API DOCUMENTATION

PENGEMBANGAN APLIKASI BERBASIS MOBILE

EDUASSES : APLIKASI PENILAIAN KINERJA GURU

DAFTAR ISI

1. **POST '/api/login'**
   - Deskripsi : autentikasi guru agar dapat mengakses aplikasi
   - Request Parameter :
     1. nip (required) : Guru nip.
     2. password (required) : Guru password.
   - Response
     - Success :

```
{
    "token": "login",
    "role": "kepala_sekolah",
    "message": "Login successful"
}
```

```
{
    "token": "login",
    "role": "guru",
    "message": "Login successful"
}
```

     - Error :

```
{
    "message": "Wrong Password"
}
```

   - Contoh Penggunaan :

```
final response = await http.post(
  Uri.parse('http://127.0.0.1:8000/api/login'),
  body: {
    'nip': nip,
    'password': password,
  },
);
```

2. **GET '/api/guru'**
   - Deskripsi : mengambil seluruh data guru dari database
   - Request Parameter : -
   - Response :
     - Success :

```
{
    "message": "Success",
    "data": [
        {
            "id": 1,
            "nip": "1234567",
            "nama": "Kepala Sekolah",
            "pangkat": "IV/a",
            "golongan": "IV",
            "email": "kepsek@gmail.com",
            "password": "kepsek123",
            "created_at": "2024-06-09T22:13:36.000000Z",
            "updated_at": "2024-06-09T22:13:36.000000Z"
        },
        {
            "id": 2,
            "nip": "221511004",
            "nama": "Aulia Aziiah Fauziyyah",
            "pangkat": "III/a",
            "golongan": "III",
            "email": "aulia@gmail.com",
            "password": "aulia123",
            "created_at": "2024-06-09T22:13:36.000000Z",
            "updated_at": "2024-06-09T22:13:36.000000Z"
        },
```


- Contoh Penggunaan :
  http://127.0.0.1:8000/api/guru

```
final response =
    await http.get(Uri.parse('http://127.0.0.1:8000/api/guru'));
```

3. GET '/guru/id'

- Deskripsi : mengambil data guru berdasarkan id tertentu dari database
- Request Parameter :
  1. id (integer, required) : Guru id.
- Response :
  - Success :


```
{
    "message": "Data berhasil ditemukan",
    "data": {
        "id": 1,
        "nip": "1234567",
        "nama": "Kepala Sekolah",
        "pangkat": "IV/a",
        "golongan": "IV",
        "email": "kepsek@gmail.com",
        "password": "kepsek123",
        "created_at": "2024-06-09T22:13:36.000000Z",
        "updated_at": "2024-06-09T22:13:36.000000Z"
    }
}
```


  - Error :


```
{
    "message": "Guru not found"
}
```


- Contoh Penggunaan :

```
var response = await http
    .get(Uri.parse('http://127.0.0.1:8000/api/guru/${widget.id}'));
```

4. GET '/guru/read/{nip}'

- Deskripsi : mengambil data guru berdasarkan nip tertentu dari database
- Request Parameter :
  1. nip (string, required) : Guru nip
- Response :
  - Success :


```
{
    "message": "Data berhasil ditemukan",
    "data": {
        "id": 2,
        "nip": "221511004",
        "nama": "Aulia Aziiah Fauziyyah",
        "pangkat": "III/a",
        "golongan": "III",
        "email": "aulia@gmail.com",
        "password": "aulia123",
        "created_at": "2024-06-09T22:13:36.000000Z",
        "updated_at": "2024-06-09T22:13:36.000000Z"
    }
}
```


  - Error :


```
{
    "message": "Guru not found"
}
```


- Contoh Penggunaan :

```
final response =
    await http.get(Uri.parse('http://127.0.0.1:8000/api/guru/read/$nip'));
```

5. POST '/guru/create'

- Deskripsi : menambahkan data guru ke database
- Request Parameter :
  1. nip (string, required) : Guru nip
  2. nama (string, required) : Guru nama
  3. golongan (string, required) : Guru golongan
  4. pangkat (string, required) : Guru pangkat
  5. email (string, required) : Guru email
  6. password (string, required) : Guru password
- Response :
  - Success :

```
{
    "message": "Data berhasil disimpan",
    "data": {
        "nama": "Aziizah",
        "nip": "221511006",
        "golongan": "III/a",
        "pangkat": "III",
        "email": "aziizah@gmail.com",
        "password": "$2y$12$CjyzQ6daLlomi53qOHY22.YcegZbMQN8vXvqafSRDYB0laPIKjbva",
        "updated_at": "2024-06-10T00:13:03.000000Z",
        "created_at": "2024-06-10T00:13:03.000000Z",
        "id": 5
    }
}
```


