**שאלה 1**:

על מנת להציג Heat Map Chart יצרתי אובייקט בעל התכונות series ו options. שדה ה series יכיל שדה data בו ישמר מערך של כל הערכים המספריים מjason שמתקבל כקלט. שדה הoption מכיל את ההוראות הדרושות לתצוגה של ה Heat Map Chart כפי שהתבקש.

הקוד לחילוץ ה data מה input:

        let data = []

        for(let i=0 ;i<10000;i++){

            if(props.input[i]!=null && props.input[i].value!=undefined && props.input[i].value>0){

                data.push(props.input[i].value)

            }

        }

על מנת להציג את האובייקט השתמשתי בקומפוננטה ReactApexChart בצורה הבאה:

<ReactApexChart options={this.state.options} series={this.state.series} type="heatmap" height={350} />

עבור הקלט:

let q1Input = [

    {timestamp : "01-01-2020T01:23:34"},

    {timestamp : "01-01-2020T01:23:34",value: 56},

    {timestamp : "01-01-2020T01:23:34",value: -4},

    {timestamp : "01-01-2020T01:23:34",value: 22},

    {timestamp : "01-01-2020T01:23:34",value: 99},

    {timestamp : "01-01-2020T01:23:34",value: 1},

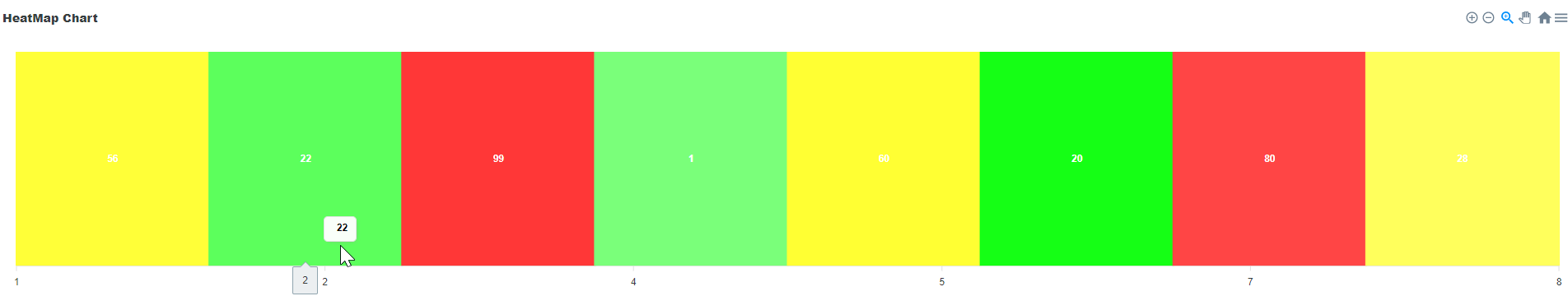
    {timestamp : "01-01-2020T01:23:34",value: 60},

    {timestamp : "01-01-2020T01:23:34",value: 20},

    {timestamp : "01-01-2020T01:23:34",value: 80},

    {timestamp : "01-01-2020T01:23:34",value: 28},

]

התקבל ה Heat Map Chart הבא:

**שאלה 2**

על מנת לפתור את הבעיה השתמשתי בשיטות הבאות הלקוחות מתוך rxjs:

import { defer } from 'rxjs';

import { shareReplay} from "rxjs/operators";

הגדרתי את הפונקציה הבאה שתדמה גישה לסרבר חיצוני:

function APIGetCities(){

    console.log("api")

    const data = [

        { name: "Jerusalem", Population : 936425 },

        { name: "Tel-Aviv", Population: 460613 },

        { name: "Haifa", Population: 285316}

    ]

    var promise = new Promise(function (resolve, reject) {

        window.setTimeout(function () {

            resolve(data);

        }, 1000);

    })

    return promise

}

לה הגדרתי observer בצורה הבאה:

const observable = defer(APIGetCities).pipe(shareReplay())

את שתי הקומפוננטות השונות המעוניינות במידע מהסרבר הגדרתי "כמשקיפות" בצורה הבאה בעת עלייתן:

function Component1(){

    const [cities, setCities] = useState(null)

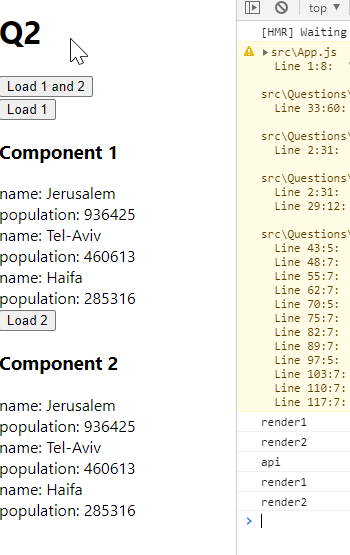
    console.log("render1")

    useEffect(()=>{

        observable.subscribe(x=>setCities(x))

