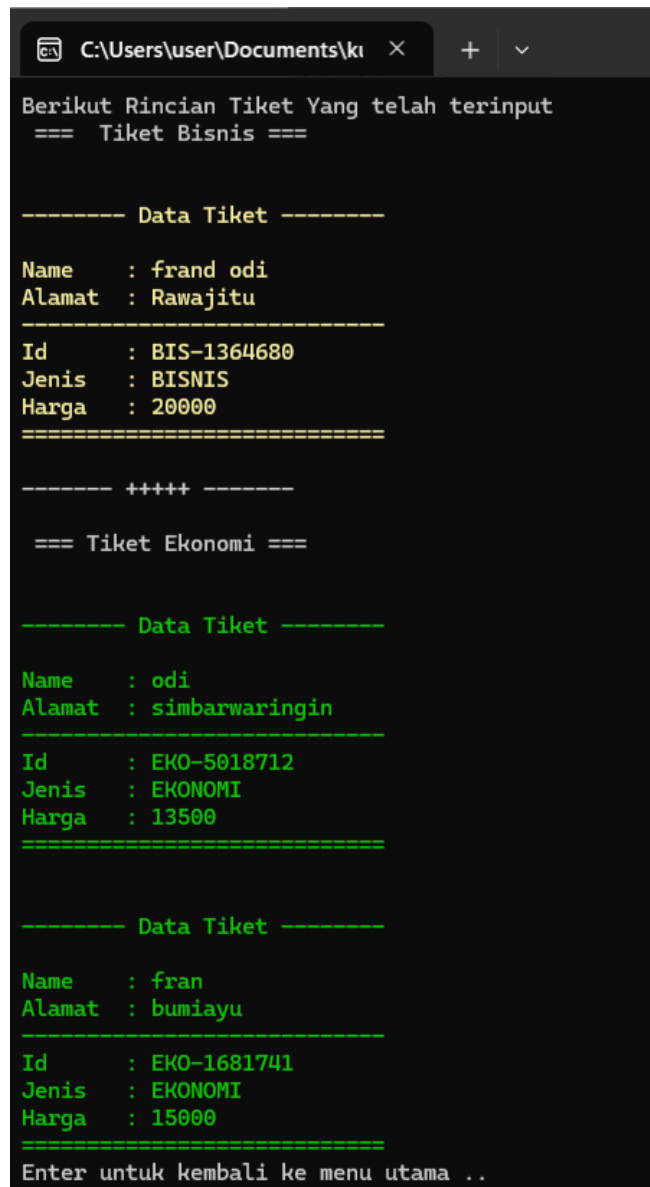
Namun berbeda ketika tidak ingin mengambil diskon maka harga tiketnya akan normal

```
C:\Users\user\Documents\k1 x + v
Tiket Ekonomi
=====
+ Masukkan Data Anda
Nama : fran
Alamat : bumiayu
Ingin Didiskon ? (y/n) :n
```

```
C:\Users\user\Documents\k1 x + v
Data Sukses terInput ..
Berikut data Tiket Ekonomi Anda :

----- Data Tiket -----
Name      : fran
Alamat    : bumiayu
-----
Id        : EKO-1681741
Jenis     : EKONOMI
Harga     : 15000
=====
```