- Contoh Penggunaan :

```
String apiUrl = 'http://127.0.0.1:8000/api/guru/create';

try {
  // Kirim permintaan POST ke server
  var response = await http.post(
    Uri.parse(apiUrl),
    body: json.encode(guruData),
    headers: {'Content-Type': 'application/json'},
  );
```

6. PUT '/guru/update/id'

- Deskripsi : melakukan update data guru berdasarkan id
- Request Parameter :
  1. id (integer, required) : Guru id
- Response :
  - Success :


```
{
    "message": "Data berhasil diupdate",
    "data": {
        "id": 1,
        "nip": "1234567",
        "nama": "Aziizah Aulia",
        "pangkat": "IV/a",
        "golongan": "IV",
        "email": "kepsek@gmail.com",
        "password": "kepsek123",
        "created_at": "2024-06-09T22:13:36.000000Z",
        "updated_at": "2024-06-10T00:26:14.000000Z"
    }
}
```


- Contoh Penggunaan :

```
String apiUrl =
    'http://127.0.0.1:8000/api/guru/update/${widget.guru['id']}';
```

7. DELETE '/guru/delete/id'

- Deskripsi : menghapus data guru berdasarkan id

- Request Parameter :
  1. id (integer, required) : Guru id
- Response :
  - Success :

```
{
    "message": "Data berhasil dihapus",
    "data": null
}
```

- Contoh Penggunaan :

```
final apiUrl = 'http://127.0.0.1:8000/api/guru/delete/$id';
final response = await http.delete(
  Uri.parse(apiUrl),
  headers: {'Content-Type': 'application/json'},
);
```

8. GET '/api/penilaian'

- Deskripsi : mengambil data penilaian yang ada dalam database
- Request Parameter : -
- Response
  - Success :

```
{
    "message": "Succes",
    "data": [
        {
            "id": 2,
            "judul_penilaian": "RPS",
            "tgl_penilaian": "2024-06-30 00:00:00",
            "created_at": "2024-06-09T18:47:22.000000Z",
            "updated_at": "2024-06-09T18:47:22.000000Z"
        },
        {
            "id": 3,
            "judul_penilaian": "Program Tahunan",
            "tgl_penilaian": "2024-06-30 00:00:00",
            "created_at": "2024-06-09T19:00:07.000000Z",
            "updated_at": "2024-06-09T19:00:07.000000Z"
        }
    ]
}
```

- Contoh Penggunaan :

```dart
final response =
    await http.get(Uri.parse('http://127.0.0.1:8000/api/penilaian'));

if (response.statusCode == 200) {
  final List<dynamic> data = json.decode(response.body)['data'];
  return data.map((e) => Penilaian.fromJson(e)).toList();
} else {
  throw Exception('Failed to load penilaian');
}
```

9. GET '/api/penilaian/id'

- Deskripsi : mengambil data penilaian sesuai dengan id
- Request Parameter :
  1. id(integer, required) : Penilaian id
- Response
  - Success :

```
{
    "message": "Data berhasil ditemukan",
    "data": {
        "id": 1,
        "judul_penilaian": "RPS",
        "tgl_penilaian": "2024-06-30 00:00:00",
        "created_at": "2024-06-09T18:04:39.000000Z",
        "updated_at": "2024-06-09T18:04:39.000000Z"
    }
}
```

  - Error :

```
{
    "message": "Penilaian not found"
}
```

10. DELETE '/api/penilaian/delete/id'

- Deskripsi : menghapus data penilaian sesuai dengan id
- Request Parameter :
  1. id (integer, required) : Penilaian id
- Response
  - Success :

```
{
    "message": "Data berhasil dihapus",
    "data": null
}
```

11. PUT '/api/penilaian/update/id'

- Deskripsi : mengupdate data penilaian sesuai dengan id
- Request Parameter :

1. id (integer, required) : Penilaian id

- Response
  - Success :


```
{
    "message": "Data berhasil diupdate",
    "data": {
        "id": 2,
        "judul_penilaian": "RPS",
        "tgl_penilaian": "2024-06-30 00:00:00",
        "created_at": "2024-06-09T18:47:22.000000Z",
        "updated_at": "2024-06-09T18:47:22.000000Z"
    }
}
```


