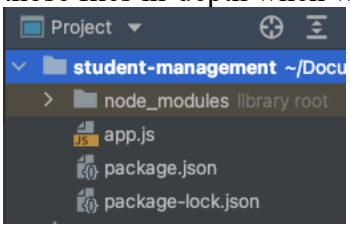
Assignment 2

Due Date Saturday, Feb 26, 2022 , 11: 59 PM

Q1. Student Management App

1. Create a new project named **student-management**
2. Create a JavaScript file named **app.js**
3. Open your terminal and type the following command **[npm i prompt-sync]** in order to get the '**prompt-sync'** NPM package, that allows you to read user input from the terminal. This command will create multiple files inside your project. We will talk about those files in-depth when we discuss about Node.js.



4. Open the package.json and add the following line **"type" : "module",**
5. Import the **prompt-sync package** inside the app.js

```
import promptSync from 'prompt-sync';
const prompt = promptSync()
```

6. Create an array named **students**
7. Create a loop that asks the user to add 5 students. You should only ask the user the **name** and **gender** of the student. The age and grade of the students should be randomly generated. The age should be between [**17 - 35**] and the grade should be between [**0 and 100**] inclusive.

   Sample: the students array after adding the five students should look something like this

```
[
  { name: 'Hani', gender: 'Male', grade: 92, age: 19 },
  { name: 'Ahmed', gender: 'Male', grade: 81, age: 30 },
  { name: 'Sara', gender: 'Female', grade: 99, age: 24 },
  { name: 'Emany', gender: 'Female', grade: 50, age: 26 },
  { name: 'Issa', gender: 'Male', grade: 70, age: 24 }
]
```

8. The write the code that answers the following questions
   - **a.** Find the student with the highest grade
   - b. Find both the youngest and oldest student in the class
   - c. Find the average and median student age in the class
   - d. Calculate the mean and variance of the student grades.

     Note: **Do not use the traditional loops to solve the above questions. You should use the arrow function.**

Mean	Variance

$\bar{x} = \frac{1}{n}\sum_{i=0}^{n-1} x_i$	$\sigma^2 = \frac{1}{n-1}\sum_{i=0}^{n-1}(x_i - \bar{x})^2$

Q2. Book Donation App

You are required to develop a Book Donation App that allows people to donate books.

The book donation system should have the following classes

a) **Book** – bookId, title, author, imageUrl, donorId, status

DonorId property is should hold , the ID of the person who donated the book.

Note that the **status** attribute can have one of the following values:

- **pending** : As soon as the user adds a donated book to the system, the status is set to pending.
- **available** : When the book donor delivered the book to the store then status of the book can be changed to available.

b) **Donor** - donorId, firstname , lastname , phone, street, city, email, password

c) **BookCatalog** – contains list of **books** and a list of **donors**. Also the class contains the following methods

addBook(book)	Adds a book to the list of books (the status of newly added boom should be set to pending).
updateBook(book)	Updates the book having the matching bookId.
deleteBook(bookId)	Deletes the book from the list of books.
getBooks(status=available)	Returns the books by status. By default, only available books should be return.
getDonorBooks(donorId)	Returns the books donated by a particular donor.
getTopDonors(topCount)	Returns the top book donors. Eg. If the user passes 3 as topCount parameter, then this function returns the top 3 donors and the list of books each one has donated.
addDonor	Add new donor to the list of **donors**
updateDonor(donorId)	allow the user to update the donor details
deleteDonor(donorId)	You can delete a donor, if they are not associated with any book

Test your code by creating **app.js file** to instantiate the **BookCatalog** class and test its methods.

Note that you should make use of the **JavaScript features and capabilities** such as **arrow functions**, array functions **(.map, .reduce, .filter, .splice, .sort…), spread operator**, **object literals**, and **classes**.

Q.3 Basic JS Unit Testing Using Mocha and Chai

1. In this section you will create a **book-catalog.spec.js** file to unit test the methods of the **BookCatalog** class.
2. Write a unit test for each of the following methods inside the **book-catalog.spec.js** file.

    addBook(book)
    updateBook(book)
    deleteBook(bookId)
    getBooks(status=available)
    getDonorBooks(donorId)
    getTopDonors(topCount)

After you complete the assignment, fill in the ***TestingDoc-Grading-Sheet.docx*** and save it inside {**Your Git Hub Repo}/Assignments/*Assignment2*** folder. Push your work to GitHub repository. No email submissions will be accepted.