- Tampilan saat memilih 3, Lihat rincian tiket  
Terlihat rincian tiket yang sebelumnya sudah terinput



```

C:\Users\user\Documents\k1 x + v
Berikut Rincian Tiket Yang telah terinput
=== Tiket Bisnis ===

----- Data Tiket -----
Name      : frand odi
Alamat    : Rawajitu
-----
Id        : BIS-1364680
Jenis     : BISNIS
Harga     : 20000
=====

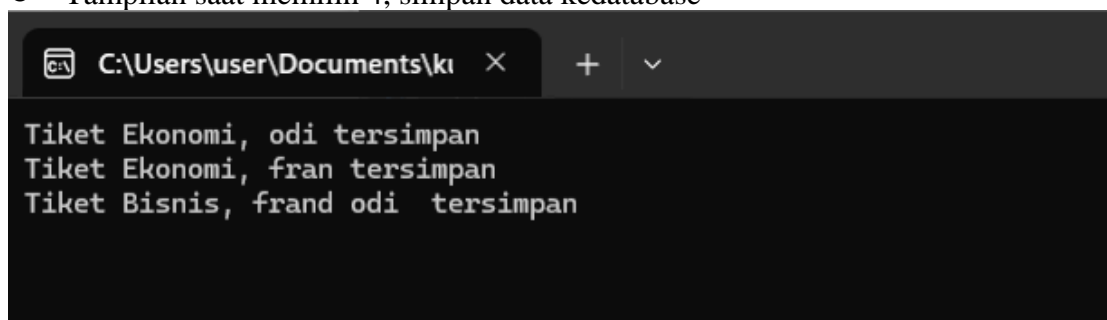
----- +++++ -----

=== Tiket Ekonomi ===

----- Data Tiket -----
Name      : odi
Alamat    : simbarwaringin
-----
Id        : EKO-5018712
Jenis     : EKONOMI
Harga     : 13500
=====

----- Data Tiket -----
Name      : fran
Alamat    : bumiayu
-----
Id        : EKO-1681741
Jenis     : EKONOMI
Harga     : 15000
=====
Enter untuk kembali ke menu utama ..
  
```

- Tampilan saat memilih 4, simpan data ke database



```

C:\Users\user\Documents\k1 x + v
Tiket Ekonomi, odi tersimpan
Tiket Ekonomi, fran tersimpan
Tiket Bisnis, frand odi tersimpan
  
```

Terlihat bahwa akan tersimpan di database, dibuktikan pada row MySql yang bertambah sebagaimana terlihat di phpMyAdmin , saya mengintegrasikan aplikasi ini dengan database bernama dbTiket, berikut hasilnya

- tabel tiket\_ekonomi: terlihat ada 2 baris yang sesuai dengan value yang terinput

Server: 127.0.0.1 / Database: dbticket / Tabel: tiket\_ekonomi

SELECT \* FROM `tiket\_ekonomi`

Tampilkan semua Jumlah baris: 25 Saring baris: Cari di tabel ini Sort by ke

Extra options

	nama	alamat	harga	id
<input type="checkbox"/> Ubah <input type="checkbox"/> Salin <input type="checkbox"/> Hapus	fran	bumiayu	15000	Eko-1681741
<input type="checkbox"/> Ubah <input type="checkbox"/> Salin <input type="checkbox"/> Hapus	frand	lampung	15000	Eko-2107264
<input type="checkbox"/> Ubah <input type="checkbox"/> Salin <input type="checkbox"/> Hapus	ww	qww	15000	Eko-3864128
<input type="checkbox"/> Ubah <input type="checkbox"/> Salin <input type="checkbox"/> Hapus	odi	simbarwaringin	15000	Eko-5018712
<input type="checkbox"/> Ubah <input type="checkbox"/> Salin <input type="checkbox"/> Hapus	nanda	sumatra	15000	Eko-6310776
<input type="checkbox"/> Ubah <input type="checkbox"/> Salin <input type="checkbox"/> Hapus	fran	lampung	15000	Eko-8288654
<input type="checkbox"/> Ubah <input type="checkbox"/> Salin <input type="checkbox"/> Hapus	salsa	surabaya	15000	Eko-9010822

Pilih Semua Dengan pilihan: Ubah Salin Hapus Ekspor

Tampilkan semua Jumlah baris: 25 Saring baris: Cari di tabel ini Sort by ke

Operasi hasil kueri

Konsol Salin ke clipboard Ekspor Tampilkan bagan Buat tampilan

- tabel tiket\_bisnis: terlihat ada 1 baris yang sesuai dengan value yang terinput