12. POST '/api/penilaian/create'

- Deskripsi : membuat data penilaian
- Request Parameter :
  1. judul_penilaia (string, required) : Penilaian judul_penilaian
  2. tgl_penilaian (string, required) : Penilaian tgl_penilaian
- Response
  - Success :


```
{
    "message": "Data berhasil disimpan",
    "data": {
        "judul_penilaian": "Penilaian 7",
        "tgl_penilaian": "2024-10-29",
        "updated_at": "2024-06-10T12:00:13.000000Z",
        "created_at": "2024-06-10T12:00:13.000000Z",
        "id": 6
    }
}
```


- Contoh Penggunaan :

```
String apiUrl = 'http://127.0.0.1:8000/api/penilaian/create';
```

```
var response = await http.post(
  Uri.parse(apiUrl),
  body: json.encode(penilaianData),
  headers: {'Content-Type': 'application/json'},
);
```

13. GET '/api/komponen'

- Deskripsi : mengambil seluruh data komponen yang ada di database
- Request Parameter : -
- Response :
  - Success :

```
{
    "message": "Succes",
    "data": [
        {
            "id": 2,
            "judul_penilaian": "RPS",
            "tgl_penilaian": "2024-06-30 00:00:00",
            "created_at": "2024-06-09T18:47:22.000000Z",
            "updated_at": "2024-06-09T18:47:22.000000Z"
        },
        {
            "id": 3,
            "judul_penilaian": "Program Tahunan",
            "tgl_penilaian": "2024-06-30 00:00:00",
            "created_at": "2024-06-09T19:00:07.000000Z",
            "updated_at": "2024-06-09T19:00:07.000000Z"
        }
    ]
}
```


- Contoh Penggunaan :

```
final response =
    await http.get(Uri.parse('http://127.0.0.1:8000/api/komponen'));
if (response.statusCode == 200) {
  final List<dynamic> data = json.decode(response.body)['data'];
  return List<Map<String, dynamic>>.from(
      data.map((e) => Map<String, dynamic>.from(e))); // List.from
} else {
  throw Exception('Failed to load data: ${response.statusCode}');
}
```

14. GET '/api/komponen/id'

- Deskripsi : mengambil data komponen berdasarkan id
- Request Parameter :
  1. id(integer, required) : Komponen id
- Response
  - Success :


```
{
    "message": "Success",
    "data": [
        {
            "id": 1,
            "id_komponen": 1,
            "pertanyaan": "Jabatan",
            "created_at": "2024-06-09T19:13:05.000000Z",
            "updated_at": "2024-06-09T19:13:05.000000Z"
        }
    ]
}
```


  - Error :


```
{
    "message": "Komponen not found"
}
```

15. POST '/api/komponen/create'

- Deskripsi : menambah data komponen ke database
- Request Parameter :
  1. nama_komponen(string, required) : Komponen nama_komponen
- Response
  - Success :

```
{
    "message": "Data berhasil disimpan",
    "data": {
        "nama_komponen": "AKPG",
        "updated_at": "2024-06-10T00:54:32.000000Z",
        "created_at": "2024-06-10T00:54:32.000000Z",
        "id": 5
    }
}
```

- Contoh Penggunaan :

```
apiUrl = 'http://127.0.0.1:8000/api/komponen/create';
await http.post(
  Uri.parse(apiUrl),
  body: json.encode(komponenData),
  headers: {'Content-Type': 'application/json'},
)
```

16. PUT '/api/komponen/update/id'

- Deskripsi : melakukan update data komponen berdasarkan id
- Request Parameter :
  1. id (integer, required) : Komponen id
- Response
  - Success :

```
{
    "message": "Data berhasil diupdate",
    "data": {
        "id": 1,
        "nama_komponen": "Alat Penilaian Kompetensi Guru",
        "tipe_jawaban": "skor",
        "kesimpulan": 1,
        "created_at": "2024-06-09T19:09:48.000000Z",
        "updated_at": "2024-06-09T19:09:48.000000Z"
    }
}
```

- Contoh Penggunaan :

```
apiUrl =
    'http://127.0.0.1:8000/api/komponen/update/${widget.komponen!['id']}';
```

```
await http.put(
  Uri.parse(apiUrl),
  body: json.encode(komponenData),
  headers: {'Content-Type': 'application/json'},
);
```

