Laporan Implementasi Basis Data:
Optimisasi Informasi Biaya Pendidikan dan Prospek Kerja Tiap Prodi bagi Calon Mahasiswa UI

Anggota Kelompok 7:

Naufal Elban Musyaffa L. (2206053865)

Kayla Zahira Amadya (2206053890)

Meuthia Nabila Jauhari (2206820434)

Najwa Putri Faradila (2206051355)

Soraya Indira Putri D. (2206053902)

PROGRAM STUDI STATISTIKA

UNIVERSITAS INDONESIA

2024

DAFTAR ISI

BAB I
PENDAHULUAN

1.1 Latar Belakang

Banyak kandidat mahasiswa yang tertarik untuk melanjutkan pendidikan ke jenjang perguruan tinggi. Namun, sering kali mereka menghadapi kesulitan dalam memperoleh informasi yang komprehensif dan terperinci mengenai program-program studi yang ditawarkan. Informasi yang tersebar di berbagai sumber dapat menimbulkan kebingungan dan sulit diakses.

Keterbatasan akses terhadap informasi ini dapat menjadi hambatan bagi kandidat mahasiswa dalam membuat keputusan yang tepat mengenai pendidikan lanjutannya. Dengan adanya sebuah basis data yang mengintegrasikan semua informasi tersebut, terutama di lingkungan Universitas Indonesia, diharapkan dapat memberikan pemahaman yang lebih mendalam bagi kandidat mahasiswa mengenai opsi yang tersedia, serta mendukung mereka dalam mengambil keputusan yang tepat terkait pendidikan dan karier mereka.

1.2 Tujuan

- Mengetahui daftar lengkap fakultas dan jurusan yang ditawarkan oleh UI
- Mengetahui biaya pendidikan (UKT) terhadap masing-masing program studi
- Mendapatkan informasi tentang prospek karir setelah lulus dari program studi tertentu

1.3 Metode Penelitian

Dalam project ini, kami melakukan beberapa tahapan dalam melakukan perancangan database kami,

1. Analisis dan Pengumpulan Informasi
   Langkah awal dari perancangan ini adalah menganalisis masalah berdasarkan latar belakang yang dibuat. Setelah itu, dilanjutkan dengan melakukan pengumpulan informasi mengenai masalah tersebut.

2. Pembuatan Entitas dan Atribut

   Langkah selanjutnya adalah membuat entitas dan atribut pada kasus ini. Entitas menjelaskan mengenai jenis objek yang relevan dalam sistem database. Setelah itu, menentukan atribut yang menjadi karakteristik atau properti yang dimiliki oleh entitas.

3. Menentukan Primary Key dan Foreign Key untuk setiap Entitas

   Primary key dan foreign key menjadi kunci yang membedakan setiap entitas serta menghubungkan antar entitas. Primary key menjadi suatu nilai dalam basis data yang digunakan untuk mengidentifikasi suatu baris dalam tabel yang bernilai unik. Sedangkan foreign key menjadi pengenal unik atau kombinasi pengenal unik yang menghubungkan dua tabel atau lebih dalam database.

4. Pembuatan ER Diagram

   Pembuatan ER Diagram bertujuan untuk menggambarkan entitas, atribut, dan relasi antar entitas untuk memahami kebutuhan sistem, mengidentifikasi kelemahan desain, dan mengkomunikasikan pemahaman tentang struktur data secara visual. Dalam tahap ini, pembuatan ER Diagram menggunakan bantuan software online, yaitu draw.io untuk memudahkan dalam visualisasi.

5. Implementasi SQL

   dalam implementasi SQL digunakan perintah INSERT, CREATE, READ, dan sebagainya. Dalam tahap ini, perintah tersebut digunakan untuk membuat tabel, menyisipkan data, memilih data. Setelah itu, hasil dari simulasi dianalisis untuk mengetahui hubungan antar tabel relasi yang telah dibuat.

6. Implementasi GUI

   Pada implementasi GUI dibuat user interface yang intuitif dan ramah pengguna (user-friendly) untuk memudahkan calon mahasiswa dalam mengakses informasi.

7. Testing

   Selanjutnya dilakukan testing untuk memastikan bahwa aplikasi berfungsi dengan baik dan memenuhi kebutuhan pengguna. Dalam testing dilakukan pengujian

tombol, fungsi pencarian, dll. Akan dipastikan juga bahwa GUI terintegrasi dengan database dan sistem backend dengan baik.

8. Membuat kesimpulan

   Langkah akhir tahapan perancangan adalah membuat kesimpulan dari hasil analisis simulasi yang telah dilakukan untuk mendapatkan output dari sistem database yang telah dirancang.

BAB II
LANDASAN TEORI

2.1. Sistem Informasi Akademik

Sistem informasi akademik menjadi hal yang sangat penting bahkan wajib dimiliki oleh setiap perguruan tinggi. Perannya yang sangat besar dapat membantu pihak penyelanggara pendidikan terutama perguruan tinggi dalam manajemen dan mengolah data seluruh civitasnya. Berdasarkan definisinya, sistem informasi akademik adalah sebuah platform yang dirancang khusus untuk mengelola dan menyimpan data akademik di lingkungan pendidikan, terutama di perguruan tinggi dan institusi pendidikan tinggi lainnya. Sistem ini membantu dalam pengelolaan berbagai aspek administratif dan akademik, mulai dari penerimaan mahasiswa baru hingga proses kelulusan.

Dengan adanya sistem informasi akademik ini harapannya seluruh instansi terutama calon mahasiswa dapat melakukan proses pengambilan keputusan secara efektif dan efisien sesuai database yang ada.

2.2. Komponen Utama Sistem Informasi Akademik

Setiap perguruan tinggi yang ada pastilah memiliki landasan perihal informasi akademik yang dapat diakses oleh seluruh masyarakat dengan tak terkecuali. Berikut beberapa komponen utama informasi akademik.

2.2.1. Program Pendidikan

Program pendidikan merujuk pada serangkaian rencana, kegiatan, dan upaya yang dirancang untuk mencapai tujuan tertentu dalam proses belajar dan pengembangan individu. Program pendidikan bisa merujuk pada berbagai tingkatan seperti D3, D4, S1, S2, dll.

2.2.2. Rumpun

Komponen ini mencakup klasifikasi program-program studi berdasarkan kesamaan bidang atau disiplin ilmu yang mereka wakili. Biasanya, program-program

studi yang memiliki fokus atau kecenderungan pada bidang tertentu akan dikelompokkan bersama dalam suatu rumpun pendidikan.

2.2.3. Biaya Pendidikan

Biaya pendidikan mengacu pada semua biaya yang terkait dengan pendidikan atau pelatihan seseorang. Ini mencakup berbagai macam pengeluaran yang diperlukan untuk mendapatkan pendidikan tertentu.

2.2.4. Fakultas

Fakultas adalah unit organisasi di sebuah institusi pendidikan tinggi yang terdiri dari sekelompok departemen atau program studi yang terkait secara akademis. Fakults biasnya memiliki fokus atau spesialisasi dalam bidang ilmu atau disiplin ilmu tertentu.

2.2.5. Program Studi

Komponen ini merupakan unit akademik di sebuah institusi pendidikan tinggi yang menyelenggarakan program-program pendidikan dalam bidang ilmu atau disiplin ilmu tertentu. Program studi biasanya menawarkan gelar akademik yang sesuai dengan program yang diselenggarakan.

2.2.6. Prospek Kerja

Prospek kerja mencakup peluang atau kemungkinan untuk mendapatkan pekerjaan atau karir di bidang tertentu setelah menyelesaikan pendidikan atau pelatihan yang sesuai. Komponen ini terdapat berbagai faktor yang mempengaruhi tingkat permintaan dan persaingan untuk posisi kerja dalam suatu bidang atau industri.

2.3. Basis Data

Basis data adalah kumpulan terstruktur dari informasi yang disimpan dalam suatu sistem komputer. Basis data berfungsi untuk menyimpan, mengelola, dan mengakses data dengan efisien. Data dalam basis data diatur dalam tabel yang terdiri dari baris dan kolom, di mana setiap baris mewakili satu entitas atau objek, dan setiap kolom berisi atribut-atribut yang menggambarkan entitas tersebut.

Basis data digunakan sebagai tempat penyimpanan data yang terkait dan terintegrasi, sehingga memudahkan pengolahan dan analisis data. Dalam basis data, data dapat dicari, dimodifikasi, dan dihapus dengan menggunakan bahasa kueri, seperti

SQL (*Structured Query Language*). Penerapan basis data sangat luas, baik dalam skala kecil maupun besar, di berbagai industri dan sektor, termasuk bisnis, pendidikan, pemerintahan, dan lainnya. Dalam dunia akademik, basis data digunakan untuk menyimpan informasi program pendidikan, program studi, rumpun, fakultas, biaya pendidikan, prospek kerja,dll.

2.4. Entity Relationship Diagram

Entity Relationship Diagram (ERD) adalah sebuah model konseptual yang digunakan untuk menggambarkan hubungan antara entitas-entitas (objek atau konsep) dalam suatu sistem informasi. ERD menggunakan notasi grafis untuk mengilustrasikan entitas, atribut-atributnya, serta hubungan antara entitas-entitas tersebut. ERD terdiri dari tiga komponen utama, yaitu:

1. Entitas

   Entitas dalam ERD mewakili objek atau konsep yang ada dalam sistem. Entitas dapat berupa orang, tempat, objek, atau konsep lain yang relevan dalam domain sistem yang sedang dianalisis. Setiap entitas diidentifikasi oleh suatu atribut kunci (primary key) yang unik.

2. Atribut

   Atribut adalah karakteristik atau properti yang dimiliki oleh entitas. Atribut menjelaskan informasi yang terkait dengan entitas tersebut.