Server: 127.0.0.1 / Database: dbticket / Tabel: tiket\_bisnis

SELECT \* FROM `tiket\_bisnis`

Tampilkan semua Jumlah baris: 25 Saring baris: Cari di tabel ini Sort by ke

Extra options

	id	nama	harga	alamat
<input type="checkbox"/> Ubah <input type="checkbox"/> Salin <input type="checkbox"/> Hapus	Bis-1364680	frand odi	20000	Rawajitu
<input type="checkbox"/> Ubah <input type="checkbox"/> Salin <input type="checkbox"/> Hapus	Bis-2999641	frna	20000	de
<input type="checkbox"/> Ubah <input type="checkbox"/> Salin <input type="checkbox"/> Hapus	Bis-7084090	frand	20000	jambi
<input type="checkbox"/> Ubah <input type="checkbox"/> Salin <input type="checkbox"/> Hapus	Bis-7221423	fata#	20000	

Pilih Semua Dengan pilihan: Ubah Salin Hapus Ekspor

Tampilkan semua Jumlah baris: 25 Saring baris: Cari di tabel ini Sort by ke

Operasi hasil kueri

Cetak Salin ke clipboard Ekspor Tampilkan bagan Buat tampilan

Markahi kueri SQL ini

- Tampilan saat user memilih 5, Lihat data Tersimpan

```

C:\Users\user\Documents\ki x + v
List Data yg Tersimpan di database

Data Tiket Bisnis
1. Id tiket      : Bis-1364680, Nama      : frand odi , Alamat : Rawajitu, Harga : 20000
2. Id tiket      : Bis-2999641, Nama      : frna, Alamat : de, Harga : 20000
3. Id tiket      : Bis-7084090, Nama      : frand, Alamat : jambi, Harga : 20000
4. Id tiket      : Bis-7221423, Nama      : fata#, Alamat : , Harga : 20000

Data Tiket Ekonomi
1. Id tiket      : Eko-1681741, Nama      : fran, Alamat : bumiayu, Harga : 15000
2. Id tiket      : Eko-2107264, Nama      : frand, Alamat : lampung, Harga : 15000
3. Id tiket      : Eko-3864128, Nama      : ww, Alamat : qww, Harga : 15000
4. Id tiket      : Eko-5018712, Nama      : odi, Alamat : simbarwaringin, Harga : 15000
5. Id tiket      : Eko-6310776, Nama      : nanda, Alamat : sumatra, Harga : 15000
6. Id tiket      : Eko-8288654, Nama      : fran, Alamat : lampung, Harga : 15000
7. Id tiket      : Eko-9010822, Nama      : salsa, Alamat : surabaya, Harga : 15000

tekan sembarang tombol untuk kembali ke menu utama

```

Terlihat rincian tiket yang telah terinput oleh sistem ke database mysql, hal ini tentu membuat data menjadi permanen . tidak lagi temporari seperti halnya program yang tidak memiliki sistem database

- Tampilan saat user memilih 6, hapus semua data tersimpan (menghapus semua data di database)

```

C:\Users\user\Documents\ki x + v
Hapus Semua data pada database

Apakah Anda Yakin? (y/n) : |

```

Sebelumnya dilakukan konfirmasi terlebih dahulu apakah ingin melakukannya atau tidak, jika iya maka data akan terhapus seluruhnya, juga akan di tampilkan jumlah keseluruhan data yang terhapus

```

C:\Users\user\Documents\ki x + v
Hapus Semua data pada database

Apakah Anda Yakin? (y/n) : y
Data Sukses terhapus !!
Jumlah terhapus
- Tiket Bisnis      : 2
- Tiket Ekonomi     : 3
|

