Back-end Golang

==========

1. Simple Database querying

==========

Terdapat sebuah table "USER" yg memiliki 3 kolom: ID, UserName, Parent. Di mana:

Kolom ID adalah Primary Key

Kolom UserName adalah Nama User

Kolom Parent adalah ID dari User yang menjadi Creator untuk User tertentu.

eg.

_____

| ID | UserName | Parent |

_____

| 1 | Ali | 2 |

| 2 | Budi | 0 |

| 3 | Cecep | 1 |

_____-

Tuliskan SQL Query untuk mendapatkan data berisi:

_____

| ID | UserName | ParentUserName |

_____

| 1 | Ali | Budi |

| 2 | Budi | NULL |

| 3 | Cecep | Ali |

_____

*Kolom ParentUserName adalah UserName berdasarkan value Parent

==========

2. Please write a microservice to search movies from http://www.omdbapi.com/

The microservice should be able to handle two transports : REST JSON HTTP and GRPC

Access credentials :

OMDBKey : "faf7e5bb&s"

URL : http://www.omdbapi.com/

* Example url call to search is --> GET http://www.omdbapi.com/?apikey=faf7e5bb&s=Batman&page=2

Functions to be implemented are :

- Search with pagination --> 2 parameters : "pagination" and "searchword"

- Get single detail of the movie

- Log each search calls to a dummy DB eg. let's just say we have a MySQL DB table for this.

Important aspects :

- Readability of code

- Good display on the knowledge of "Separation of Concerns for Codes"
- Write unit tests on some of the important files. (For Bigger plus points see below)
- Good use of asynchronousy with Go-Routine

Plus points:

- Implementation of Clean Architecture is a BIG plus
- Complete Unit tests on all codes

==========

3. Please refactor the code below to make it more concise, efficient and readable with good logic flow.

```
func findFirstStringInBracket(str string) string {
	if (len(str) > 0) {
		indexFirstBracketFound := strings.Index(str,"(")
		if indexFirstBracketFound >= 0 {
			runes := []rune(str)
			wordsAfterFirstBracket := string(runes[indexFirstBracketFound:len(str)])
			indexClosingBracketFound := strings.Index(wordsAfterFirstBracket,")")
			if indexClosingBracketFound >= 0 {
				runes := []rune(wordsAfterFirstBracket)
				return string(runes[1:indexClosingBracketFound-1])
			}else{
				return ""
			}
		}else{
	  return ""
	}
		}else{
			return ""
		}
		return ""
}
```

==========

4. Logic Test

Anagram adalah istilah dimana suatu string jika dibolak balik ordernya maka akan sama eg. 'aku' dan 'kua' adalah Anagram, 'aku' dan 'aka' bukan Anagram.

Dibawah ini ada array berisi sederetan Strings.

['kita', 'atik', 'tika', 'aku', 'kia', 'makan', 'kua']

Silahkan kelompokkan/group kata-kata di dalamnya sesuai dengan kelompok Anagramnya,

Expected Outputs

[

```
  ["kita", "atik", "tika"],
  ["aku", "kua"],
  ["makan"],
  ["kia"]
]
```