Relazione Progetto Programmazione di Reti

Paulo Kravchuk - 0001122761

June 26, 2025

1 Introduzione

Per questo progetto è stata scelta la traccia numero uno. È stato realizzato un server HTTP multithread che soddisfa sia i requisiti minimi sia alcune estensioni opzionali. Il server fornisce al client quattro pagine HTML: una pagina principale, due pagine dedicate a minigiochi, una di errore 404.

2 Struttura del Progetto

- `server.py`: implementazione server HTTP multithread
- `www/`: root directory
  - `index.html`: pagina principale / landing page
  - `404/`: pagina errore
  - `square/`: pagina minigioco 1
  - `guess/`: pagina minigioco 2
  - `images/`: directory contenente immagini come favicon
  - `css/`: directory dei file di stile CSS
  - `js/`: directory dei file JavaScript

3 Funzionalità Implementate

- Gestione richieste su thread multipli;
- Gestione delle richieste HTTP GET e blocco delle altre;
- Directory routing (/minigame a /minigame.index.html);
- Pagina di errore 404 personalizzata;
- Logging lato server delle richieste ricevute;
- Gestione dei MIME types;
- Pagine responsive con diverse animazioni;

4 Architettura del Server

Sono state utilizzate librerie di supporto come: `socket`, `threading`, `os`, `sys`, `signal`, `mimetypes`, `http`, `datetime`, `urllib.parse`.

Il server apre un socket TCP in ascolto sulla porta `8080` e accetta connessioni ad esso in loop. Ogni nuova connessione viene gestita da un nuovo thread attraverso la libreria `threading`. Il server si interrompe in caso di errori o interrupt signal come `Ctrl + C`.

Il routing e la gestione dei file viene eseguita normalizzando il path richiesto e associandolo ad un file in `www/`. Se il path corrisponde a una directory, viene cercato all'interno della essa un corrispondente `index.html`. Altrimenti, il path non porta da nessuna parte e quindi viene servita una pagina 404.

Ogni richiesta indipendentemente dallo stato è sottoposta a logging da parte del server in console, dove vengono specificati: indirizzo ip, data, prima riga della richiesta (metodo, path, versione), status code e status message.

Vengono accettate solo richieste di tipo GET, le altre vengono bloccate dato che non sono stati implementati gestori per esse.

Sono stati implementati diversi safecheck in modo che, in caso di fallimenti, la richiesta venga gestita in modo opportuno: rimozione query parameters, normalizzazione del path, fallback body se la pagina 404 viene trovata, etc.

5 Pagine Statiche

In totale vengono servite quattro pagine statiche: una pagina principale (landing page) che funge da punto di accesso alle altre, due pagine dedicate ai minigiochi accessibili cliccando uno dei due lati della homepage, e una pagina di errore 404. Quest'ultima può essere visualizzata cliccando un apposito pulsante nella pagina principale oppure digitando un percorso non valido. Il tutto è stilizzato con CSS e reso funzionante grazie a JavaScript.