```

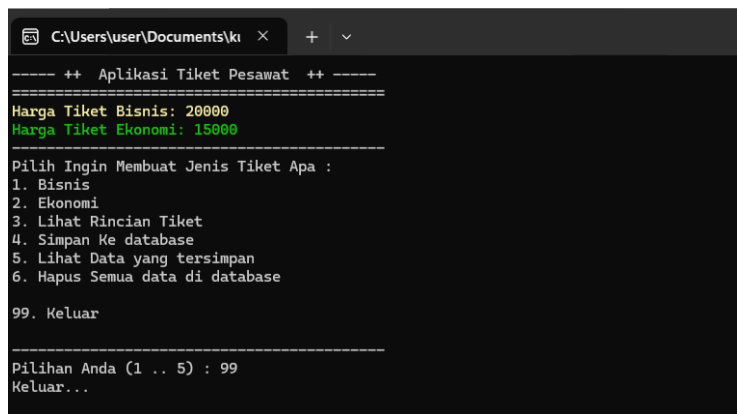
Dibuktikan juga di MySQL yang datanya menjadi kosong  
- tabel tiket\_bisnis

The screenshot shows the phpMyAdmin interface for the 'dbticket' database. The left sidebar lists various databases and tables, with 'tiket\_bisnis' selected. The main panel displays a green message: 'MySQL memberikan hasil kosong (atau nol baris). (Pencarian dilakukan dalam 0,0003 detik.)'. Below this, the SQL query 'SELECT \* FROM `tiket\_bisnis`' is shown. The table structure is visible with columns: id, nama, harga, and alamat. The 'Operasi hasil kueri' section includes a 'Buat tampilan' button. The 'Markahi kueri SQL ini' section has a 'Markahi kueri SQL ini' button.

- tabel tiket\_ekonomi

The screenshot shows the phpMyAdmin interface for the 'dbticket' database. The left sidebar lists various databases and tables, with 'tiket\_ekonomi' selected. The main panel displays a green message: 'MySQL memberikan hasil kosong (atau nol baris). (Pencarian dilakukan dalam 0,0005 detik.)'. Below this, the SQL query 'SELECT \* FROM `tiket\_ekonomi`' is shown. The table structure is visible with columns: nama, alamat, harga, and id. The 'Operasi hasil kueri' section includes a 'Buat tampilan' button. The 'Markahi kueri SQL ini' section has a 'Markahi kueri SQL ini' button.

User ketika ingin keluar dari aplikasi



```
----- ++ Aplikasi Tiket Pesawat ++ -----  
=====  
Harga Tiket Bisnis: 20000  
Harga Tiket Ekonomi: 15000  
=====  
Pilih Ingin Membuat Jenis Tiket Apa :  
1. Bisnis  
2. Ekonomi  
3. Lihat Rincian Tiket  
4. Simpan ke database  
5. Lihat Data yang tersimpan  
6. Hapus Semua data di database  
  
99. Keluar  
  
=====  
Pilihan Anda (1 .. 5) : 99  
Keluar...
```

Maka user akan keluar dari aplikasi .



## REFERENSI

Berikut referensi yang di gunakan dalam pembuatan aplikasi ini

“C# OOP (Object-Oriented Programming).” Accessed July 26, 2024.

[https://www.w3schools.com/cs/cs\\_oop.php](https://www.w3schools.com/cs/cs_oop.php).

Kamarudin. “Mengakses Database MySQL Menggunakan C# Bagian 1.” >>

Coding4ever’s Blog <<, June 25, 2011.

<http://coding4ever.net/blog/2011/06/25/mengakses-database-mysql-menggunakan-c-bagian-1/>.

———. “Mengakses Database MySQL Menggunakan C# Bagian 2.” >>

Coding4ever’s Blog <<, June 26, 2011.

<http://coding4ever.net/blog/2011/06/26/mengakses-database-mysql-menggunakan-c-bagian-2/>.

Rached, Etienne. “Connect C# to MySQL.” CodeProject, November 18, 2009.

<https://www.codeproject.com/Articles/43438/Connect-C-to-MySQL>.