3. Relasi

   Relasi adalah hubungan dalam ERD menggambarkan keterkaitan antara entitas-entitas. Dalam relasi tersebut memiliki *link* yang menjelaskan jumlah hubungan antara satu entitas dengan entitas lainnya yang disebut kardinalitas. Memiliki empat jenis kardinalitas, yaitu satu-ke-satu (*One-to-One*), satu-ke-banyak (*One-to-Many*), banyak-ke-satu (*Many-to-One*) atau banyak-ke-banyak (*Many-to-Many*). *One-to-One* (1:1) menunjukkan bahwa satu entitas pada satu sisi hubungan hanya terhubung dengan satu entitas pada sisi lainnya. *One-to-Many* (1:N) menunjukkan bahwa satu entitas pada satu sisi hubungan dapat terhubung dengan banyak entitas pada sisi lainnya. *Many-to-One* (N:1) menunjukkan bahwa banyak entitas pada satu sisi hubungan terhubung dengan satu entitas pada sisi lainnya. *Many-to-Many*

(N:N) menunjukkan bahwa banyak entitas pada satu sisi hubungan dapat terhubung dengan banyak entitas pada sisi lainnya.

ERD membantu dalam pemodelan dan perencanaan struktur basis data. Dengan menggambarkan entitas, atribut, dan relasi antara mereka, ERD membantu pemangku kepentingan dalam memahami kebutuhan sistem, mengidentifikasi kelemahan desain, dan mengkomunikasikan pemahaman tentang struktur data secara visual. ERD juga merupakan alat yang berguna dalam pengembangan perangkat lunak, karena menyediakan panduan yang jelas dalam merancang skema basis data dan menjaga konsistensi dalam relasi antara entitas-entitas yang ada.

2.5. Tabel dan Deskripsi

Tabel memiliki beberapa komponen penting seperti:

1. Kolom (*Field*)

   Kolom dalam tabel relasional mewakili atribut atau karakteristik dari entitas yang direpresentasikan oleh tabel tersebut. Setiap kolom memiliki nama yang unik dan tipe data yang mendefinisikan jenis nilai yang dapat disimpan di dalamnya, seperti teks, angka, tanggal, atau boolean.

2. Baris (*Record*)

   Baris dalam tabel relasional mewakili instance atau contoh dari entitas yang direpresentasikan oleh tabel tersebut. Setiap baris menyimpan nilai-nilai yang sesuai untuk setiap kolom dalam tabel.

3. Kunci (*Key*):

   Kunci dalam tabel relasional digunakan untuk mengidentifikasi secara unik setiap baris dalam tabel. Kunci utama (*primary key*) adalah kunci yang unik untuk setiap baris dan memungkinkan penggunaan operasi CRUD (CREATE, READ, UPDATE, DELETE) untuk manipulasi data. Ada juga kunci asing (*foreign key*) yang menghubungkan tabel relasional dengan tabel lain dalam basis data relasional.

4. Relasi antara Tabel

   Relasi antara tabel dalam basis data relasional ditentukan oleh *foreign key* yang menghubungkan satu tabel dengan tabel lain. *Foreign key* mengacu pada *primary key* di tabel yang terkait, membentuk keterkaitan antara entitas yang direpresentasikan oleh tabel-tabel tersebut.

2.6. Tabel Relasional

2.6.1. Primary Key

Primary key adalah sebuah kolom atau kombinasi kolom dalam sebuah tabel yang secara unik mengidentifikasi setiap baris atau record dalam tabel tersebut. *Primary key* digunakan untuk memberikan identitas unik pada setiap *record* dalam tabel, sehingga tidak ada dua baris yang memiliki nilai *primary ke*y yang sama. *Primary key* memiliki karakteristik yaitu harus unik atau tidak boleh ada duplikat atau nilai yang sama dalam kolom *primary key*; *not null* atau memiliki nilai null (kosong); tetap atau tidak boleh diubah setelah ditetapkan untuk sebuah *record*. *Primary key* digunakan sebagai *foreign key* (kunci asing) dalam tabel lain yang berhubungan.

2.6.2. Foreign Key

Foreign key (kunci asing) adalah kolom atau kelompok kolom dalam sebuah tabel yang merujuk pada primary key dari tabel lain. *Foreign key* membangun hubungan atau asosiasi antara dua tabel dalam basis data relasional. *Foreign key* memiliki karakteristik yaitu merujuk ke *primary key* dari tabel referensi; menjaga integritas referensial antara dua tabel dan mencegah terjadinya anomali data atau kehilangan integritas; membentuk hubungan atau asosiasi antara tabel yang berbeda dalam basis data; dan memberlakukan batasan pada data yang dimasukkan atau dihapus dari tabel. Penggunaan *foreign key* membantu dalam pengorganisasian data yang terkait, menjaga konsistensi dan integritas referensial, serta memungkinkan pengambilan data terkait antara tabel dalam basis data relasional.

2.7. DB Browser SQLite

Data yang digunakan merupakan data relasional, karena saling berhubungan satu dengan yang lainnya. Oleh karena itu, bahasa pemrograman yang dapat digunakan untuk komunikasi dengan sistem basis data relasional adalah *Sturctured Query Language* (SQL).

Terdapat 6 perintah yang dapat dilakukan oleh SQL terhadap *relational database*, yakni:

1. Membuat tabel (CREATE);
2. Menyisipkan data (INSERT);

3. Memperbaharui data (UPDATE);
4. Memilih data (READ);
5. Menghapus data (DELETE); dan
6. Menyaring data (WHERE).

Salah satu *software* yang menggunakan bahasa pemrograman SQL adalah DB Browser for SQLite dengan keunggulan sebagai berikut:

- Berbasis file, sehingga mudah untuk diatur dan digunakan;
- Cocok untuk pengembangan dan pengujian dasar;
- *Open source;*
- Menggunakan sintaks SQL standar dengan perubahan kecil; dan
- Mudah digunakan.

BAB III
TABEL RELASIONAL

3.1 Metode Perancangan Basis Data

Database Planning

Basis data yang akan digunakan menyimpan rincian data entitas seperti program pendidikan, rumpun, biaya pendidikan, fakultas, program studi, program kerja. Berikut adalah atribut dari masing-masing entitas.

- Program pendidikan memiliki atribut ID_Program (Primary Key) dan Jenjang.
- Rumpun memiliki atribut ID_Rumpun (Primary Key) dan Nama_Rumpun.
- Biaya pendidikan memiliki atribut ID_Biaya (Primary Key), ID_Rumpun (Foreign Key), Golongan, Harga.
- Fakultas memiliki atribut ID_Fakultas (Primary Key), Nama_Fakultas, Biaya_Golongan_Terendah, Biaya_Golongan_Tertinggi, ID_Rumpun (Foreign Key).
- Program studi memiliki atribut ID_Prodi (Primary Key) , Nama_Prodi, Akreditasi, Daya_Tampung, ID_Fakultas (Foreign Key), ID_Program (Foreign Key).
- Prospek kerja memiliki atribut ID_Prospek, (Primary Key), Nama_Prodi, Gaji_per_bulan, ID_Prodi (Foreign Key).

3.2 Tabel Relasional dan Deskripsi

- Tabel Program Pendidikan

  Tabel ini berisikan informasi mengenai program-program pendidikan yang dapat diambil oleh calon mahasiswa Universitas Indonesia. Contohnya seperti program pendidikan D3, D4, dn S1. Tabel Program Pendidikan memiliki 2 degree dan 3 kardinalitas.

Nama Field	Tipe Data	Domain	Keterangan
ID_Program	CHAR/TEXT	Terdiri dari kumpulan karakter yang merepresentasikan ID program pendidikan {J001, J002, J003}	PK

Jenjang	CHAR/TEXT	Terdiri dari kumpulan karakter yang merepesentasikan jenjang pendidikan {S1, D3, D4}	

- Tabel Rumpun

  Tabel ini berisikan informasi mengenai rumpun pendidikan yang tersedia di Universitas Indonesia seperti Rumpun Ilmu Kesehatan, Sains dan Teknologi, Ilmu Sosial dan Humaniora, dan Vokasi. Tabel ini terdiri dari 2 degree dan 4 kardinalitas.

Nama Field	Tipe Data	Domain	Keterangan
ID_Rumpun	CHAR/TEXT	Terdiri dari kumpulan karakter yang merepresentasikan ID rumpun yang tersedia {R001, R002, ...}	PK
Nama_Rumpun	CHAR/TEXT	Terdiri dari kumpulan karakter string yang merepresentasikan nama rumpun	

- Tabel Biaya Pendidikan

  Tabel ini berisikan informasi mengenai golongan biaya pendidikan yang harus dibayar oleh mahasiswa persemester. Golongan biaya terdiri dari 11 golongan dan besar biaya dipengaruhi oleh rumpun pendidikan. Tabel ini terdiri dari 4 degree dan 44 kardinalitas.

Nama Field	Tipe Data	Domain	Keterangan
ID_Biaya	CHAR/TEXT	Terdiri dari kumpulan karakter yang merepresentasikan ID biaya pendidikan {B00101, B00102, ...}	PK
Golongan	CHAR/TEXT	Terdiri dari kumpulan karakter yang merepresentasikan golongan dari biaya pendidikan	
Harga	CHAR/TEXT	Terdiri dari kumpulan karakter yang merepresentasikan rentang harga dari suatu biaya pendidikan persemester	

ID_Rumpun	CHAR/TEXT	Terdiri dari kumpulan karakter yang merepresentasikan ID rumpun yang tersedia {R001, R002, ...}	FK

- Tabel Fakultas

  Tabel ini berisikan informasi mengenai fakultas-fakultas yang tersedia di Universitas Indonesia seperti Fakultas Teknik, Kedokteran, Kedokteran Gigi, Matematika dan Ilmu Pengetahuan Alam, Hukum, Psikologi, Ilmu Budaya, Ilmu Sosial dan Politik, dan sebagainya. Tabel ini terdiri dari 5 degree dan 15 kardinalitas.

Nama Field	Tipe Data	Domain	Keterangan
ID_Fakultas	CHAR/TEXT	Terdiri dari kumpulan karakter yang merepresentasikan ID sebuah fakultas {F001, F002, ...}	PK
Nama_Fakultas	CHAR/TEXT	Terdiri dari kumpulan karakter string yang merepresentasikan nama fakultas	
Biaya_Golongan_Terendah	CHAR/TEXT	Terdiri dari kumpulan karakter merepresentasikan biaya terendah dalam sebuah fakultas	
Biaya_Golongan_Tertinggi	CHAR/TEXT	Terdiri dari kumpulan karakter merepresentasikan biaya tertinggi dalam sebuah fakultas	
ID_Rumpun	CHAR/TEXT	Terdiri dari kumpulan karakter yang merepresentasikan ID rumpun yang tersedia {R001, R002, ...}	FK

- Tabel Program Studi

  Tabel ini berisikan informasi mengenai program studi yang ditawarkan untuk calon mahasiswa baru oleh Universitas Indonesia. Contohnya seperti Pendidikan Dokter, Gizi, Farmasi, Ilmu Aktuari, Sistem Informasi, Teknik Industi, Psikologi, Ilmu Politik, Akutansi, Administrasi Rumah Sakit dan masih banyak lagi. Tabel ini terdiri dari 6 degree dan 79 kardinalitas (terdapat 79 program studi yang ditawarkan).

