Problem Statement 1

1. Create an REST API server in **Golang** which implements the endpoints mentioned below.
2. Containize the application using Docker and host the container image in DockerHub (optional, preferred)
3. Push the code to GitHub and submit the URL of the git repository.

REST API server
The server acts as a backend to a note taking application. The REST server implements the following endpoints.

Description	HTTP Method & URL	Request	Response

Endpoint for creating new user	`[POST]`
	`–-----`
	`/signup`
		`{`
		`“name”: <string>,`
		`“email”: <string>,`
		`“Password”: <string>`
		`}`
			`200 OK`
			(on success)
			`400 Bad Request`
			(if request format is invalid)

Endpoint for login	`[POST]`
	`–-----`
	`/login`
		`{`
		`“email”: <string>,`
		`“Password”: <string>`
		`}`
			`200 OK`
			`{`
			`“sid”: <string>`
			`}`
			(“sid” is session_id which is unique for each user login)
			`–---------------`
			`400 Bad Request`
			(if request format is invalid)
			`401 Unauthorized`
			(if username and password doesn’t match)

Endpoint for listing all the notes created by an user.	[GET]
	------ 	
	/notes
		{
		  “sid”: <string>,
		}
			200 OK
			
			{
			  “notes”: [
			    {
			      “id”: <uint32>,
			      “note”: <string>
			    },
			    {
			      “id”: <uint32>,
			      “note”: <string>
			    }
			    {
			      “id”: <uint32>,
			      “note”: <string>
			    }
			  ]
			}
			
			---------------- 	
			400 Bad Request
			(if request format is invalid)
			
			401 Unauthorized
			(if “sid” is invalid)

Endpoint for creating a new note.	[POST]
	------ 	
	/notes
		{
		  “sid”: <string>,
		  “note”: <string>
		}
			200 OK
			
			{
			  “id”: <uint32>
			}
			
			(“id” of the newly created note)
			
			---------------- 	
			400 Bad Request
			(if request format is invalid)
			
			401 Unauthorized
			(if “sid” is invalid)

Endpoint for deleting a note.	[DELETE]
	------ 	
	/notes
		{
		“sid”: <string>,
		“id”: <uint32>
		}
			200 OK
			(on success)
			
			400 Bad Request
			(if request format or “id” is invalid)
			
			401 Unauthorized
			(if “sid” is invalid)

Problem Statement 2

Explain the following code snippet. Explain what the code is attempting to do? You can explain by: Giving use-cases of what this construct/pattern could be used for?

```
package main

import "fmt"

func main() {
    cnp := make(chan func(), 10)
    for i := 0; i < 4; i++ {
        go func() {
            for f := range cnp {
                f()
            }
        }()
    }
    cnp <- func() {
        fmt.Println("HERE1")
    }
    fmt.Println("Hello")
}
```