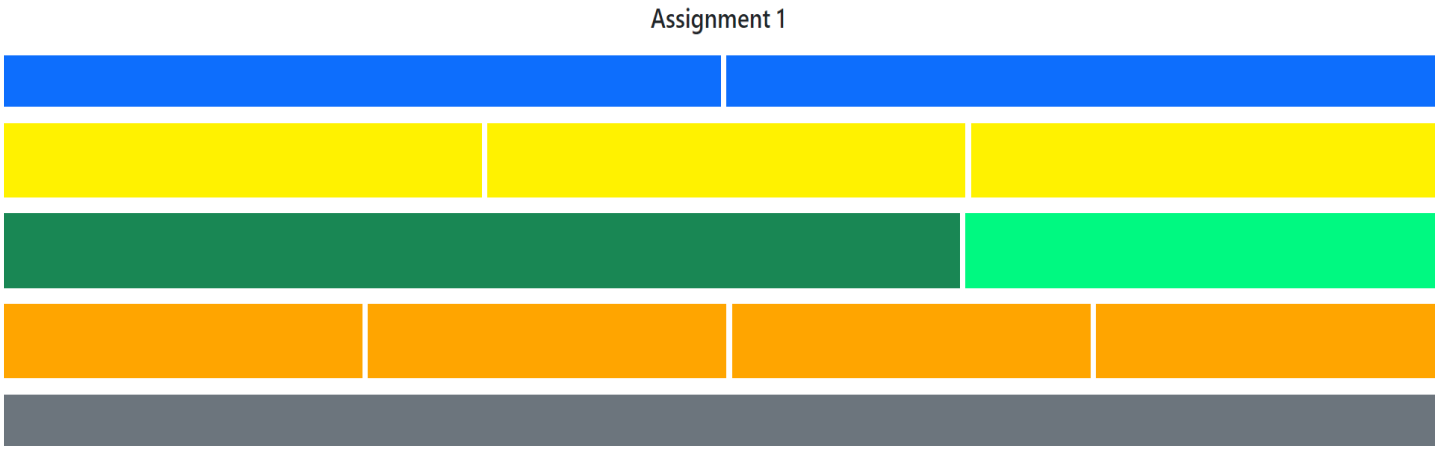
Tutorial 1

1. Design html page to get below given output using bootstrap css.

```
<!doctype html>
<html lang="en">
<head>
   <!-- meta tags -->
   <meta charset="utf-8">
   <meta name="viewport" content="width=device-width, initial-scale=1">

   <!-- CSS -->
   <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css"
rel="stylesheet" integrity="sha384-
EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65VohhpuuCOmLASjC"
crossorigin="anonymous">
   <title>Assignment 1</title>
   <style>
      .row{flex-wrap: nowrap;}
      .col-sm-6, .col-sm-12{ height: 45px; margin:3px}
      .col-sm-4, .col-sm-8, .col-sm-3{ height: 65px; margin:3px}
      .col-sm-3{ background: orange;} .col-sm-4{ background: #fff200;}
   </style>
</head>
<body>
   <div class="container-fluid mt-2">
      <h4 class="text-center">Assignment 1</h4>
      <div class="row mt-3">
         <div class="col-sm-6 bg-primary "> </div>
         <div class="col-sm-6 bg-primary "> </div>
      </div>
      <div class="row mt-2">
         <div class="col-sm-4 "> </div>
         <div class="col-sm-4 "> </div>
         <div class="col-sm-4 "> </div>
      </div>
      <div class="row mt-2">
```

```
            <div class="col-sm-8 bg-success "> </div>
            <div class="col-sm-4 " style="background: #00F981"> </div>
        </div>
        <div class="row mt-2">
            <div class="col-sm-3 "> </div>
            <div class="col-sm-3 "> </div>
            <div class="col-sm-3 "> </div>
            <div class="col-sm-3 "> </div>
        </div>
        <div class="row mt-2">
            <div class="col-sm-12 bg-secondary "> </div>
        </div>
    </div>

    <!-- JS -->
    <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.bundle.min.js"
integrity="sha384-
MrcW6ZMFYlzcLA8Nl+NtUVF0sA7MsXsP1UyJoMp4YLEuNSfAP+JcXn/tWtIaxVXM"
crossorigin="anonymous"></script>
</body>
</html>
```

OUTPUT :



2. Use a Bootstrap class to style the table properly and get the following output (with padding and horizontal dividers).

