■ PANDUAN CRUD GPLink - Bahasa Malaysia

• Untuk Capstone Project Presentation**

• -

■ Pengenalan

CRUD bermaksud:

• **C**reate (Buat) - Tambah data baru

• **R**ead (Baca) - Ambil data dari database

• **U**pdate (Kemaskini) - Tukar data yang sudah ada

• **D**elete (Padam) - Buang data dari database

GPLink menggunakan **MongoDB** untuk simpan data consultations (permintaan daripada GP kepada Cardiologist).

• -

■■ File Python Yang Penting

Untuk CRUD Operations:

File	Lokasi	Fungsi
------	--------	--------
`crud.py`	`backend/crud.py`	Semua operasi CRUD (Create, Read, Update, Delete)
`models.py`	`backend/models.py`	Struktur data (bagaimana data disusun)
`main.py`	`backend/main.py`	API endpoints (URL untuk CRUD)

Untuk Frontend (UI):

File	Lokasi	Fungsi
------	--------	--------
`app.py`	`frontend/app.py`	Interface Streamlit (yang user lihat)

• -

■ Struktur Database MongoDB

Collection: consultations

Satu consultation document (satu permintaan) kelihatan macam ini:

```
{
  "_id": ObjectId("507f1f77bcf86cd799439011"),
  "patient": {
    "name": "Ahmad Bin Ali",
    "age": 45,
    "gender": "Male",
    "ic_number": "800101-01-1234",
    "ecg_image": "ecg_507f1f77bcf86cd799439011.png",
    "xray_image": null
  },
  "symptoms": "Sakit dada, sesak nafas",
  "vital_signs": {
    "blood_pressure": "140/90",
    "heart_rate": 95,
    "temperature": 37.5,
    "spo2": 98,
    "respiratory_rate": 20
  },
  "lab_investigations": [
    {
      "test_name": "FBC",
      "date_time": "2025-11-16 09:30",
      "result": "WBC 12.5, RBC 4.8"
    }
  ],
  "assigned_cardiologist_email": "dr.sarah@hospital.com",
  "assigned_cardiologist_name": "Dr. Sarah Lim",
  "clinic_doctor_email": "dr.john@clinic.com",
  "clinic_doctor_name": "Dr. John Wong",
  "urgency": "urgent",
  "status": "pending",
  "created_at": "2025-11-16T09:45:00",
  "diagnosis": null,
  "recommendations": null,
  "cardiologist_notes": null,
  "response_date": null
}
```

- -

1■■ CREATE - Tambah Consultation Baru

Tujuan:

GP ingin membuat permintaan konsultasi baru kepada Cardiologist.

File Yang Digunakan:

• `backend/crud.py` - fungsi `create_consultation()`

• `frontend/app.py` - section "■ New Consultation"

Langkah-langkah (Dalam Code):

```
def create_consultation(consultation_data):
    """
    Fungsi untuk buat consultation baru
    Input: Data consultation dari GP
    Output: ID consultation yang baru dibuat
    """
    # 1. Ambil data dari input
    new_consultation = {
        "patient": {
            "name": consultation_data["patient_name"],
            "age": consultation_data["patient_age"],
            "gender": consultation_data["patient_gender"],
            "ic_number": consultation_data["patient_ic"],
            "ecg_image": consultation_data.get("ecg_image"),
            "xray_image": consultation_data.get("xray_image")
        },
        "symptoms": consultation_data["symptoms"],
        "vital_signs": {
            "blood_pressure": consultation_data["blood_pressure"],
            "heart_rate": consultation_data["heart_rate"],
            "temperature": consultation_data["temperature"],
            "spo2": consultation_data["spo2"],
            "respiratory_rate": consultation_data["respiratory_rate"]
        },
        "lab_investigations": consultation_data.get("lab_investigations", []),
        "assigned_cardiologist_email": consultation_data.get("assigned_cardiologist_email"),
        "clinic_doctor_email": consultation_data["clinic_doctor_email"],
        "clinic_doctor_name": consultation_data["clinic_doctor_name"],
        "urgency": consultation_data["urgency"],
        "status": "pending",  # Status awal
        "created_at": datetime.now()
    }

    # 2. Simpan ke MongoDB
    result = db.consultations.insert_one(new_consultation)

    # 3. Return ID yang baru dibuat
    return str(result.inserted_id)
```

User masukkan:

```
Nama Patient: Ahmad Bin Ali
Umur: 45
Gender: Male
IC/Passport: 800101-01-1234
Symptoms: Sakit dada
Blood Pressure: 140/90
Heart Rate: 95
Temperature: 37.5
SpO2: 98
Respiratory Rate: 20
```

Kemudian click **"■ Submit"** → Frontend hantar ke backend → Backend simpan ke MongoDB.

- -

2■■ READ - Baca Data Consultations

Tujuan:

Cardiologist atau GP ingin lihat senarai consultations.