17. DELETE '/api/komponen/delete/id'

- Deskripsi : menghapus data komponen berdasarkan id
- Request Parameter :
  1. id (integer, required) : Komponen id
- Response
  - Success :

```
{
    "message": "Data berhasil dihapus",
    "data": null
}
```

- Contoh Penggunaan :

```
String apiUrl = 'http://127.0.0.1:8000/api/komponen/delete/$id';
```

```
var response = await http.delete(
  Uri.parse(apiUrl),
  headers: {'Content-Type': 'application/json'},
);
```

18. GET 'api/pertanyaan'

- Deskripsi : menampilkan seluruh data pertanyaan yang ada di database
- Request Parameter : -
- Response
  - Success :

```
{
    "message": "Success",
    "data": [
        {
            "id": 1,
            "id_komponen": 1,
            "pertanyaan": "Kalender Pendidikan?",
            "created_at": "2024-06-10T01:58:16.000000Z",
            "updated_at": "2024-06-10T01:58:16.000000Z"
        },
        {
            "id": 2,
            "id_komponen": 2,
            "pertanyaan": "Analisi KKM?",
            "created_at": "2024-06-10T01:58:16.000000Z",
            "updated_at": "2024-06-10T01:58:16.000000Z"
        },
        {
            "id": 3,
            "id_komponen": 3,
            "pertanyaan": "Daftar Nilai?",
            "created_at": "2024-06-10T01:58:16.000000Z",
            "updated_at": "2024-06-10T01:58:16.000000Z"
        },
```


- Contoh Penggunaan :

```dart
final response =
    await http.get(Uri.parse('http://127.0.0.1:8000/api/komponen'));
if (response.statusCode == 200) {
  final List<dynamic> data = json.decode(response.body)['data'];
  return List<Map<String, dynamic>>.from(
      data.map((e) => Map<String, dynamic>.from(e))); // List.from
} else {
  throw Exception('Failed to load data: ${response.statusCode}');
}
```

19. GET '/api/pertanyaan/{id_komponen}'

- Deskripsi : mengambil data pertanyaan berdasarkan id komponen yang sesuai
- Request Parameter :
  1. id_komponen (integer, required) : Pertanyaan (id_komponen)
- Response
  - Success :

```
{
    "message": "Success",
    "data": [
        {
            "id": 1,
            "id_komponen": 1,
            "pertanyaan": "Kalender Pendidikan?",
            "created_at": "2024-06-10T01:58:16.000000Z",
            "updated_at": "2024-06-10T01:58:16.000000Z"
        },
        {
            "id": 4,
            "id_komponen": 1,
            "pertanyaan": "RPP?",
            "created_at": "2024-06-10T03:00:31.000000Z",
            "updated_at": "2024-06-10T03:00:31.000000Z"
        },
        {
            "id": 5,
            "id_komponen": 1,
            "pertanyaan": "Silabus",
            "created_at": "2024-06-10T03:01:01.000000Z",
            "updated_at": "2024-06-10T03:01:01.000000Z"
        }
    ]
```


➢ Error :


```
{
    "message": "No questions found for this component"
}
```


- Contoh Penggunaan :

```dart
final response = await http.get(
  Uri.parse(
    'http://127.0.0.1:8000/api/pertanyaan/$idKomponen',
  ),
);

if (response.statusCode == 200) {
  final List<dynamic> data = json.decode(response.body)['data'];
  return List<Map<String, dynamic>>.from(
    data.map((e) => Map<String, dynamic>.from(e)),
  ); // List.from
} else {
  throw Exception('Failed to load data: ${response.reasonPhrase}');
}
```

20. GET '/api/pertanyaan/read/id'

- Deskripsi : mengambil data pertanyaan berdasarkan id pertanyaan
- Request Parameter :
  1. id (integer, required) : Pertanyaan id
- Response
  - ➢ Success :

```
{
    "message": "Data berhasil ditemukan",
    "data": {
        "id": 1,
        "id_komponen": 1,
        "pertanyaan": "Kalender Pendidikan?",
        "created_at": "2024-06-10T01:58:16.000000Z",
        "updated_at": "2024-06-10T01:58:16.000000Z"
    }
}
```