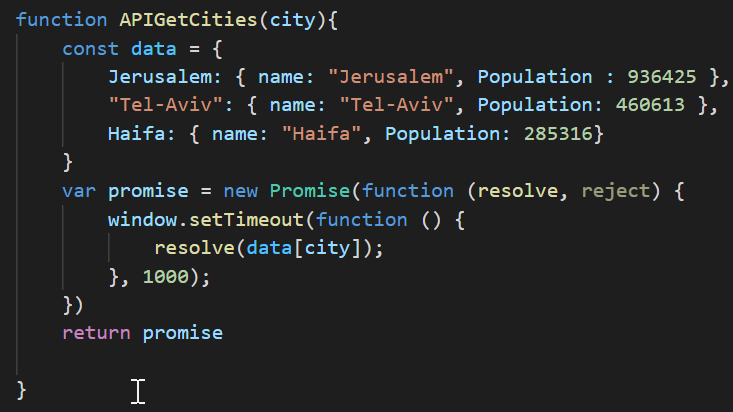
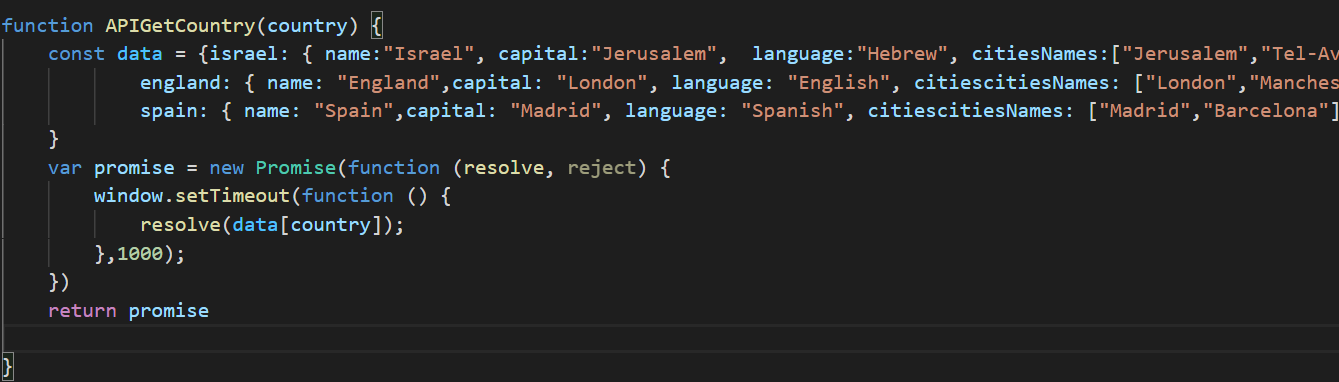
    },[])

בצורה הזאת ללא קשר לסדר טעינתן של הקומפוננטות הן קיבלו את אותו מידע אשר נלקח פעם אחת בלבד מהסרבר החיצוני.

ניתן לראות פה ששתי הקומפוננטות נטענו במקביל תוך כדי גישה אחת בלבד לסרבר:

**שאלה 3**:

על מנת לגשת לבעיה הגדרתי את שתי הפונציות הבאות שידמו גישה ל api חיצוני:



על מנת לפתור את הבעיה כתבתי Promise שעבור כל מדינה שהוא מקבל שולף את המידע עבורה מה API הרלוונטי, ולאחר שמסיים לשלוף את המידע, בודק האם המדינה נמצאת ברשימת המדינות עליהן יש לשלוף את המידע על הערים בה, במידה וכן שולף את המידע על כל הערים בה. לאחר שמסיים מחזיר את המידע ששלף.

להלן הקוד:

    function getCountryAndCities(countryName){

        var promise = new Promise(function (resolve, reject) {

            APIGetCountry(countryName).then((ans)=>{

                if(props.favoriteCountriesNames.includes(countryName)){

                    Promise.all(ans.citiesNames.map((city)=>{

                        return APIGetCities(city)

                    })).then(values=> {ans.cities=values; resolve(ans) })

                }

                else{

                    resolve(ans)

                }

            })