File Yang Digunakan:

- `backend/crud.py` - fungsi `get_consultations()`
- `frontend/app.py` - section "■ View Consultations"

Langkah-langkah:

```
def get_consultations(status=None):
    """
    Fungsi untuk baca semua consultations
    Input: Status (optional) - "pending", "reviewed", "completed"
    Output: Senarai consultations
    """
    # 1. Buat query (pertanyaan)
    query = {}
    if status:
        query["status"] = status  # Filter by status jika ada

    # 2. Ambil dari MongoDB
    consultations = list(db.consultations.find(query))

    # 3. Return data
    return consultations

def get_consultation_by_id(consultation_id):
    """
    Baca satu consultation menggunakan ID
    """
    from bson import ObjectId
    consultation = db.consultations.find_one(
        {"_id": ObjectId(consultation_id)}
    )
    return consultation
```

```
// Baca semua consultations
db.consultations.find()

// Baca hanya consultation yang pending
db.consultations.find({status: "pending"})

// Baca satu consultation menggunakan ID
db.consultations.findOne({_id: ObjectId("507f1f77bcf86cd799439011")})

// Baca dan display cantik
```

```
db.consultations.find().pretty()
```

• -

3■■ UPDATE - Kemaskini Consultation

Tujuan:

• Cardiologist respond dengan diagnosis

• GP ubah status menjadi "completed"

• GP edit pending consultation

File Yang Digunakan:

• `backend/crud.py` - fungsi `update_consultation()` dan `respond_to_consultation()`

• `frontend/app.py` - section "■ Respond to Consultation" dan "■ Edit"

Langkah-langkah:

```
def respond_to_consultation(consultation_id, cardiologist_email, response_data):
    """
    Cardiologist hantar response dengan diagnosis
    """
    from bson import ObjectId

    # 1. Cari consultation
    consultation = db.consultations.find_one(
        {"_id": ObjectId(consultation_id)}
    )

    # 2. Ambil info cardiologist dari doctors collection
    cardiologist = db.doctors.find_one(
        {"email": cardiologist_email}
    )

    # 3. Update consultation dengan response
    result = db.consultations.update_one(
        {"_id": ObjectId(consultation_id)},
        {
            "$set": {
                "diagnosis": response_data["diagnosis"],
                "recommendations": response_data["recommendations"],
                "cardiologist_notes": response_data["notes"],
                "cardiologist_email": cardiologist_email,
                "cardiologist_name": cardiologist["name"],
                "status": "reviewed",  # Tukar status ke reviewed
                "response_date": datetime.now()
            }
        }
    )

    return result.modified_count > 0
```

```
def update_consultation(consultation_id, updated_data):
    """
    Update consultation yang masih pending
    """
    from bson import ObjectId

    result = db.consultations.update_one(
        {"_id": ObjectId(consultation_id)},
        {
            "$set": {
                "patient.name": updated_data.get("patient_name"),
                "symptoms": updated_data.get("symptoms"),
                "vital_signs": updated_data.get("vital_signs"),
                "urgency": updated_data.get("urgency")
            }
        }
    )

    return result.modified_count > 0
```

```
// Update status consultation menjadi "completed"
db.consultations.updateOne(
  {_id: ObjectId("507f1f77bcf86cd799439011")},
  {
    $set: {
      status: "completed",
      diagnosis: "Arrhythmia",
      recommendations: "Ambil ubat jantung"
    }
  }
)

// Confirm update (lihat record yang di-update)
db.consultations.findOne({_id: ObjectId("507f1f77bcf86cd799439011")})
```

• -

4■■ DELETE - Padam Consultation

Tujuan:

Padam consultation yang tidak perlu lagi.

File Yang Digunakan:

• `backend/crud.py` - fungsi `delete_consultation()`

• `frontend/app.py` - button "■■ Delete"

Langkah-langkah:

```
def delete_consultation(consultation_id):
    """
```

```
        Padam consultation dari database
        """
        from bson import ObjectId

        result = db.consultations.delete_one(
            {"_id": ObjectId(consultation_id)}
        )

        # Return true jika berjaya padam (1 record)
        return result.deleted_count == 1

    def delete_multiple_consultations(consultation_ids):
        """
        Padam banyak consultations sekaligus
        """
        from bson import ObjectId

        ids = [ObjectId(id) for id in consultation_ids]

        result = db.consultations.delete_many(
            {"_id": {"$in": ids}}
        )

        return result.deleted_count
```

```
// Padam satu consultation
db.consultations.deleteOne(
  {_id: ObjectId("507f1f77bcf86cd799439011")}
)

// Padam banyak consultations (status = "completed")
db.consultations.deleteMany(
  {status: "completed"}
)

// Check berapa banyak consultations yang tinggal
db.consultations.countDocuments()
```

- -

■ API Endpoints (URL untuk CRUD)

• Backend** menyediakan URL yang boleh dipanggil:

CREATE:

```
POST /api/consultations
Body: {patient info, symptoms, vital signs, etc.}
Response: {consultation_id: "507f1f77bcf86cd799439011"}
```

READ:

```
GET /api/consultations
GET /api/consultations?status=pending
GET /api/consultations/{id}
Response: Senarai atau satu consultation
```

UPDATE:

```
PUT /api/consultations/{id}
PUT /api/consultations/{id}/respond
Body: {diagnosis, recommendations, notes}
Response: {success: true, message: "Updated"}
```

DELETE:

```
DELETE /api/consultations/{id}
Response: {success: true, message: "Deleted"}
```

- -

■ Demo CRUD (Praktikal)

Cara Test Di Terminal:

Guna MongoDB Compass (GUI) atau mongosh (CLI)

```
mongosh "mongodb+srv://drsy4h:password@cluster0.wqpjziw.mongodb.net/gplink_db"
```

```
db.consultations.insertOne({
  patient: {
    name: "Siti Aminah",
    age: 50,
    gender: "Female",
    ic_number: "750505-01-5678"
  },
  symptoms: "Tekanan tinggi",
  vital_signs: {
    blood_pressure: "150/95",
    heart_rate: 88,
    temperature: 37.2,
    spo2: 99,
    respiratory_rate: 18
  },
  clinic_doctor_email: "dr.john@clinic.com",
  clinic_doctor_name: "Dr. John Wong",
  urgency: "normal",
  status: "pending",
  created_at: new Date()
})


// Lihat semua
db.consultations.find().pretty()

// Cari consultations yang pending
db.consultations.find({status: "pending"}).pretty()

// Cari consultation tertentu
db.consultations.findOne({_id: ObjectId("...")})


db.consultations.updateOne(
  {_id: ObjectId("...")},
  {
    $set: {
      status: "reviewed",
      diagnosis: "Hypertension Stage 2",
      recommendations: "Monitor BP daily, take medication",
      cardiologist_name: "Dr. Sarah Lim",
      cardiologist_email: "dr.sarah@hospital.com"
    }
  }
)


db.consultations.deleteOne({_id: ObjectId("...")})
```

- -

■ Hubungan Antara Files

```
User/GP menggunakan → Frontend (app.py)
                          ↓
                  Streamlit Interface
                          ↓
User klik button → Frontend call API → Backend (main.py)
                          ↓
                  API endpoint terima request
                          ↓
                  CRUD function (crud.py) execute
                          ↓
                  Database Operation (MongoDB)
                          ↓
                  Return hasil ke Frontend
                          ↓
                  User lihat result di UI
```

- -

■ Untuk Capstone Presentation

Soalan Yang Mungkin Ditanya:

• Q: "Bagaimana anda buat consultation baru?"**

A: "Saya gunakan fungsi `create_consultation()` dalam `crud.py`. Function ini ambil data daripada frontend (patient info, symptoms, vital signs), susun dalam format document MongoDB, kemudian insert ke collection `consultations`."

• Q: "Bagaimana data disimpan?"**

A: "Data disimpan dalam MongoDB Atlas cloud database. Setiap consultation adalah satu document dengan struktur JSON yang sudah ditentukan dalam `models.py`."

• Q: "Bagaimana Cardiologist respond?"**

A: "Cardiologist gunakan page '■ Respond to Consultation'. Dia select consultation, masukkan diagnosis dan recommendations. Frontend hantar ke endpoint `PUT /api/consultations/{id}/respond`. Backend function `respond_to_consultation()` update document dengan status berubah menjadi 'reviewed'."

• Q: "Boleh delete consultation?"**

A: "Ya, boleh padam jika diperlukan. Gunakan button 'Delete'. Backend call `delete_consultation()` function yang gunakan MongoDB `deleteOne()` untuk remove document."

• Q: "Bagaimana filter consultation by status?"**

A: "Dalam `get_consultations()` function, saya create MongoDB query dengan filter status. Contoh: `query = {"status": "pending"}` kemudian `find(query)` untuk ambil hanya pending consultations."

• -

■ Ringkasan CRUD

Operasi	Fungsi (crud.py)	API Endpoint	Database Query
---------	------------------	--------------	-----------------
Create	`create_consultation()`	POST /consultations	`insert_one()`
Read	`get_consultations()`	GET /consultations	`find()`
Update	`update_consultation()`	PUT /consultations/{id}	`update_one()`
Delete	`delete_consultation()`	DELETE /consultations/{id}	`delete_one()`

• -

■ Fail Python Penting

• Untuk menunjukkan code:**

1. **`backend/crud.py`** - Tunjukkan fungsi CRUD (Create, Read, Update, Delete)

2. **`backend/models.py`** - Tunjukkan struktur data consultation

3. **`backend/main.py`** - Tunjukkan API endpoints (route yang berbeza untuk CRUD)

4. **`frontend/app.py`** - Tunjukkan bagaimana user interact dengan CRUD

• -

• Semoga membantu untuk capstone presentation! Good luck! ■**