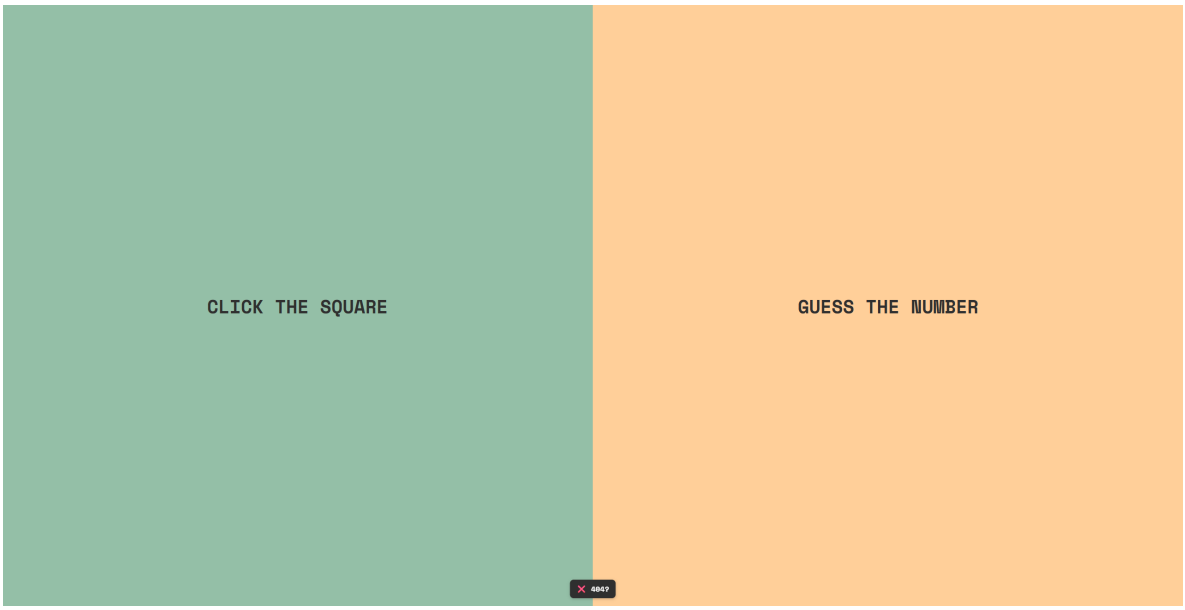

Figure 1: Default mainpage

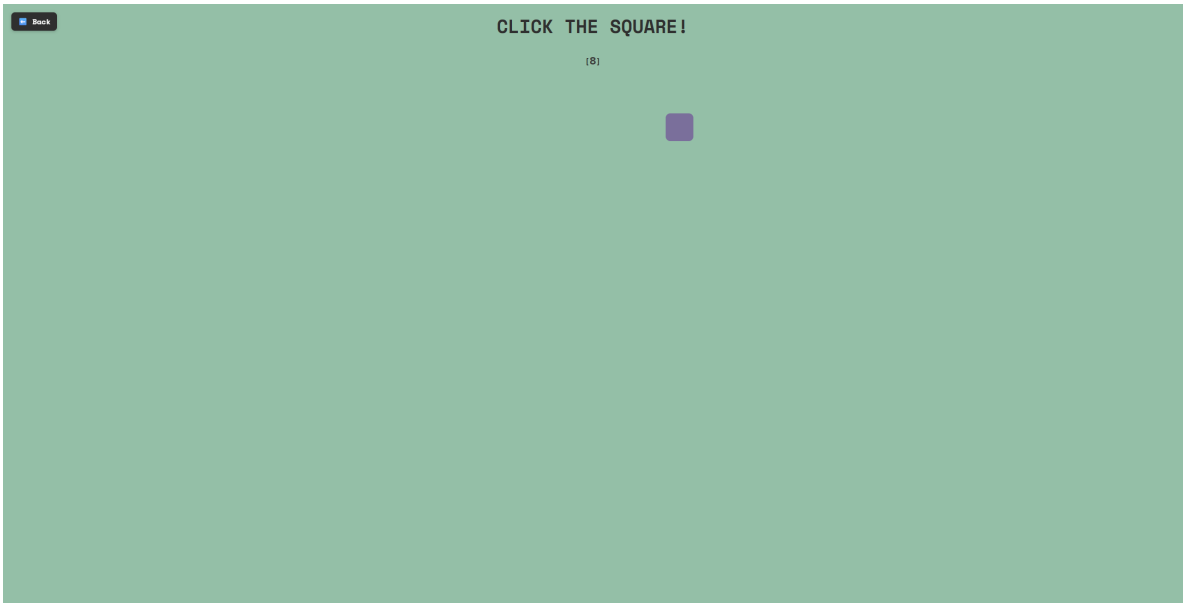

Figure 2: Minigame 1

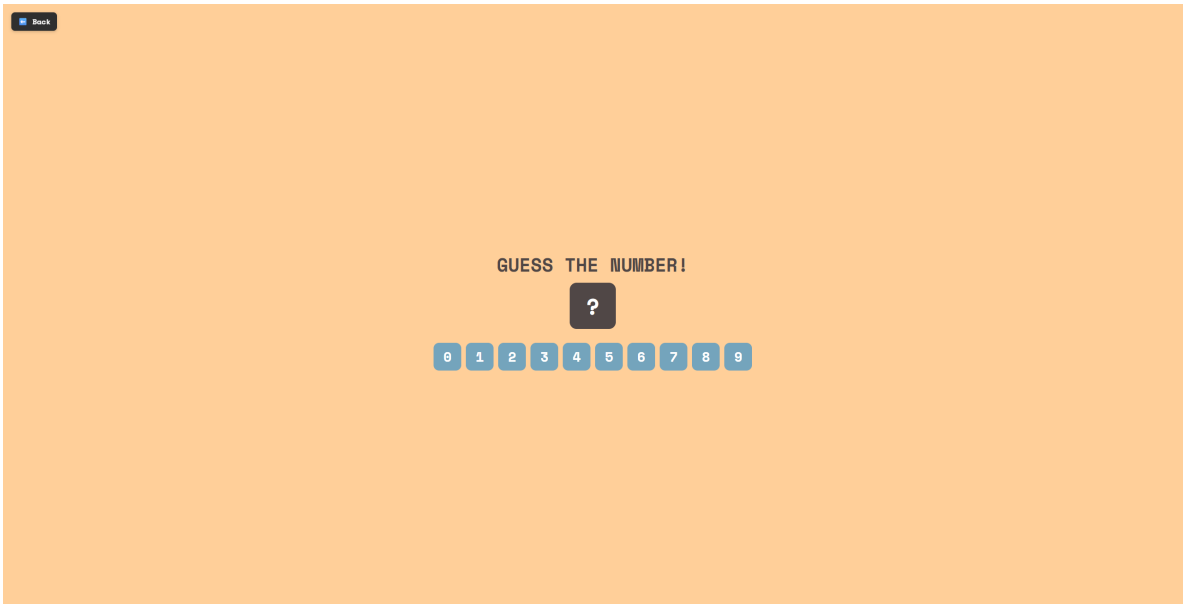

Figure 3: Minigame 2

Figure 4: Error 404