        })

        return promise

    }

עבור כל המדינות הנדרשות קראתי לפרומיס הזה בצורה הבאה:

        useEffect(() => {

            Promise.all(props.countriesNames.map((name) => {

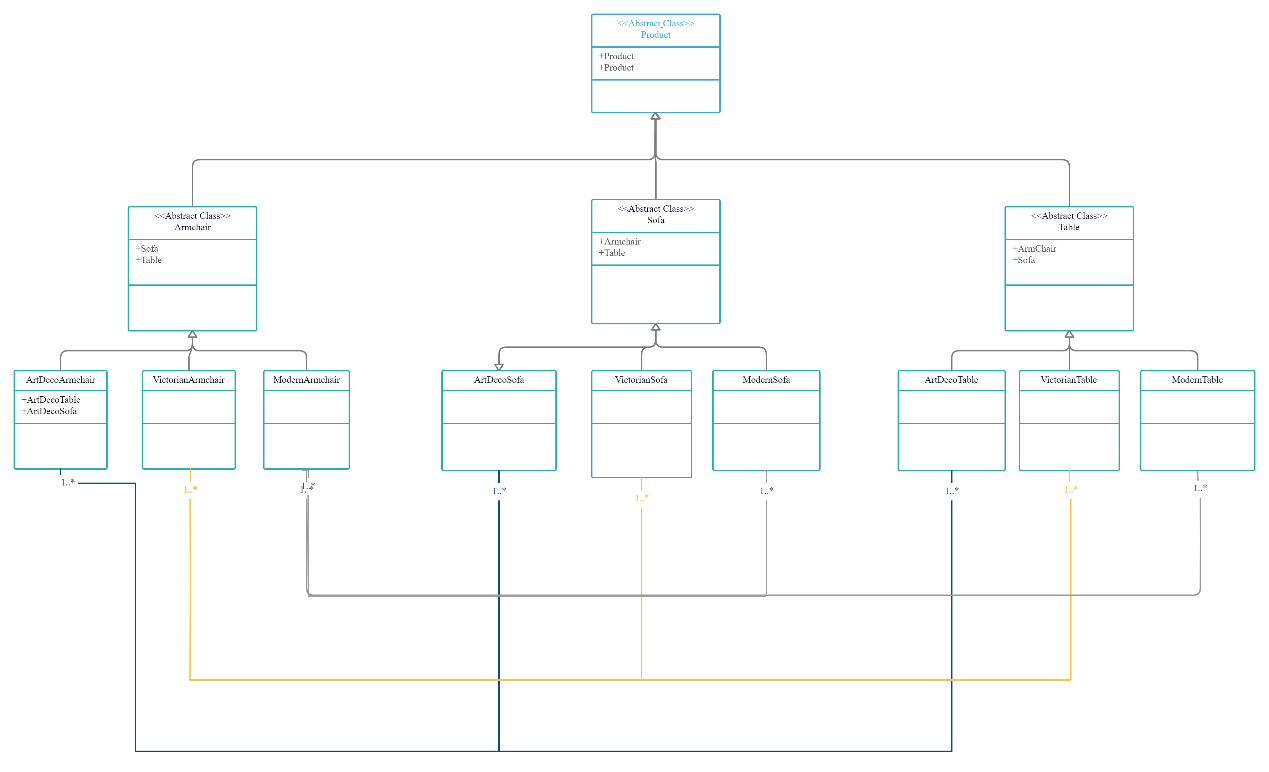
            return getCountryAndCities(name)})).then(values =>{ setCountries(values)})

        }, [])

בצורה הזאת שמרתי על קוד סינכרוני, אשר מבצע במקביל מספר קריאות לapi ללא צורך להמתין לקבלת תשובה לפני שמבצע את הקריאה הבאה.

**שאלה 4**:

להלן תרשים המחלקות עבור המקרה:



כלל האובייקטים יירשו מהמחלקה האבסטרקטית Product, ממנה יירשו באופן ישיר 3 המחלקות האבסטרקטיות Armchair,Sofa ו Table. עבור כל אחת מן המחלקות יירשו 3 מחלקות, לדוגמה עבור Armchair יירשו אותה ArtDecoArmchair, VictorianArmchair ו ModernArmchair. לכל אחת מן המחלקות האלו יהיה מצביע למבנה נתונים שמכיל את הפריטים מאותה family-style. לדוגמא ModernArmchair יצביע למבנה נתונים שמכיל את כלל ה ModernSofa ומבנה נתונים המכיל את כלל ה ModernTable.

בצורת המימוש הזאת אנחנו יכולים בקלות ליצור קישור בין אובייקט לאובייקטים אחרים שמתאימים להגיע איתו. נוסף לכך עניין הירושות יחסוך כפל קוד כיוון שמחלקות דומות שייכות לאותה מחלקת אב כך שתכונות זהות יכנסו למחלקות האב. וקוד כזה גם ישמור על מודולריות, שכן בקלות יחסית ניתן להכניס פרטי ריהוט או family-style נוספים כמעט ללא צורך בלשנות את הקוד הקיים.