Nama Field	Tipe Data	Domain	Keterangan

ID_Prodi	CHAR/TEXT	Terdiri dari kumpulan karakter yang merepresentasikan ID sebuah program studi {P001, P002, ...}	PK
Nama_Prodi	CHAR/TEXT	Terdiri dari kumpulan karakter string yang merepresentasikan nama program studi	
Akreditasi	CHAR/TEXT	Terdiri dari kumpulan karakter string yang merepresentasikan akreditasi nasional sebuah program studi	
Daya Tampung	INTEGER	Terdiri dari kumpulan integer postif yang merepresentasikan jumlah daya tampung sebuah program studi	
ID_Fakultas	CHAR/TEXT	Terdiri dari kumpulan karakter yang merepresentasikan ID sebuah fakultas {F001, F002, ...}	FK
ID_Program	CHAR/TEXT	Terdiri dari kumpulan karakter yang merepresentasikan ID program pendidikan {J001, J002, J003}	FK

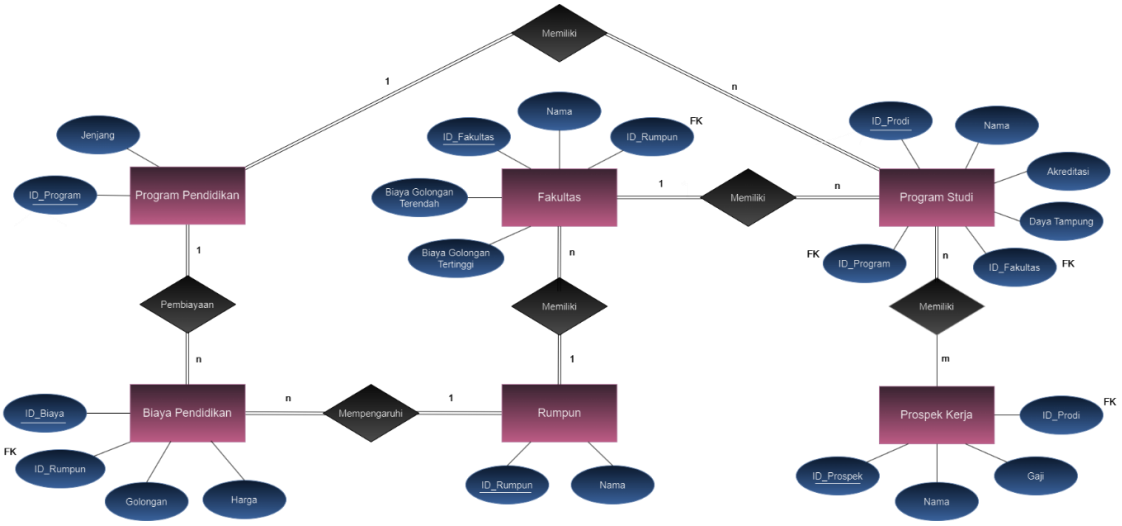
- Tabel Prospek Kerja

  Tabel ini berisikan informasi mengenai prospek kerja yang mungkin dari program-program studi yang ditawarkan. Tabel ini terdiri adri 4 degree dan 158 kardinalitas.

Nama Field	Tipe Data	Domain	Keterangan
ID_Prospek	CHAR/TEXT	Terdiri dari kumpulan karakter yang merepresentasikan ID suatu prospek kerja {JP0001, JP0002, ...}	PK
Nama_Prospek	CHAR/TEXT	Terdiri dari kumpulan karakter string yang merepresentasikan nama prospek kerja.	
Gaji_per_bulan	CHAR/TEXT	Terdiri dari kumpulan karakter yang merepresentasikan rentang gaji perbulan suatu prospek	
ID_Prodi	CHAR/TEXT	Terdiri dari kumpulan karakter yang merepresentasikan ID sebuah program studi {P001, P002, ...}	FK

3.3 Entity Relationship (ER) Diagram

Berikut ini adalah ER Diagram untuk rancangan database kami.



Berikut adalah penjelasan dari ER Diagram di atas:

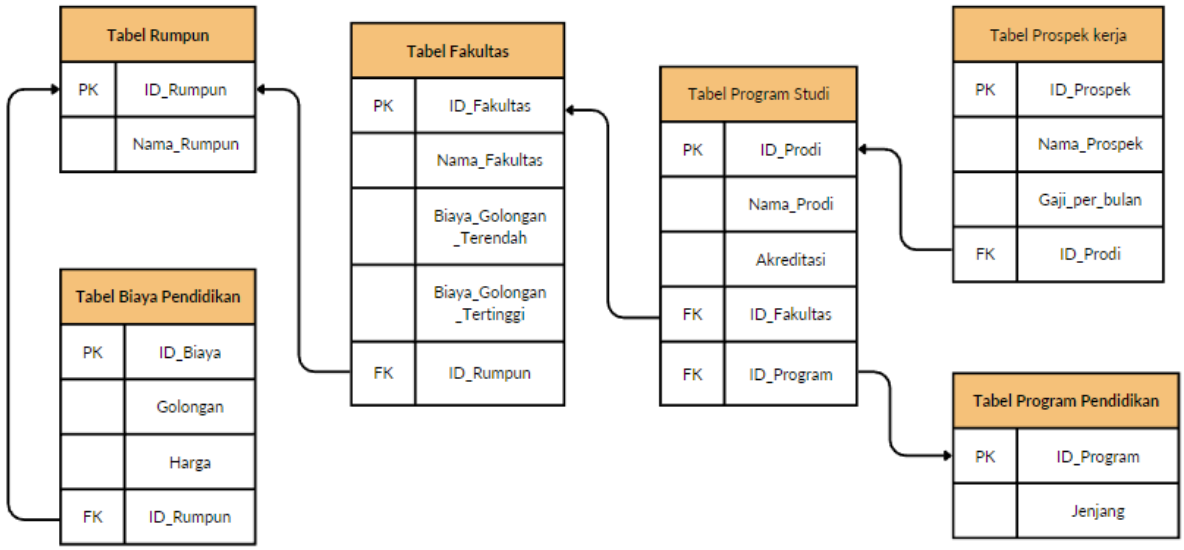
- Dalam satu program pendidikan terdapat banyak jenis pembiayaan untuk biaya pendidikan, tetapi satu biaya pendidikan hanya untuk pembiayaan satu program pendidikan (hubungan one to many). Semua program pendidikan sudah pasti harus melewati pembiayaan (total participation) dan semua biaya pendidikan pasti harus dilakukan pembiayaan (total participation).
- Rumpun mempengaruhi biaya pendidikan. Sebuah rumpun mempengengaruhi banyak jenis biaya pendidikan, tetapi satu biaya pendidikan hanya dipengaruhi oleh satu rumpun (hubungan one to many). Semua rumpun sudah pasti mempengaruhi setidaknya satu biaya pendidikan (total participation) dan semua biaya pendidikan pasti dipengaruhi rumpun (total participation).
- Sebuah rumpun memiliki banyak fakultas di dalamnya, tetapi satu fakultas hanya dimiliki oleh satu rumpun (hubungan one to many). Masing-masing rumpun sudah pasti memiliki setidaknya satu fakultas (total participation) dan masing-masing fakultas dimiliki oleh satu rumpun (total participation).
- Sebuah fakultas memiliki banyak program studi yang ditawarkan, tetapi satu program studi hanya dimiliki oleh satu fakultas (hubungan one to many). Masing-masing

fakultas sudah pasti memiliki setidaknya satu program studi (total participation) dan masing-masing program studi dimiliki oleh sebuah fakultas (total participation).

- Sebuah program studi memiliki banyak prospek kerja dan sebuah prospek kerja dapat dimiliki oleh banyak program studi (hubungan many to many). Masing-masing program studi pasti setidaknya memiliki satu prospek kerja (total participation), tetapi tidak semua prospek kerja dimiliki oleh program studi (partial participation)
- Sebuah program pendidikan memiliki banyak program studi, tetapi sebuah program studi hanya dimiliki oleh satu program pendidikan (hubungan one to many). Masing-masing program pendidikan setidaknya memiliki satu program studi (total participation) dan masing-masing program studi dimiliki oleh sebuah program pendidikan (total participation).

3.4 Relasi Tabel

Berikut adalah relasi dari masing-masing tabel.