➢ Error :


```
{
    "message": "Pertanyaan not found"
}
```


- Contoh Penggunaan :

```
final response = await http.get(
    Uri.parse('http://127.0.0.1:8000/api/pertanyaan/read/$idPertanyaan'));

if (response.statusCode == 200) {
  final Map<String, dynamic> responseData = json.decode(response.body);
  if (responseData['data'] != null && responseData['data'].isNotEmpty) {
    return responseData['data']['pertanyaan'];
  } else {
    throw ('Pertanyaan tidak ditemukan');
  }
} else {
  throw ('Gagal mendapatkan pertanyaan');
}
```

21. POST '/api/pertanyaan/create'

- Deskripsi : menambahkan data pertanyaan ke dalam database
- Request Parameter :
  1. id_komponen (integer, required) : Pertanyaan id_komponen
  2. pertanyaan (string, required) : Pertanyaan pertanyaan
- Response
  - ➢ Success :


```
{
    "message": "Data berhasil disimpan",
    "data": {
        "id_komponen": "1",
        "pertanyaan": "Nilai?",
        "updated_at": "2024-06-10T12:13:40.000000Z",
        "created_at": "2024-06-10T12:13:40.000000Z",
        "id": 7
    }
}
```


- Contoh Penggunaan :

```
final response = await http.get(Uri.parse('http://127.0.0.1:8000/api/pertanyaan/create'));
```

```
await http.post(
  Uri.parse(apiUrl),
  body: json.encode(pertanyaanData),
  headers: {'Content-Type': 'application/json'},
)
```

22. PUT '/api/pertanyaan/update/id'

- Deskripsi : melakukan update data pertanyaan berdasarkan id
- Request Parameter :
  1. id (integer, required) : Pertanyaan id
- Response
  - Success :


```
{
    "message": "Data berhasil diupdate",
    "data": {
        "id": 1,
        "id_komponen": 1,
        "pertanyaan": "RPS?",
        "created_at": "2024-06-10T01:58:16.000000Z",
        "updated_at": "2024-06-10T12:22:10.000000Z"
    }
}
```


- Contoh Penggunaan :

```
apiUrl =
    'http://127.0.0.1:8000/api/pertanyaan/update/${widget.pertanyaan!['id']}';
```

```
await http.put(
  Uri.parse(apiUrl),
  body: json.encode(pertanyaanData),
  headers: {'Content-Type': 'application/json'},
);
```

23. DELETE '/api/pertanyaan/delete/id'

- Deskripsi : menghapus data pertanyaan berdasarkan id
- Request Parameter :
  1. id (integer, required) : Pertanyaan id
- Response
  - Success :


```
{
    "message": "Data berhasil dihapus",
    "data": null
}
```


- Contoh Penggunaan :

```
String apiUrl = 'http://127.0.0.1:8000/api/pertanyaan/delete/$pertanyaanId';
```

```
var response = await http.delete(Uri.parse(apiUrl));
```

24. GET '/api/jawaban'

- Deskripsi : mengambil seluruh data jawaban dari database
- Request Parameter : -
- Response
  - Success :


```
{
    "message": "Success",
    "data": [
        {
            "id": 1,
            "id_penilaian": 1,
            "id_guru": 2,
            "id_komponen": 1,
            "id_pertanyaan": 1,
            "skor": 3,
            "ketersediaan": 1,
            "keterangan": "hai",
            "created_at": "2024-06-10T02:28:29.000000Z",
            "updated_at": "2024-06-10T02:28:29.000000Z"
        },
        {
            "id": 2,
            "id_penilaian": 1,
            "id_guru": 3,
            "id_komponen": 1,
            "id_pertanyaan": 1,
            "skor": 5,
            "ketersediaan": 1,
            "keterangan": "lengkap",
            "created_at": "2024-06-10T03:02:22.000000Z",
            "updated_at": "2024-06-10T03:02:22.000000Z"
        },
```


25. GET '/api/jawaban/{id_penilaian}/{id_guru}/{id_komponen}'

- Deskripsi : mengambil data pertanyaan berdasarkan id_penilaian, id_guru, dan id_komponen yang sama
- Request Parameter :
  1. id_penilaian (integer, required) : Jawaban id_penilaian
  2. id_guru (integer, required) : Jawaban id_guru
  3. id_komponen (integer, required) : Jawaban id_komponen
- Response
  - Success :

```
{
    "success": true,
    "message": "Data jawaban ditemukan",
    "data": [
        {
            "id": 2,
            "id_penilaian": 1,
            "id_guru": 3,
            "id_komponen": 1,
            "id_pertanyaan": 1,
            "skor": 5,
            "ketersediaan": 1,
            "keterangan": "lengkap",
            "created_at": "2024-06-10T03:02:22.000000Z",
            "updated_at": "2024-06-10T03:02:22.000000Z"
        },
        {
            "id": 3,
            "id_penilaian": 1,
            "id_guru": 3,
            "id_komponen": 1,
            "id_pertanyaan": 4,
            "skor": 4,
            "ketersediaan": 1,
            "keterangan": "kelengkapan kurang",
            "created_at": "2024-06-10T03:02:23.000000Z",
            "updated_at": "2024-06-10T03:02:23.000000Z"
        },
```


