Counsellors API — MindEase

Audience : Frontend developer (Tejas)
Base URL : `http://localhost:8000` (replace with production base in prod)

Endpoint (search & list) :
`GET /api/counsellors/`

Auth : *Public* — clients can call without authentication.

Purpose

Allow clients to **search and browse counsellors** by :

- name (partial, case-insensitive),
- specialization (one or many names),
- fees / price range.

Response returns a paginated list of counsellors with the following data for each counsellor :

- `id`, `name` (first + last), `profile_picture`, `specializations` (list of names), `fees_per_session` (decimal string), `summary`/`bio`.

Query parameters (all optional)

- `q` — search by counsellor name (partial, case-insensitive). Example: `q=paras`
- `specialization` — filter by specialization **name(s)**. Can appear multiple times or be comma-separated.
    - Example single: `specialization=Anxiety`

    - Example multi: `specialization=Anxiety&specialization=Depression` or `specialization=Anxiety,Depression`

- `min_fee` — minimum fees_per_session (inclusive). Example: `min_fee=500`
- `max_fee` — maximum fees_per_session (inclusive). Example: `max_fee=2000`
- `page` — page number (1-indexed). Default: `1`.
- `page_size` — items per page. Default: `10`. Max allowed: `20`.
- `sort` — optional sorting:
    - `fee_asc` — fees low → high
    - `fee_desc` — fees high → low
    - `name_asc` — name A→Z
    - `name_desc` — name Z→A
      Default: server-defined (e.g., relevance or newest).

Notes

- `specialization` and `availability` use **names**, not IDs.
- If a provided specialization name does not exist, the server returns an empty result (not an error).
- Passing invalid numeric parameters (non-numeric `min_fee`, `page` etc.) will return `400 Bad Requests` with a short error message.

Response format (list)

Status : `200 OK`
JSON

{

```
 "count": 123,
 "page": 1,
 "page_size": 10,
 "next": "http://localhost:8000/api/counsellors/?page=2&page_size=10",
 "previous": null,
 "results": [
  {
   "id": 42,
   "name": "Pooja Shah",
   "profile_picture": "https://cdn.example.com/avatars/42.png",
   "specializations": ["Anxiety", "Relationship Counselling"],
   "fees_per_session": "1500.00",
   "bio": "Clinical counsellor specialising in anxiety and relationships.",
   "availability": ["weekdays", "evenings"]
  },
  {
   "id": 43,
   "name": "Amit Kumar",
   "profile_picture": null,
   "specializations": ["Stress Management"],
   "fees_per_session": "1200.00",
   "bio": "CBT practitioner",
   "availability": ["weekends"]
  }
  // ...
 ]
}
```


Response format (single counsellor)


```
{
 "id": 42,
 "name": "Pooja Shah",
 "profile_picture": "https://cdn.example.com/avatars/42.png",
 "specializations": ["Anxiety", "Relationship Counselling"],
 "fees_per_session": "1500.00",
 "bio": "Clinical counsellor specialising in anxiety and relationships.",
 "availability": ["weekdays", "evenings"],
 "experience": "10 years private practice"
}
```

Example queries & expected behavior

Search by name only

GET /api/counsellors/?q=pooja

1.
    - Returns counsellors whose first or last name contains `pooja` (case-insensitive).

Filter by specialization only

GET /api/counsellors/?specialization=Anxiety

2.
    - Returns counsellors who have `anxiety` in their specializations.

Filter by price range only

GET /api/counsellors/?min_fee=1000&max_fee=2000

3.
    - Returns counsellors with `fees_per_session` between 1000 and 2000 inclusive.

Any two filters (e.g., name + specialization)

GET /api/counsellors/?q=amit&specialization=Stress%20Management

4.

All filters (combined)

GET /api/counsellors/?q=pooja&specialization=Anxiety,Depression&min_fee=1000&max_fee=2000&page=1&page_size=10&sort=fee_asc

5. **Pagination**
    - `page=2&page_size=5` returns a second page with 5 items per page.

Error cases

`400 Bad Request` — invalid parameter (e.g., non-int `page`, non-number `min_fee`):

{ "detail": "Invalid page parameter." }

Frontend examples — browser console (copy/paste)

> These use `fetch` (no credentials required since API is public). Replace `localhost:8000` with production base when needed.

- Name search:

```
fetch("http://localhost:8000/api/counsellors/?q=pooja")
  .then(r => r.json()).then(console.log);
```

- Specialization only (single):

```
fetch("http://localhost:8000/api/counsellors/?specialization=Anxiety")
  .then(r => r.json()).then(console.log);
```

- Specialization multiple (comma separated):

```
fetch("http://localhost:8000/api/counsellors/?specialization=Anxiety,Depression")
  .then(r => r.json()).then(console.log);
```

- Price range + pagination:

```
fetch("http://localhost:8000/api/counsellors/?min_fee=800&max_fee=1500&page=1&page_size=10&sort=fee_asc")
  .then(r => r.json()).then(console.log);
```

- Combine name + specialization + availability:

```
fetch("http://localhost:8000/api/counsellors/?q=amit&specialization=Stress%20Management&availability=weekends")
  .then(r => r.json()).then(console.log);
```

Pagination best-practices for frontend

- Default `page_size=10` is reasonable. Use `page` & `page_size` to fetch more results.
- Show `count` to allow UI to compute total pages.
- Use `next` / `previous` values in response to navigate pages — they are full URLs already encoded.