Assignment 3

Instructions

To run Problem 1:

In terminal 1:

```
go run srv.go # runs on port 3000
```

In terminal 2:

```
go run p1.go
```

To run Problem 2:

In terminal 1:

```
cd p2_srv/
go run . # runs on port 3000
```

In terminal 2:

```
go run p2.go
```

Note: Can only run server at a time and if you don't get p2 to work check with `init.sql` file that has the correct schema to make sure the db tables are correct.

Problems

Problem 2

Consists of two main components: a client-side application (`p2.go`) and a server-side application (`p2_srv.go`). The server-side application interacts with a MySQL database to perform CRUD operations on student records, while the client-side application facilitates user interaction via the interface in the terminal.

Client-side Application (`p2.go`)

1. It provides an interface in the terminal for users to interact with students stored in the database.
2. Users can perform operations such as creating, retrieving, updating, and deleting students.

3. Each operation sends HTTP requests to the server-side application (`p2_srv.go`) to perform the corresponding database operation.
4. It uses the `http` package in Go to make HTTP requests to the server.

Server-side Application (`p2_srv.go`)

1. It uses the Gorilla Mux router to define routes for handling HTTP requests.
2. It connects to a MySQL database and with a the `students_db` schemas `students` table. The structure of the database is according to the provided `init.sql` script.
3. It defines HTTP handler functions for handling CRUD operations on student records.
4. It uses SQL queries to interact with the MySQL database (`INSERT`, `SELECT`, `UPDATE`, `DELETE`).

Concurrency Handling

Concurrency in web applications typically involves handling multiple requests simultaneously without data corruption or inconsistency. Though if the concurrency has not been tested due to unsure how it should be implemented.