➢ Error :


```
{
    "success": false,
    "message": "Data jawaban tidak ditemukan",
    "data": null
}
```


- Contoh Penggunaan :

```
final response = await http.get(Uri.parse(
    'http://127.0.0.1:8000/api/jawaban/${widget.idPenilaian}/${widget.idGuru}/${widget.idKomponen}'));

if (response.statusCode == 200) {
  final Map<String, dynamic> responseData = json.decode(response.body);
  return List<Map<String, dynamic>>.from(responseData['data']);
} else {
  throw ('Penilaian Belum Dilakukan');
}
```

26. GET '/api/jawaban/{id_penilaian}/{id_guru}/{id_komponen}/{id_pertanyaan}'

- Deskripsi : mengambil data pertanyaan berdasarkan id_penilaian, id_guru, id_komponen, dan id_pertanyaan yang sama
- Request Parameter :
  1. id_penilaian (integer, required) : Jawaban id_penilaian
  2. id_guru (integer, required) : Jawaban id_guru
  3. id_komponen (integer, required) : Jawaban id_komponen
  4. id_pertanyaan (integer, required) : Jawaban id_pertanyaan
- Response
  ➢ Success :

```
{
    "success": true,
    "message": "Data jawaban ditemukan",
    "data": {
        "id": 2,
        "id_penilaian": 1,
        "id_guru": 3,
        "id_komponen": 1,
        "id_pertanyaan": 1,
        "skor": 5,
        "ketersediaan": 1,
        "keterangan": "lengkap",
        "created_at": "2024-06-10T03:02:22.000000Z",
        "updated_at": "2024-06-10T03:02:22.000000Z"
    }
}
```


➢ Error :


```
{
    "success": false,
    "message": "Data jawaban tidak ditemukan",
    "data": null
}
```


- Contoh Penggunaan :

```
final response = await http.get(Uri.parse(
    'http://127.0.0.1:8000/api/jawaban/${widget.idPenilaian}/${widget.idGuru}/${widget.idKomponen}/$idPertanyaan'));

if (response.statusCode == 200) {
  final Map<String, dynamic> responseData = json.decode(response.body);
  return responseData['data'];
} else {
  throw ('Gagal mendapatkan jawaban dan skor');
}
```

27. POST '/api/jawaban/create'

- Deskripsi : menambahkan data jawaban ke dalam database
- Request Parameter :
  1. id_penilaian (integer, required) : Jawaban id_penilaian
  2. id_guru (integer, required) : Jawaban id_guru
  3. id_komponen (integer, required) : Jawaban id_komponen
  4. id_pertanyaan (integer, required) : Jawaban id_pertanyaan
  5. skor (integer, required) : Jawaban skor
  6. ketersediaan (integer, required) : Jawaban ketersediaan
  7. keterangan (string, required) : Jawaban keterangan
- Response
  - ➢ Success :

```
{
    "message": "Data berhasil disimpan",
    "data": {
        "id_penilaian": "4",
        "id_guru": "3",
        "id_komponen": "1",
        "id_pertanyaan": "1",
        "skor": "5",
        "ketersediaan": "1",
        "keterangan": "lengkap",
        "updated_at": "2024-06-10T12:38:01.000000Z",
        "created_at": "2024-06-10T12:38:01.000000Z",
        "id": 8
    }
}
```


- Contoh Penggunaan :

```
final response = await http.post(
  Uri.parse('http://127.0.0.1:8000/api/jawaban/create'),
  headers: {'Content-Type': 'application/json'},
  body: json.encode({
    'id_penilaian': widget.idPenilaian,
    'id_guru': widget.idGuru,
    'id_komponen': widget.komponenId,
    'id_pertanyaan': idPertanyaan,
    'skor': skor,
    'ketersediaan': ketersediaan,
    'keterangan': keterangan,
  }),
);
```