BAB IV
IMPLEMENTASI DAN TESTING

4.1 Implementasi SQL

1. Import package

```
import sqlite3
import pandas as pd
```

2. Create table dan database
   - Membuat database

```
def create_db_and_tables():
    conn = sqlite3.connect('C:/Users/najwa/Downloads/Database_data/Database UI.db')
    cursor = conn.cursor()
```

   - Tabel Rumpun

```
    cursor.executescript('''
    CREATE TABLE IF NOT EXISTS Rumpun (
                        ID_Rumpun TEXT PRIMARY KEY NOT NULL,
                        Nama_Rumpun TEXT);
```

   - Tabel Program Studi

```
    CREATE TABLE IF NOT EXISTS ProgramStudi (
                        ID_Prodi TEXT PRIMARY KEY NOT NULL,
                        Nam_Prodi TEXT,
                        Akreditasi TEXT,
                        Daya_Tampung INTEGER,
                        ID_Fakultas TEXT,
                        ID_Program TEXT,
                        FOREIGN KEY (ID_Fakultas) REFERENCES Fakultas(ID_Fakultas),
                        FOREIGN KEY (ID_Program) REFERENCES Program(ID_Program));
```

   - Tabel Biaya Pendidikan

```
    CREATE TABLE IF NOT EXISTS BiayaPendidikan (
                        ID_Biaya TEXT PRIMARY KEY NOT NULL,
                        Golongan TEXT,
                        Harga TEXT,
                        ID_Rumpun,
                        FOREIGN KEY (ID_Rumpun) REFERENCES Rumpun(ID_Rumpun));
```

   - Tabel Prospek Kerja

```
    CREATE TABLE IF NOT EXISTS ProspekKerja (
                        ID_Prospek TEXT PRIMARY KEY NOT NULL,
                        Nama_Prospek TEXT,
                        Gaji_per_bulan TEXT,
                        ID_Prodi,
                        FOREIGN KEY (ID_Prodi) REFERENCES ProgramStudi(ID_Prodi));
```

   - Tabel Program Pendidikan

```
    CREATE TABLE IF NOT EXISTS ProgramPendidikan (
                        ID_Program TEXT PRIMARY KEY NOT NULL,
                        Jenjang TEXT);
```

   - Tabel Fakultas

```
    CREATE TABLE IF NOT EXISTS Fakultas (
                        ID_Fakultas TEXT PRIMARY KEY NOT NULL,
                        Nama_Fakultas TEXT,
                        Biaya_Golongan_Terendah TEXT,
                        Biaya_Golongan_Tertinggi TEXT,
                        ID_Rumpun,
                        FOREIGN KEY (ID_Rumpun) REFERENCES Rumpun(ID_Rumpun));
    ''')
```

3. Memasukkan data ke dalam tabel

Sebelumnya data sudah dikumpulkan dalam format csv yang dapat diakses pada tautan bit.ly/Data_MiniProjectDatabase_Kelompok7.

```
def load_data():
    conn = sqlite3.connect('C:/Users/najwa/Downloads/Database_data/Database UI.db')

    df_fakultas = pd.read_csv('C:/Users/najwa/Downloads/Database_data/Fakultas.csv', encoding='utf-8')
    df_fakultas.to_sql('Fakultas', conn, if_exists='replace', index=False)

    df_program_studi = pd.read_csv('C:/Users/najwa/Downloads/Database_data/Program Studi.csv', encoding='utf-8')
    df_program_studi.to_sql('ProgramStudi', conn, if_exists='replace', index=False)

    df_biaya_pendidikan = pd.read_csv('C:/Users/najwa/Downloads/Database_data/Biaya Pendidikan.csv', encoding='utf-8')
    df_biaya_pendidikan.to_sql('BiayaPendidikan', conn, if_exists='replace', index=False)

    df_rumpun = pd.read_csv('C:/Users/najwa/Downloads/Database_data/Rumpun.csv', encoding='utf-8')
    df_rumpun.to_sql('Rumpun', conn, if_exists='replace', index=False)

    df_prospek_kerja = pd.read_csv('C:/Users/najwa/Downloads/Database_data/Prospek Kerja.csv', encoding='utf-8')
    df_prospek_kerja.to_sql('ProspekKerja', conn, if_exists='replace', index=False)

    df_program_pendidikan = pd.read_csv('C:/Users/najwa/Downloads/Database_data/Program Pendidikan.csv', encoding='utf-8')
    df_program_pendidikan.to_sql('ProgramPendidikan', conn, if_exists='replace', index=False)

    conn.commit()
    conn.close()
```

4. Implementasi SQL ketika memilih opsi Biaya Pendidikan
   - Fungsi query untuk Biaya Pendidikan berdasarkan keyword

   Keyword yang akan digunakan ialah nama Program Studi yang lengkap sehingga digunakan "=".

```
def query_biaya_pendidikan(keyword):
    conn = sqlite3.connect('C:/Users/najwa/Downloads/Database_data/Database UI.db')
    cursor = conn.cursor()

    query = """
    SELECT b.ID_Biaya, b.Golongan, b.Harga
    FROM BiayaPendidikan b
    JOIN Rumpun r ON b.ID_Rumpun = r.ID_Rumpun
    JOIN Fakultas f ON f.ID_Rumpun = r.ID_Rumpun
    JOIN ProgramStudi ps ON f.ID_Fakultas = ps.ID_Fakultas
    WHERE ps.Nama_Prodi = ? OR f.Nama_Fakultas = ?
    """
    cursor.execute(query, (keyword, keyword))
    results = cursor.fetchall()
    conn.close()
    return results
```

   Query ini memilih kolom ID_Biaya, Golongan, dan Harga dari tabel BiayaPendidikan (disingkat b). Tabel ini satukan dengan tabel Rumpun (r), Fakultas (f), dan ProgramStudi (ps) berdasarkan ID terkait. Kondisi pada WHERE menyaring data berdasarkan Nama_Prodi atau Nama_Fakultas yang sesuai dengan keyword yang diberikan.

5. Implementasi SQL ketika memilih opsi Program Pendidikan dan Fakultas
   - Fungsi query untuk Program Pendidikan berdasarkan Jenjang

```
def query_program_pendidikan(jenjang):
    conn = sqlite3.connect('C:/Users/najwa/Downloads/Database_data/Database UI.db')
    cursor = conn.cursor()
    query = """
    SELECT r.Nama_Rumpun, f.Nama_Fakultas, f.Biaya_Golongan_Terendah, f.Biaya_Golongan_Tertinggi
    FROM ProgramPendidikan pp
    JOIN ProgramStudi ps ON ps.ID_Program = pp.ID_Program
    JOIN Fakultas f ON ps.ID_Fakultas = f.ID_Fakultas
    JOIN Rumpun r ON f.ID_Rumpun = r.ID_Rumpun
    WHERE pp.Jenjang = ?
    """
    cursor.execute(query, (jenjang,))
    results = cursor.fetchall()
    conn.close()
    return results
```

Query ini memilih kolom Nama_Rumpun, Nama_Fakultas, Biaya_Golongan_Terendah, dan Biaya_Golongan_Tertinggi dari tabel ProgramPendidikan (disingkat pp). Tabel ini di-join dengan tabel ProgramStudi (ps), Fakultas (f), dan Rumpun (r) berdasarkan ID terkait. Kondisi pada WHERE menyaring data berdasarkan Jenjang yang sesuai dengan parameter jenjang yang diberikan.

6. Implementasi SQL ketika memilih opsi Program Studi dan Prospek Kerja
   - Fungsi query untuk Program Studi berdasarkan keyword

```
def query_program_studi(keyword=None):
    conn = sqlite3.connect('C:/Users/najwa/Downloads/Database_data/Database UI.db')
    cursor = conn.cursor()
    if keyword:
        query = """
        SELECT ps.Nama_Prodi, ps.Akreditasi, ps.Daya_Tampung, f.Nama_Fakultas
        FROM ProgramStudi ps
        JOIN Fakultas f ON ps.ID_Fakultas = f.ID_Fakultas
        WHERE ps.Nama_Prodi LIKE ?
        ORDER BY ps.Nama_Prodi
        """
        cursor.execute(query, ('%' + keyword + '%',))
```

Pada bagian ini, jika keyword diberikan, query akan memilih Nama_Prodi, Akreditasi, Daya_Tampung, dan Nama_Fakultas dari tabel ProgramStudi (ps) yang berhubungan dengan tabel Fakultas (f). Kondisi WHERE menggunakan LIKE untuk mencocokkan Nama_Prodi yang mengandung keyword dan hasilnya diurutkan berdasarkan Nama_Prodi.

```
    else:
        query = """
        SELECT ps.Nama_Prodi, ps.Akreditasi, ps.Daya_Tampung, f.Nama_Fakultas
        FROM ProgramStudi ps
        JOIN Fakultas f ON ps.ID_Fakultas = f.ID_Fakultas
        ORDER BY ps.Nama_Prodi
        """
        cursor.execute(query)
    results = cursor.fetchall()
    conn.close()
    return results
```

Jika keyword tidak diberikan (None), query akan memilih kolom yang sama dari tabel yang sama tetapi tanpa kondisi WHERE, dan hasilnya tetap diurutkan berdasarkan Nama_Prodi.

```
def query_prospek_kerja(prodi):
    conn = sqlite3.connect('C:/Users/najwa/Downloads/Database_data/Database UI.db')
    cursor = conn.cursor()
    query = """
    SELECT Nama_Prospek, Gaji_per_bulan
    FROM ProspekKerja
    JOIN ProgramStudi ON ProspekKerja.ID_Prodi = ProgramStudi.ID_Prodi
    WHERE ProgramStudi.Nama_Prodi = ?
    """
    cursor.execute(query, (prodi,))
    results = cursor.fetchall()
    conn.close()
    return results
```

Query ini memilih kolom Nama_Prospek dan Gaji_per_bulan dari tabel ProspekKerja yang di-join dengan tabel ProgramStudi berdasarkan ID_Prodi. Kondisi pada WHERE menyaring data berdasarkan Nama_Prodi yang sesuai dengan parameter prodi yang diberikan.

