You need to create the **backend** of a GuestBook as attached in the mockup. **You don’t have to build the frontend.**

There should be two new tables:

* User (fields: Name, Created Date)
* Entry (fields: Subject, Message, Created Date, Fk: User).

Each unique name should create a new user in the table.

**The assignment should have 3 different APIs:**

* **Create entry: Allowing the end user to add an entry by providing a name, message, and subject.**
  + **For each new unique name, there should be a new user created in the model.**
* **Get entries: Return a list of all GuestBook’s latest entries.**
  + **Pagination: 3 items per page**
  + **Order by: Created date descending.**
  + **The name also should be provided as in the example.**
* **Get users’ data: Displaying the data according to the users as explained. For each user:**
  + **total count of messages**
  + **the subject of user’s last entry | message of user’s last entry (as a whole string, divided by ‘|’**
  + **DO NOT USE PAGINATION**

**Please OPTIMISE queries as much as possible, and keep in mind that data can get relatively big. Advance ORM usage will be rewarded additionally.**

**Writing an end-to-end test will be rewarded additionally.**

**Writing comments and clean code will be rewarded additionally.**

**Feel free to use any database.**

Get entries response example:

{

  "count": 3,

  "page\_size": 3,

  "total\_pages": 1,

  "current\_page\_number": 1,

  "links": {

    "next": **null**,

    "previous": **null**

  },

  "entries": [

    {

      "user": <user>,

      "subject": <subject>,

      "message": <message>,

    },

    {

      "user": <user>,

      "subject": <subject>,

      "message": <message>,

    },

    ...

  ]

}

Get users response example:

{

    "users": [

        {

            "username": "user\_1",

            "last\_entry": "subject\_3 | message\_3"

        } ,

        {

            "username": "user\_2",

            "last\_entry": "subject\_9 | message\_9"

        }

    ]

}