READ FOR STEP 2 : MONGO DB

- MongoDB name comes from **HuMONGOus** data.
- MongoDB is comprised of DATABASES -> COLLECTIONS -> DOCUMENTS

  MongoDB is a document database – scalable, flexible (with flexible structure), fast and appropriate as for small so for huge stocks of data

- It is a NoSQL Database.
- It is Open source and free to use.
- It is a cross-platform database which works with almost every platform (Windows, Mac, Linux) .
- It stores data in flexible, JSON like documents - BSON.

READ FOR STEP 2:
BASIC SQL TO MONGODB TERMINOLOGY COMPARISON

Terminology & Concept Comparison:

SQL	MongoDB
database	database
table	collection
row	document or BSON document
column	field
index	index
table joins	$lookup, embedded documents
primary key (need to specify column or combination of column)	primary key (automatically set to _id field)
aggregation	aggregation pipeline

READ FOR STEP 2 :
ROLES IN OLYMPICS APP:

- Admin, Moderator, User – each one of them can use only his views
- All others – are public, guests – they also could use some views
- For this we will need Database and inside it the collections of “roles” and “users”.

STEP 2

- A) Use “Compas” GUI of MongoDB and create Database named “olympics” – it will ask you about the name for the first Collection – so let it be “sports”
- B) Now back to the server, there we install “mongoose”
- C) Inside olympics/server we create the file .env – for environment variables and fill it by:
  ```
  MONGODB_URI='mongodb://127.0.0.1:27017/olympics’
  ```
- D) Now stop the server. In the window of the server run:
  $ source .env
  (reminder – this command runs the script .env that creates the environment variables, for now it is only MONGODB_URI, but there will be more)
- E) You could also install dotenv package to perform this action automatically, but it should be never used in production, so install it for development only like this:
  $ npm i –D dotenv

  and after that put into server.js:

  ```
  const express = require('express');
  ```
- ```
  require('dotenv').config();
  ```
  ```

STEP 2 - CONTINUED

- F) Create directories olympics/server/db and olympics/server/model

- G) In olympics/server/db we create file db.js and create inside async connect function, for example:

- H) Import it into the main server file, and run the function somewhere there (before the creation of the Express server, for example)

- I) Ensure that the server connects to DB

```
const mongoose = require('mongoose');

const connectDB = async () => {
    try {

        console.log(`process.env.MONGODB_URI :
                                    ${process.env.MONGODB_URI}`);
        await mongoose.connect(process.env.MONGODB_URI, {});
        console.log('Connected to MongoDB, Mazal Tov!');

    } catch (err) {

        console.log('Error connecting to MongoDB:');
        throw err;

    }
  };

  module.exports = connectDB;
```

STEP 2 - HINTS

- 1. To import the DB connection function into the main server.js file:
- `const express = require('express');`
- `require('dotenv').config();`
- `const connectDB = require('./db/db');`

- 2. To run it:
- `connectDB();`
- `const app = express();`
- (you could use the keyword await before, but we anyway kill the process, if the server does not connect to the DB)

- C) If it does not connect to the server, ensure that DB is up and you're working OK with Compass,
- And ensure, that your connection line does not include "localhost" but explicit IPV4 address (as NodeJS translates localhost to IPV6 address, but mongodb does not use this form of the address yet):
- `MONGODB_URI='mongodb://127.0.0.1:27017/olympics'`