4.2 Implementasi GUI

1. Import package

```
import tkinter as tk
from tkinter import ttk
```

2. Ketika memilih opsi Biaya Pendidikan
   - Menampilkan pencarian Biaya Pendidikan

```
def show_biaya_pendidikan_search():
    for widget in main_frame.winfo_children():
        widget.destroy()

    ttk.Label(main_frame, text="UIAcademicPortal", font=('Verdana', 50), foreground="dark blue").pack(pady=10)
    separator = ttk.Separator(main_frame, orient='horizontal')
    separator.pack(fill='x', padx=5, pady=5)
    ttk.Label(main_frame, text="Biaya Pendidikan di UI", font=('Verdana', 25)).pack(pady=10)
    highlight_label = tk.Label(main_frame, text="Cari melalui Nama Program Studi", font=('Verdana', 12), bg="yellow")
    highlight_label.pack(pady=5)
    ttk.Label(main_frame, text="Contoh: Ilmu Aktuaria", font=('Verdana', 10)).pack(pady=10)
    search_entry = ttk.Entry(main_frame, width=50)
    search_entry.pack(pady=5)

    search_button = ttk.Button(main_frame, text="Cari", command=lambda: perform_search(search_entry.get()))
    search_button.pack(pady=20)

    back_button = ttk.Button(main_frame, text="Kembali", command=reset_view, style="TButton")
    back_button.pack(pady=(20, 10), side='bottom')
```

   - Melakukan pencarian dan menampilkan hasil pencarian
     Jika terdapat kesalahan penulisan nama Program Studi, maka hasil pencarian akan menampilkan "!! Data Tidak Ditemukan !!"

```
def perform_search(keyword):
    results = query_biaya_pendidikan(keyword)
    for widget in main_frame.winfo_children():
        widget.destroy()

    ttk.Label(main_frame, text="UIAcademicPortal", font=('Verdana', 50), foreground="dark blue").pack(pady=10)
    separator = ttk.Separator(main_frame, orient='horizontal')
    separator.pack(fill='x', padx=5, pady=5)
    ttk.Label(main_frame, text="Hasil Pencarian", font=('Verdana', 30)).pack(pady=20)

    if not results:
        ttk.Label(main_frame, text="‼ Data Tidak Ditemukan  ‼", font=('Verdana', 16), foreground="#8b0000").pack(pady=20)
    else:
        tree = ttk.Treeview(main_frame, columns=("Golongan", "Uang Kuliah Tunggal (Rp)"), show="headings", height=11)
        tree.heading("Golongan", text="Golongan")
        tree.heading("Uang Kuliah Tunggal (Rp)", text="Uang Kuliah Tunggal (Rp)")
        tree.column("Golongan", anchor="center", width=100)
        tree.column("Uang Kuliah Tunggal (Rp)", anchor="center", width=200)

        style = ttk.Style()
        style.configure("Treeview", background="white", fieldbackground="white")
        style.map('Treeview', background=[('selected', 'brown')])
        style.map('Treeview.Heading', background=[('active', 'yellow')])

        for id_biaya, golongan, harga in results:
            tree.insert("", "end", values=(golongan, harga))

        tree.pack(pady=10, fill='x', expand=True)

    back_button = ttk.Button(main_frame, text="Kembali", command=show_biaya_pendidikan_search)
    back_button.pack(pady=(20, 10), side='bottom')
```

3. Ketika memilih opsi Program Pendidikan dan Fakultas
   - Menampilkan button untuk memilih Jenjang Pendidikan yang ada (S1, D3, D4)

```
def show_jenjang_buttons():
    for widget in main_frame.winfo_children():
        widget.destroy()
    ttk.Label(main_frame, text="UIAcademicPortal", font=('Verdana', 50), foreground="dark blue").pack(pady=10)
    separator = ttk.Separator(main_frame, orient='horizontal')
    separator.pack(fill='x', padx=5, pady=5)
    ttk.Label(main_frame, text="Jenjang Pendidikan di UI", font=('Verdana', 30)).pack(pady=10)

    jenjang_names = ["S1", "D3", "D4"]
    for name in jenjang_names:
        button = ttk.Button(main_frame, text=name, style="TButton",
                            command=lambda j=name: show_detail_jenjang(j))
        button.pack(pady=10)

    back_button = ttk.Button(main_frame, text="Kembali", command=reset_view, style="TButton")
    back_button.pack(pady=(20, 10), side='bottom')
```

   - Menampilkan detail dari Jenjang Pendidikan

```
def show_detail_jenjang(jenjang_name):
    results = query_program_pendidikan(jenjang_name)
    for widget in main_frame.winfo_children():
        widget.destroy()

    ttk.Label(main_frame, text="UIAcademicPortal", font=('Verdana', 50), foreground="dark blue").pack(pady=10)
    separator = ttk.Separator(main_frame, orient='horizontal')
    separator.pack(fill='x', padx=5, pady=5)
    ttk.Label(main_frame, text=f"Detail untuk Jenjang {jenjang_name}", font=('Verdana', 30)).pack(pady=10)

    style = ttk.Style()
    style.configure("Treeview", background="white", fieldbackground="white")
    style.map('Treeview', background=[('selected', 'brown')])
    style.map('Treeview.Heading', background=[('active', 'yellow')])

    tree = ttk.Treeview(main_frame, columns=("Rumpun", "Fakultas", "Biaya Terendah", "Biaya Tertinggi"), show="headings", height=15)
    tree.heading("Rumpun", text="Rumpun")
    tree.heading("Fakultas", text="Fakultas")
    tree.heading("Biaya Terendah", text="Uang Kuliah Tunggal Terendah")
    tree.heading("Biaya Tertinggi", text="Uang Kuliah Tunggal Tertinggi")

    fakultas_data = {}
    for rumpun, fakultas, biaya_terendah, biaya_tertinggi in results:
        if fakultas not in fakultas_data:
            fakultas_data[fakultas] = [rumpun, fakultas, biaya_terendah, biaya_tertinggi]
        else:
            existing_biaya_terendah, existing_biaya_tertinggi = fakultas_data[fakultas][2], fakultas_data[fakultas][3]
            fakultas_data[fakultas][2] = min(existing_biaya_terendah, biaya_terendah)
            fakultas_data[fakultas][3] = max(existing_biaya_tertinggi, biaya_tertinggi)

    for fakultas in fakultas_data.values():
        tree.insert("", "end", values=(fakultas[0], fakultas[1], fakultas[2], fakultas[3]))

    tree.pack(pady=10, fill='x', expand=True)

    back_button = ttk.Button(main_frame, text="Kembali", command=show_jenjang_buttons)
    back_button.pack(pady=(20,10), side='bottom')
```

4. Ketika memilih opsi Program Studi dan Prospek Kerja

- Menampilkan Program Studi

```
def show_program_studi():
    for widget in main_frame.winfo_children():
        widget.destroy()

    bg_color = "#ffc067"

    ttk.Label(main_frame, text="UIAcademicPortal", font=('Verdana', 50), foreground="dark blue", background=bg_color).pack(pady=10)
    separator = ttk.Separator(main_frame, orient='horizontal')
    separator.pack(fill='x', padx=5, pady=5)
    ttk.Label(main_frame, text="Program Studi", font=('Verdana', 30), background=bg_color).pack(pady=15)

    search_frame = tk.Frame(main_frame, bg=bg_color)
    search_frame.pack(fill='x', pady=10, padx=5)
    ttk.Label(search_frame, text="Cari Program Studi:", font=('Verdana', 12), background=bg_color).pack(side='left')
    search_entry = ttk.Entry(search_frame, width=50)
    search_entry.pack(side='left', padx=5)
    search_button = ttk.Button(search_frame, text="Cari", command=lambda: search_program_studi(search_entry.get()))
    search_button.pack(side='left')

    tree_frame = tk.Frame(main_frame, bg=bg_color)
    tree_frame.pack(fill='both', expand=True)

    global tree
    tree = ttk.Treeview(tree_frame, columns=("Nama Prodi", "Akreditasi", "Daya Tampung", "Asal Fakultas"), show="headings", height=11)
    tree.heading("Nama Prodi", text="Nama Program Studi")
    tree.heading("Akreditasi", text="Akreditasi")
    tree.heading("Daya Tampung", text="Daya Tampung")
    tree.heading("Asal Fakultas", text="Asal Fakultas")
    tree.column("Nama Prodi", anchor="center", width=200)
    tree.column("Akreditasi", anchor="center", width=100)
    tree.column("Daya Tampung", anchor="center", width=100)
    tree.column("Asal Fakultas", anchor="center", width=150)

    results = query_program_studi()
    for nama_prodi, akreditasi, daya_tampung, nama_fakultas in results:
        tree.insert("", "end", values=(nama_prodi, akreditasi, daya_tampung, nama_fakultas))

    style = ttk.Style()
    style.configure("Treeview", background="white", fieldbackground="white")
    style.map('Treeview', background=[('selected', 'brown')])
    style.map('Treeview.Heading', background=[('active', 'yellow')])

    tree.pack(pady=10, fill='both', expand=True)
    tree.bind("<Double-1>", lambda event: show_prospek_detail(tree))

    ttk.Label(main_frame, text="Klik baris untuk melihat detail informasi", font=('Verdana', 12), foreground="#8b0000", background="yellow").pack(pady=10, anchor="w")
    back_button = ttk.Button(main_frame, text="Kembali", command=reset_view)
    back_button.pack(pady=(20, 10), side='bottom')
```

- Menampilkan pencarian Program Studi
  Disediakan tempat untuk melakukan pencarian karena untuk memudahkan user mencari Program Studi di antara 79 Program Studi yang tersedia.

  Informasi yang ditampilkan dalam aplikasi akan disajikan dalam bentuk tabel. Tabel ini akan memiliki kolom-kolom sebagai berikut:

  - Nama Program Studi
  - Akreditasi
  - Daya Tampung
  - Asal Fakultas

```
def search_program_studi(keyword):
    for widget in main_frame.winfo_children():
        widget.destroy()

    bg_color = "#ffc067"

    ttk.Label(main_frame, text="UIAcademicPortal", font=('Verdana', 50), foreground="dark blue", background=bg_color).pack(pady=10)
    separator = ttk.Separator(main_frame, orient='horizontal')
    separator.pack(fill='x', padx=5, pady=5)
    ttk.Label(main_frame, text="Program Studi", font=('Verdana', 30), background=bg_color).pack(pady=15)

    search_frame = tk.Frame(main_frame, bg=bg_color)
    search_frame.pack(fill='x', pady=10, padx=5)
    ttk.Label(search_frame, text="Cari Program Studi:", font=('Verdana', 12), background=bg_color).pack(side='left')
    search_entry = ttk.Entry(search_frame, width=40)
    search_entry.pack(side='left', padx=5)
    search_entry.insert(0, keyword)
    search_button = ttk.Button(search_frame, text="Cari", command=lambda: search_program_studi(search_entry.get()))
    search_button.pack(side='left')

    tree_frame = tk.Frame(main_frame, bg=bg_color)
    tree_frame.pack(fill='both', expand=True)

    global tree
    tree = ttk.Treeview(tree_frame, columns=("Nama Prodi", "Akreditasi", "Daya Tampung", "Asal Fakultas"), show="headings", height=11)
    tree.heading("Nama Prodi", text="Nama Program Studi")
    tree.heading("Akreditasi", text="Akreditasi")
    tree.heading("Daya Tampung", text="Daya Tampung")
    tree.heading("Asal Fakultas", text="Asal Fakultas")
    tree.column("Nama Prodi", anchor="center", width=200)
    tree.column("Akreditasi", anchor="center", width=100)
    tree.column("Daya Tampung", anchor="center", width=100)
    tree.column("Asal Fakultas", anchor="center", width=150)

    results = query_program_studi(keyword)
    for nama_prodi, akreditasi, daya_tampung, nama_fakultas in results:
        tree.insert("", "end", values=(nama_prodi, akreditasi, daya_tampung, nama_fakultas))

    tree.pack(pady=10, fill='both', expand=True)
    tree.bind("<Double-1>", lambda event: show_prospek_detail(tree))

    ttk.Label(main_frame, text="Klik baris untuk melihat detail informasi", font=('Verdana', 12), foreground="#8b0000", background="yellow").pack(pady=10, anchor="w")
    back_button = ttk.Button(main_frame, text="Kembali", command=show_program_studi)
    back_button.pack(pady=(20, 10), side='bottom')
```

