hangman-web

Hangman-web consists in creating and running a server, in which it will be possible to use a web GUI (graphical user interface) version of your last project, hangman-classic.

Go Module

- [Go Module] (https://golang.org/doc/tutorial/create-module) You must use a go module to call your functions from hangman-classic. You should not copy/paste or rewrite them in this new hangman-web project.

You will need to create a private repository with the name `hangman-web`

Notions

- Golang Documentation: net
- Golang Documentation: ioutil
- Golang Documentation: rand
- Go Web Example Documentation: templates
- Golang Documentation: templates

Objectives

Create a program `hangman-web` that will take a file as parameter `words.txt`. Create a file `words.txt` which contains a bunch of words with which the program will play. Each word is separated with a newline.

The behavior of the game is the same as the hangman project, refers to it for more details.

In this project you will need to implement at least the following endpoints:

1. GET `/` : Sends HTML response - the main page, it will basically display your interface.
   1.1. GET Tip: go templates to receive and display data from the server
2. POST `/hangman` : that sends data to the Golang server (the letter you want to find)
   2.1. POST Tip: use form and other types of tags to make the post request. The form must redirect to `/hangman`

The main page must have at least: * A text representing the word to reveal. * A text input * A button which sends a POST request to `/hangman` and outputs the result on page.

Allowed packages

- Only the standard go packages are allowed
- No use **Framework HTML/CSS**

Instructions

- HTTP server must be written in *Go*.
- HTML templates must be in project root directory *templates*.
- The code must respect the good practices.
- Use [Part 2 of hangman-classic] (https://lyon-ynov-campus.github.io/YTrack/subjects/hangman/hangman-classic/).

Usage

- Image