Aufgabe 6 Erfahrungen

(in Zusammenarbeit mit Annsophie Rösch)

Zunächst habe ich die Server.ts Datei in mein Repository übernommen und diese ausführlich kommentiert.

```
import * as Http from "http"; // arbeitet mit Node: macht ein neues http Objekt, damit interpreter nach import dazu sucher
     namespace L05_Server {
         console.log("Starting server"); //"Starting server" wird in der konsole ausgegenen (direkt nachdem namespace erstellt
         let port: number = Number(process.env.PORT); //neue variable port vom typ number: setzen der Umgebungsvariable PORT da
         if (!port)
             port = 8100;
         let server: Http.Server = Http.createServer(); //neue Variable server mit dem typ Http.server wird erstellt, und mit e
         server.addListener("request", handleRequest); //auf dem server wird ein Event-Listener erstellt. Dieser führt die Fun
         server.addListener("listening", handleListen); //zweiter Event-listener: handleListen wird ausgeführt, wen der server
         server.listen(port);
         function handleListen(): void {
                                             //deklarienen der neue funktion handleListen, keine parameter, void -> keine rückg
             console.log("Listening"); //wenn die Funktion handleListen aufgerufen wird (was genau dann passiert wenn der ever
         function handleRequest(_request: Http.IncomingMessage, _response: Http.ServerResponse): void {    //neue funktion handleR
             console.log("I hear voices!");
             _response.setHeader("content-type", "text/html; charset=utf-8"); //_response erhält werte für den header
             _response.setHeader("Access-Control-Allow-Origin", "*"); // noch mehr was in den header von _response kommt
             _response.write(_request.url); //die soeben in _response definierten werte werden nun die url von _request geschri
             console.log(_request.url);
27
             _response.end(); //schließt _response ab und signalisiert dies dem Server
                                                                                                   7: node
                                  TERMINAL
```

Als nächstes habe ich mit npm install --save @types/node Node.js in meinem Repository Ordner installiert.

Danach wollte ich mit node (dem richtigen Dateipfad) einen lokalen Server auf meinen Rechner starten und das hat leider ewig nich geklappt. Wir haben alles versucht, stundenlang nach einem Tippfehler gesucht, im internet recherchiert, ein bisschen geweint und am Ende das Issue geschrieben. Es war kein Tippfehler man hat nur unterschiedliche Dateipfade. Now we know.

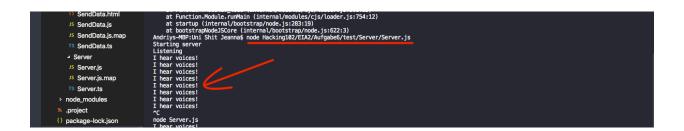
```
Andriys—MBP:Uni Shit Jeanna$ npm install ——save @types/node
npm MARX saveError BNDEN: no such file or directory, open '/Users/Jeanna/Desktop/Uni Shit/package.json'
npm notice created a lockfile as package—lock.json. You should commit this file.
npm MARX encent BNDENI: no such file or directory, open '/Users/Jeanna/Desktop/Uni Shit/package.json'
npm MARX int Shit No repository field.
npm MARX int Shit No repository field.
npm MARX int Shit No repository field.
npm MARX int Shit No README data
npm MAXX int Shit No README data
npm MAXX int Shit No README data
npm MAXX int Shit Jeanna$ node ....
internal/modules/cjs/loader.js:584
throw err;

Andriys—MBP:Uni Shit Jeanna$ node ....
internal/modules/cjs/loader.js:582:15)
at Function.Module._nod (internal/modules/cjs/loader.js:582:25)
at Function.Module.runfain (internal/modules/cjs/loader.js:754:12)
at startup (internal/bootstrap/node.js:283:19)
at bootstrapModeJSCore (internal/modules/cjs/loader.js:582:3)
Andriys—MBP:Uni Shit Jeanna$ node Server.js
internal/modules/cjs/loader.js:584
throw err;

Andriys—MBP:Uni Shit Jeanna$ node Server.js
internal/modules/cjs/loader.js:582:30
at Function.Module._load (internal/modules/cjs/loader.js:582:35)
at Function.Module._load (internal/modules/cjs/loader.js:582:35)
at startup (internal/modules/cjs/loader.js:582:35)
at startup (internal/modules/cjs/loader.js:582:35)
at startup (internal/modules/cjs/loader.js:582:35)
at Function.Module._load (internal/modules/cjs/loader.js:582:35)
at Function.Module._load (internal/modules/cjs/loader.js:582:35)
at Function.Module._load (internal/modules/cjs/loader.js:582:35)
at Function.Modules._load (internal/modules/cjs/loader.js:582:35)
at Function.Modules._load (internal/modules/cjs/loader.js:582:35)
at Function.Modules._load (internal/modules/cjs/loader.js:582:35)
at Function.Modules._load (internal/modules/cjs/loader.js:582:35)
at Function.Modules/cjs/loader.js:582:35)
at Function.Modules/cjs/loader.js:582:35

at Function.Modules/cjs/loader.js:582:35

at Function.Modules/cjs/loader.js:582:35
```



Nachdem das endlich funktioniert hat konnte ich den Server endlich starten und damit arbeiten. Am Anfang natürlich die Stimmen :)

Als nächstes habe ich den Request-Handler soweit angepasst, dass der eingegebene text auch auf der Console erscheint. Dafür habe ich mit dem Befehl console.log(_request.url) einfach auf der Konsole ausgegeben.



Danach habe ich mir einen Account auf Heroku angelegt und mich lange zeit mit dem Application Error herumgeschlagen (Logs und screenshots im Issue).

Zum Schluss muss die Eisdealer Seite angepasst werden.

Davor habe ich jedoch meine Seite auch für Unentschlossene angepasst und einen Button hinzugefügt der das letzte Child der Liste mit Eissorten wieder entfernt.

Die URL meines Localhosts hat dann endlich auch den Link zur Heroku App bekommen. Ich habe anschließend einen neuen Button vom typen submit hinzugefügt, dafür musste ich den typ aller anderen Buttons auf "Button" anpassen da sonst auch beim klick auf den "will ich" Button ein submit ausgelöst wurde.

Es werden bereits die Auswahl und die Radios angezeigt, da ich aber keine Stepper habe will das anzeigen des Warenkorbs noch nicht ganz funktionieren. Ich versuche es so schnell wie möglich zu fixen und hoffe das es bald klappt.