- Menampilkan detail Prospek Kerja untuk Program Studi yang terpilih
  Detail prospek kerja untuk program studi yang dipilih akan ditampilkan ketika pengguna mengklik dua kali pada baris yang dipilih dalam tabel program studi.

```
def show_prospek_detail(tree):
    selected_item = tree.selection()[0]
    prodi = tree.item(selected_item, 'values')[0]
    results = query_prospek_kerja(prodi)

    for widget in main_frame.winfo_children():
        widget.destroy()

    ttk.Label(main_frame, text="UIAcademicPortal", font=('Verdana', 50), foreground="dark blue").pack(pady=10)
    separator = ttk.Separator(main_frame, orient='horizontal')
    separator.pack(fill='x', padx=5, pady=5)
    ttk.Label(main_frame, text=f"Prospek Kerja Program Studi {prodi}", font=('Verdana', 30)).pack(pady=10)

    tree = ttk.Treeview(main_frame, columns=("Nama Prospek", "Gaji per Bulan"), show="headings", height=2)
    tree.heading("Nama Prospek", text="Nama Prospek Kerja")
    tree.heading("Gaji per Bulan", text="Gaji per Bulan")
    tree.column("Nama Prospek", anchor="center", width=200)
    tree.column("Gaji per Bulan", anchor="center", width=150)

    for nama_prospek, gaji_per_bulan in results:
        tree.insert("", "end", values=(nama_prospek, gaji_per_bulan))

    tree.pack(pady=10, fill='x', expand=True)

    back_button = ttk.Button(main_frame, text="Kembali", command=show_program_studi)
    back_button.pack(pady=(20, 10), side='bottom')
```

5. Mengembalikan ke tampilan awal
   Menyediakan fungsi untuk mengatur ulang tampilan aplikasi ke halaman utama.

```
def reset_view():
    for widget in main_frame.winfo_children():
        widget.destroy()
    initialize_main_view()
```

6. Inisialisasi tampilan utama

   Menginisialisasi tampilan utama aplikasi dengan opsi-opsi menu yang dapat dipilih oleh pengguna.

```
def initialize_main_view():
    global canvas, flying_text_instances, flying_text, option_var

    label_welcome = ttk.Label(main_frame, text="UIAcademicPortal", font=('Verdana', 50), foreground="dark blue")
    label_welcome.pack(pady=(20, 20))

    separator = ttk.Separator(main_frame, orient='horizontal')
    separator.pack(fill='x', padx=5, pady=5)

    canvas = tk.Canvas(main_frame, height=100, bg="#ffc067", bd=0, highlightthickness=0)
    canvas.pack(fill='x', pady=20)

    flying_text = "Selamat Datang di UIAcademicPortal"
    flying_text_instances = []

    initial_text_x = root.winfo_width()
    flying_text_id = canvas.create_text(initial_text_x, 30, text=flying_text, font=('Verdana', 8), anchor='w')
    flying_text_instances.append((flying_text_id, initial_text_x))

    option_var = tk.StringVar(value="Informasi apa yang ingin Anda cari?")
    options = ["Biaya Pendidikan", "Program Pendidikan dan Fakultas", "Program Studi dan Prospek Kerja"]
    option_menu = ttk.Combobox(main_frame, textvariable=option_var, values=options, state="readonly", width=35)
    option_menu.pack(pady=20)
    option_menu.bind("<<ComboboxSelected>>", on_combobox_select)
    option_menu.bind("<FocusIn>", lambda e: option_var.set('') if option_var.get() == "Informasi apa yang ingin Anda cari?" else None)
    option_menu.bind("<FocusOut>", lambda e: option_var.set("Informasi apa yang ingin Anda cari?") if not option_var.get() else None)

    update_flying_text()
```

7. Fungsi untuk menampilkan pilihan dari combobox
   Menangani aksi ketika pengguna memilih opsi dari menu combobox dan menampilkan tampilan yang sesuai.

```
def on_combobox_select(event):
    option = option_var.get()
    if option == "Biaya Pendidikan":
        show_biaya_pendidikan_search()
    elif option == "Program Pendidikan dan Fakultas":
        show_jenjang_buttons()
    elif option == "Program Studi dan Prospek Kerja":
        show_program_studi()
```

8. Fungsi untuk memperbarui flying text
   Membuat efek teks berjalan yang ditampilkan pada canvas.

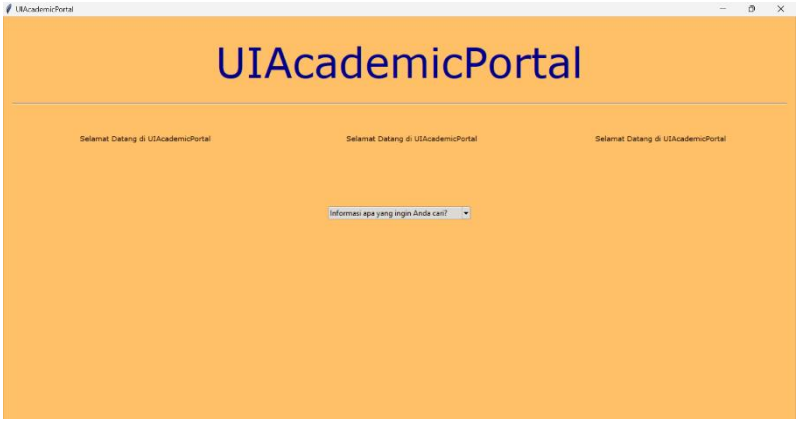
```
def update_flying_text():
    global flying_text_instances

    for text_id, _ in flying_text_instances:
        canvas.move(text_id, -2, 0)

    if canvas.coords(flying_text_instances[0][0])[0] < -canvas.bbox(flying_text_instances[0][0])[2]:
        canvas.delete(flying_text_instances[0][0])
        flying_text_instances.pop(0)

    last_text_id, _ = flying_text_instances[-1]
    if canvas.coords(last_text_id)[0] < root.winfo_width() - 400:
        new_text_x = root.winfo_width()
        new_text_id = canvas.create_text(new_text_x, 30, text=flying_text, font=('Verdana', 8), anchor='w')
        flying_text_instances.append((new_text_id, new_text_x))

    canvas.after(50, update_flying_text)
```

9. Fungsi mainframe untuk menjalankan tampilan aplikasi pertama kali
   Menginisialisasi dan menjalankan aplikasi Tkinter, membuat database, memuat data, dan menampilkan antarmuka pengguna

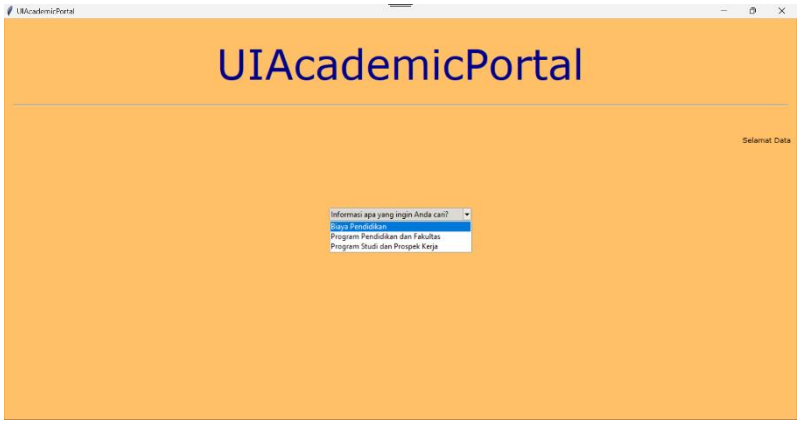
```python
def main():
    global root, main_frame, option_var, label_info, option_menu, canvas, flying_text, flying_text_instances

    create_db_and_tables()
    load_data()

    root = tk.Tk()
    root.title("UIAcademicPortal")
    root.geometry("900x600")

    style = ttk.Style()
    style.theme_use('clam')
    style.configure("Custom.TFrame", background="#ffc067")
    style.configure("TLabel", background="#ffc067", font=('Verdana', 12))
    style.configure("TButton", font=('Verdana', 12))

    main_frame = ttk.Frame(root, style="Custom.TFrame", padding="10")
    main_frame.pack(fill=tk.BOTH, expand=True)

    label_welcome = ttk.Label(main_frame, text="UIAcademicPortal", font=('Verdana', 50), foreground="dark blue")
    label_welcome.pack(pady=(20, 20))

    separator = ttk.Separator(main_frame, orient='horizontal')
    separator.pack(fill='x', padx=5, pady=5)

    canvas = tk.Canvas(main_frame, height=100, bg="#ffc067", bd=0, highlightthickness=0)
    canvas.pack(fill='x', pady=20)

    flying_text = "Selamat Datang di UIAcademicPortal"
    flying_text_instances = []

    initial_text_x = root.winfo_width()
    flying_text_id = canvas.create_text(initial_text_x, 30, text=flying_text, font=('Verdana', 8), anchor='w')
    flying_text_instances.append((flying_text_id, initial_text_x))

    option_var = tk.StringVar(value="Informasi apa yang ingin Anda cari?")
    options = ["Biaya Pendidikan", "Program Pendidikan dan Fakultas", "Program Studi dan Prospek Kerja"]
    option_menu = ttk.Combobox(main_frame, textvariable=option_var, values=options, state="readonly", width=35)
    option_menu.pack(pady=20)
    option_menu.bind("<<ComboboxSelected>>", on_combobox_select)
    option_menu.bind("<FocusIn>", lambda e: option_var.set('') if option_var.get() == "Informasi apa yang ingin Anda cari?" else None)
    option_menu.bind("<FocusOut>", lambda e: option_var.set("Informasi apa yang ingin Anda cari?") if not option_var.get() else None)

    label_info = ttk.Label(main_frame, text="")
    label_info.pack(pady=10, fill='x')

    update_flying_text()

    root.mainloop()

if __name__ == "__main__":
    main()
```

4.3 Testing

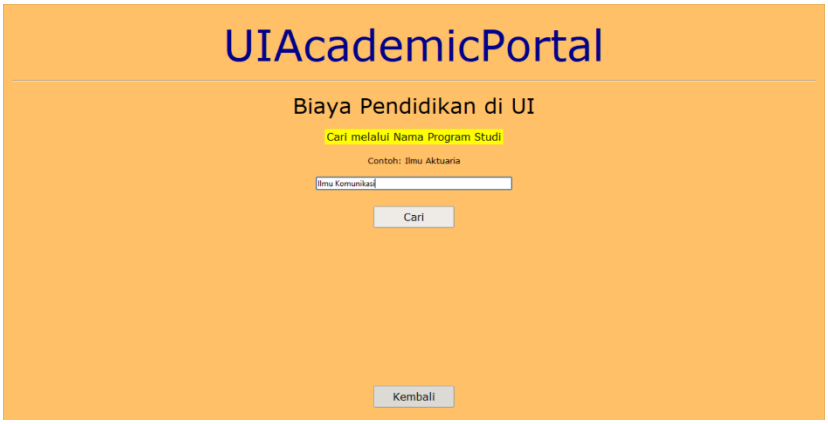
1. Tampilan awal



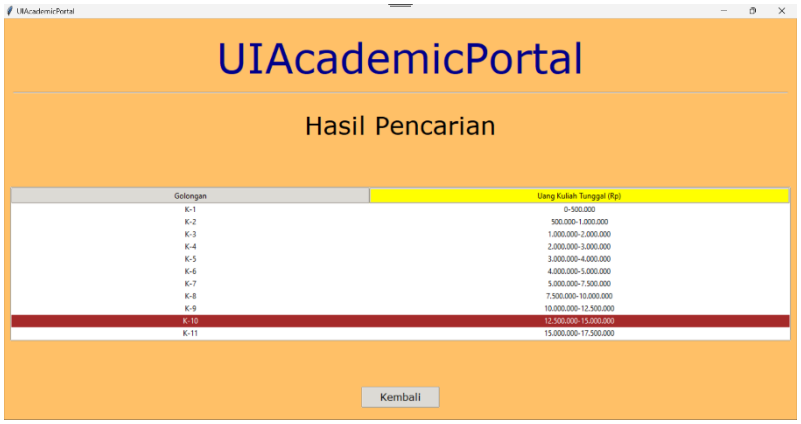
2. Pilih opsi Biaya Pendidikan



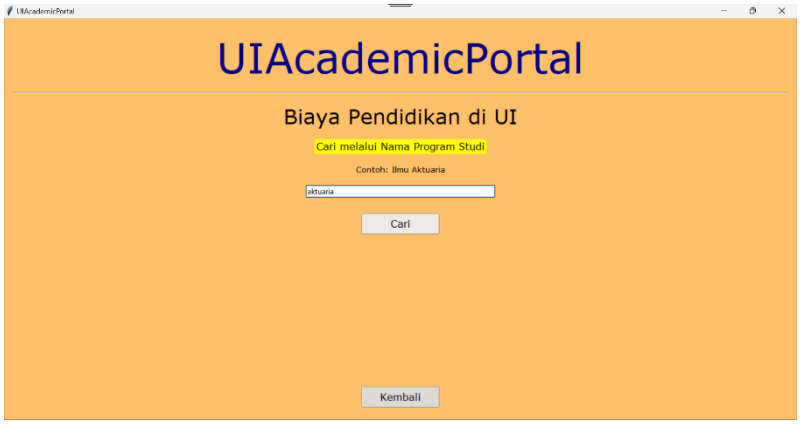
Ketika penulisan benar,



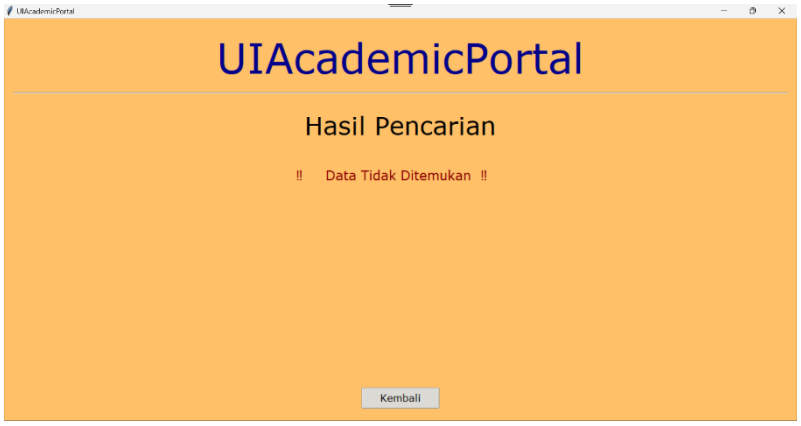
maka akan memberikan output yang sesuai



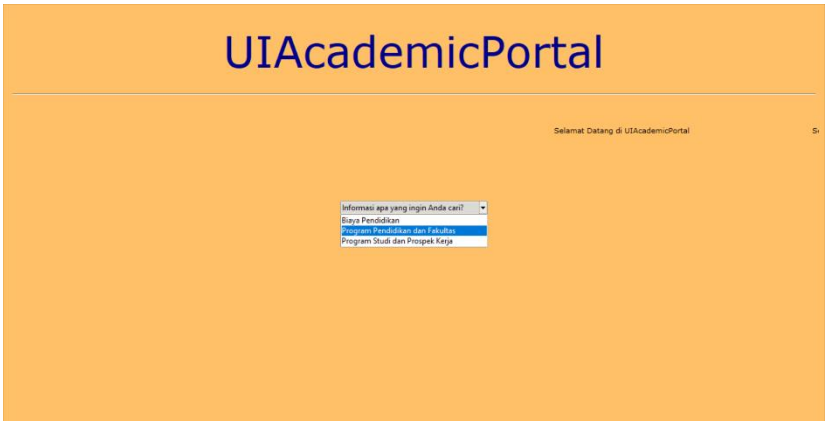
Ketika penulisan salah,



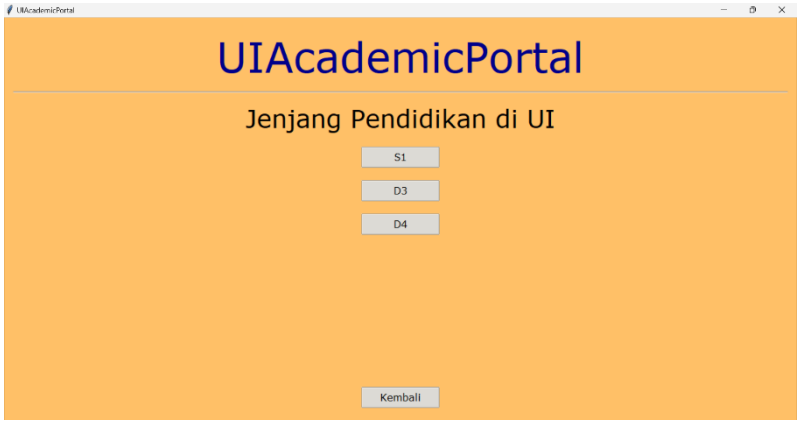
maka akan memberikan output "!! Data Tidak Ditemukan !!"



3. Pilih opsi Program Pendidikan dan Fakultas



Muncul jenjang atau program pendidikan yang ada di UI, yaitu S1, D3, dan D4.



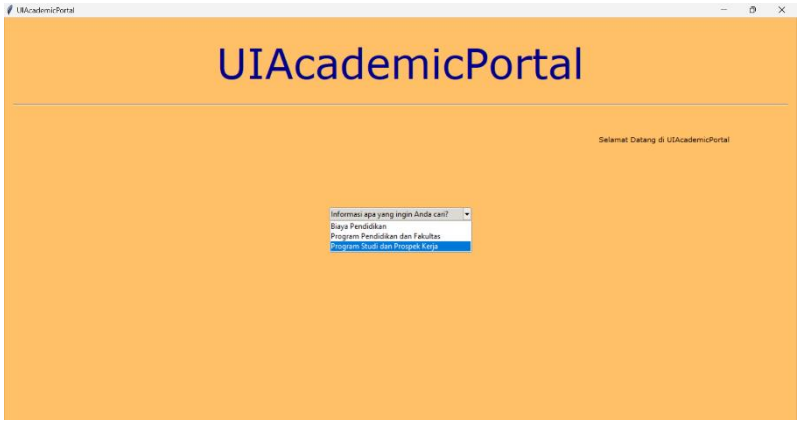
Ketika memilih jenjang S1

Ketika memilih jenjang D3

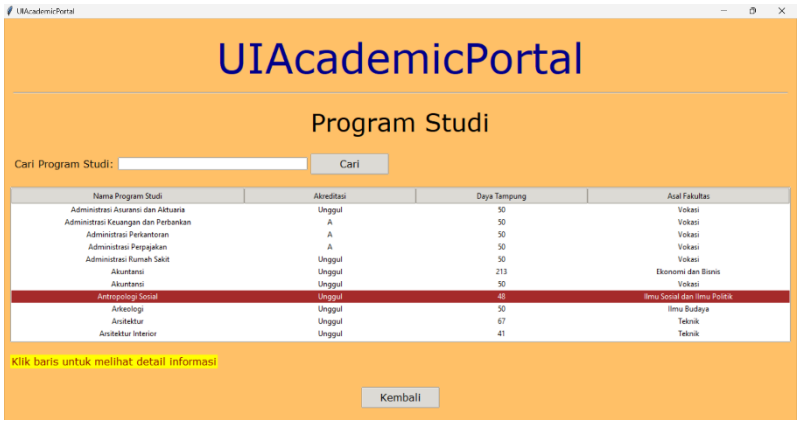
Ketika memilih jenjang D4

Alasan membuat bagian ini karena ingin memberi tahu informasi yang general terkait biaya pendidikan (melalui biaya terendah dan tertinggi) dan Fakultas serta Rumpun yang tersedia di UI.

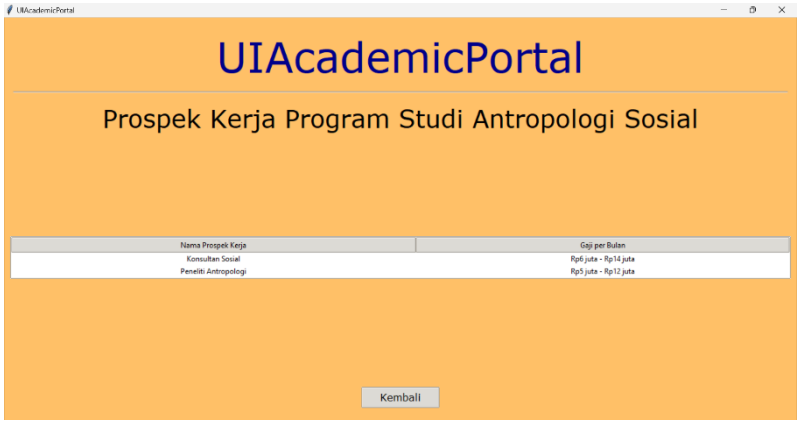
4. Pilih opsi Program Studi dan Prospek Kerja



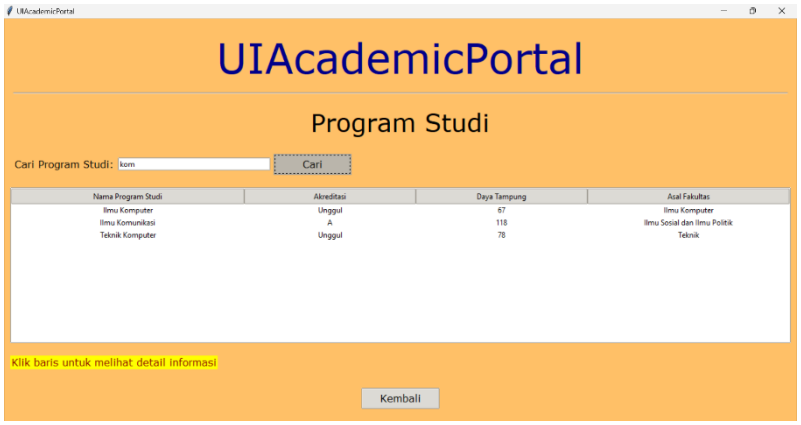
Terdapat 79 program studi yang tersedia di UI. User bisa langsung memilih nama program studi yang diinginkannya.



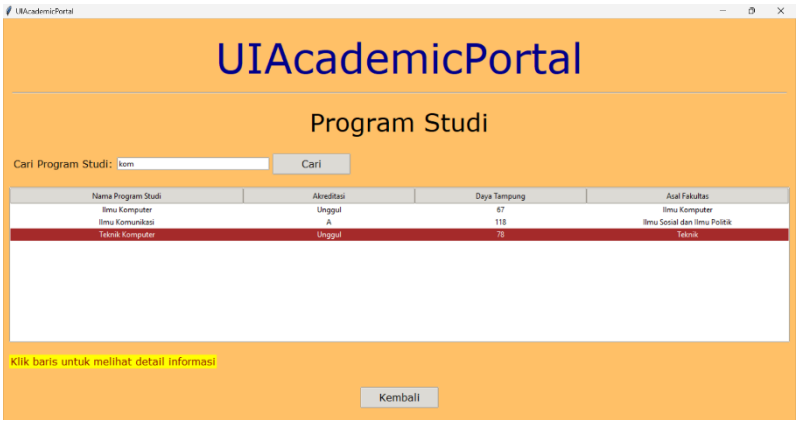
Dengan meng-klik baris yang user inginkan, maka akan langsung muncul informasi tambahan terkait prospek kerja yang sudah sesuai dengan program studinya.



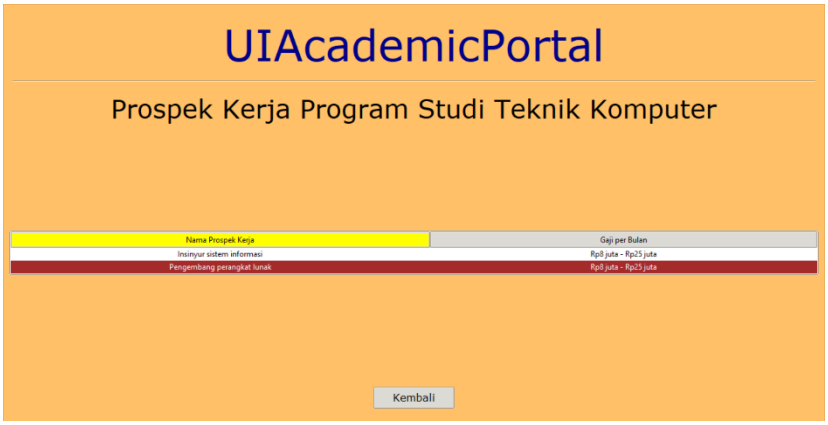
Untuk mempermudah user dalam mencari nama program studi yang diinginkannya, maka kami menyediakan fitur search button. Hal ini juga berguna untuk efektivitas dalam mencari beberapa program studi dari 79 program studi yang tersedia. User cukup mengetikkan “string” nama dari program studinya, maka akan langsung muncul program studi yang namanya mengandung string yang user ketikkan.



User bisa langsung memilih nama program studi yang diinginkannya.



Dengan meng-klik baris yang user inginkan, maka akan langsung muncul informasi tambahan terkait prospek kerja yang sudah sesuai dengan program studinya.



BAB V
KESIMPULAN

Project ini bertujuan untuk mengoptimalkan informasi biaya pendidikan dan prospek kerja untuk tiap program studi bagi calon mahasiswa Universitas Indonesia (UI). Melalui implementasi database relasional, SQL, dan antarmuka grafis (GUI), proyek ini berhasil mencapai tujuan dengan memberikan alat yang mudah digunakan untuk mengakses informasi yang relevan. Dengan detail sebagai berikut:

Struktur *Database*

- ProgramPendidikan
  Tabel ini berisi informasi mengenai jenjang pendidikan yang tersedia, yaitu S1, D3, dan D4. Setiap program pendidikan diidentifikasi dengan ID unik.
- Rumpun
  Tabel ini menyimpan data mengenai rumpun pendidikan di UI seperti Ilmu Kesehatan, Sains dan Teknologi, dan Ilmu Sosial dan Humaniora. Setiap rumpun memiliki ID unik.
- BiayaPendidikan
  Tabel ini berisi informasi tentang golongan biaya pendidikan per semester untuk setiap rumpun pendidikan. Data ini mencakup ID biaya, golongan biaya, dan rentang harga.
- Fakultas
  Tabel ini mencatat informasi tentang fakultas di UI, termasuk nama fakultas, biaya pendidikan tertinggi dan terendah, serta ID rumpun terkait.
- ProgramStudi
  Tabel ini berisi informasi detail tentang program studi yang ditawarkan di UI, termasuk nama program studi, akreditasi, daya tampung, dan ID fakultas terkait.
- ProspekKerja
  Tabel ini menyediakan informasi mengenai prospek kerja yang mungkin bagi lulusan dari setiap program studi, termasuk nama prospek dan rentang gaji per bulan.

Implementasi SQL melalui Python

SQL digunakan untuk:

- Pembuatan Tabel
  Membuat tabel-tabel yang dibutuhkan dalam database berdasarkan desain yang telah ditentukan.
- Pemasukan Data
  Memasukkan data ke dalam tabel-tabel tersebut.
- Query untuk Pencarian Informasi
  Melakukan query untuk mendapatkan informasi yang relevan seperti biaya pendidikan berdasarkan program studi dan jenjang, serta informasi program studi berdasarkan kata kunci.

Implementasi GUI

Antarmuka grafis dibuat menggunakan tkinter untuk memudahkan pengguna dalam mengakses informasi. Fitur utama GUI meliputi:

- Pencarian Biaya Pendidikan
  Pengguna dapat mencari biaya pendidikan berdasarkan nama program studi dengan filter jenjang untuk menghindari redundansi.
- Pencarian Program Studi
  Menggunakan pencarian berbasis string-substring untuk menampilkan program studi yang relevan dengan kata kunci yang dimasukkan.
- Tampilan Informasi Fakultas dan Rumpun
  Menampilkan informasi detail tentang fakultas dan rumpun pendidikan di UI.
- Tampilan Prospek Kerja
  Menampilkan prospek kerja terkait program studi yang dipilih oleh pengguna.

Testing

Pengujian dilakukan untuk memastikan setiap fitur bekerja dengan baik:

- Pencarian Program Studi
  Menghasilkan output yang sesuai dengan kata kunci yang diberikan oleh pengguna.
- Pencarian Biaya Pendidikan
  Menghasilkan output yang sesuai dengan kata kunci dan filter jenjang yang digunakan.
- Interaksi GUI
  Memastikan bahwa pengguna dapat berinteraksi dengan aplikasi secara intuitif dan mendapatkan informasi yang diperlukan dengan mudah.

Project ini berhasil mengembangkan sistem yang efektif untuk mengintegrasikan dan menampilkan informasi biaya pendidikan serta prospek kerja bagi calon mahasiswa UI. Penggunaan database relasional dan antarmuka grafis yang ramah pengguna memastikan bahwa informasi yang diberikan akurat, relevan, dan mudah diakses. Sistem ini juga fleksibel untuk dikembangkan lebih lanjut sesuai dengan kebutuhan pengguna dan perkembangan informasi di UI.

Dengan implementasi yang baik dari struktur database, SQL, dan GUI, proyek ini memberikan kontribusi signifikan dalam memudahkan calon mahasiswa UI dalam mendapatkan informasi yang mereka butuhkan untuk membuat keputusan yang tepat terkait pendidikan dan karier mereka di masa depan.