   - Add zebra-stripes to the table.
   - Add borders on all sides of the table and cells.
   - Enable a hover state on table rows.
   - Make the table more compact by cutting cell padding in half.
   - Use contextual classes to add the following:
     - Green color to the table row containing John.
     - Red color to the table row containing Mary.
     - Orange color to the last table row.

```
<!doctype html>
<html lang="en">
<head>
  <!-- meta tags -->
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">

  <!-- CSS -->
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css"
rel="stylesheet" integrity="sha384-
1BmE4kWBq78iYhFldvKuhfTAU6auU8tT94WrHftjDbrCEXSU1oBoqyl2QvZ6jIW3"
crossorigin="anonymous">
  <title>Assignment 1</title>
  <style>

  </style>
</head>
<body>
  <div class="container mt-5">
    <h6 class="text-center">Assignment 1 (2)</h6>
    <table class="table mt-3 table-striped table-bordered table-hover table-condensed">
      <thead>
        <tr>
          <th scope="col">First Name</th>
          <th scope="col">Last Name</th>
```

```
            <th scope="col">Email</th>
          </tr>
        </thead>
        <tbody>
          <tr class="bg-success">
            <td>John</td>
            <td>Doe</td>
            <td>john@example.com</td>
          </tr>
          <tr class="bg-danger">
            <td>Mary</td>
            <td>Moe</td>
            <td>mary@example.com</td>
          </tr>
          <tr class="bg-warning">
            <td>July</td>
            <td>Dooley</td>
            <td>july@example.com</td>
          </tr>
        </tbody>
      </table>
  </div>

  <!-- JS -->
  <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min.js"
integrity="sha384-
ka7Sk0Gln4gmtz2MlQnikT1wXgYsOg+OMhuP+IlRH9sENBO0LRn5q+8nbTov4+1p"
crossorigin="anonymous"></script>
</body>
</html>
```

OUTPUT :

Assignment 1 (2)

First Name	Last Name	Email
John	Doe	john@example.com
Mary	Moe	mary@example.com
July	Dooley	july@example.com

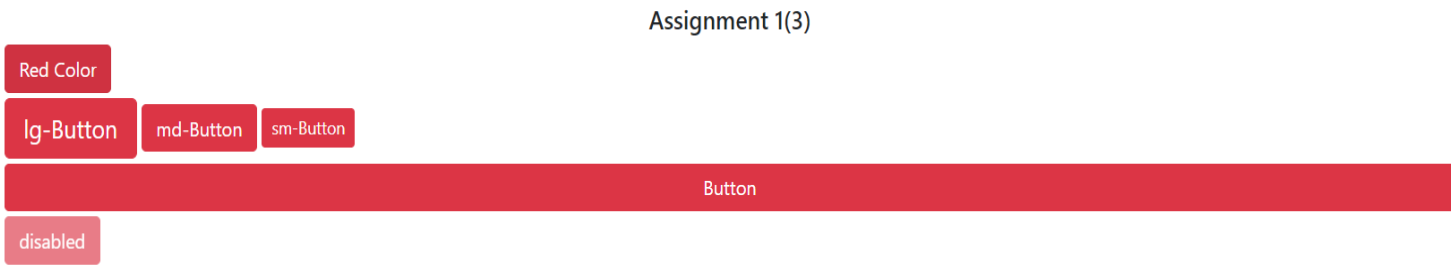
3. Bootstrapping with Buttons.

- Use a Bootstrap class to style the button properly with a red color.
- Change the size of the buttons in the following order: large, medium, small and xsmall.
- Make the button span the entire width of the parent element.
- Use a Bootstrap class to disable the button.

```
<!doctype html>
<html lang="en">
<head>
  <!-- meta tags -->
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
```

```
    <!-- CSS -->
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css"
rel="stylesheet" integrity="sha384-
1BmE4kWBq78iYhFldvKuhfTAU6auU8tT94WrHftjDbrCEXSU1oBoqyl2QvZ6jIW3"
crossorigin="anonymous">

    <title>Assignment 1</title>
    <style>
        .row{display: block; } .row button{width:auto;}
    </style>
</head>
<body>
    <div class="container mt-5">
        <h5 class="text-center">Assignment 1(3)</h5>
        <button type="button" class="btn btn-danger">Red Color</button>
        <div class="row mt-1 btn-group-xs g-0">
            <button type="button" class="btn btn-danger btn-lg ">lg-Button</button>
            <button type="button" class="btn btn-danger btn-md ">md-Button</button>
            <button type="button" class="btn btn-danger btn-sm ">sm-Button</button>
        </div>
        <div class="row mt-1 g-0">
            <button type="button" class="btn btn-danger " style="width:-webkit-fill-
available">Button</button>
        </div>
        <button type="button" class="btn btn-danger disabled mt-1">disabled</button>
    </div>
        <!-- JS -->
    <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min.js"
integrity="sha384-
ka7Sk0Gln4gmtz2MlQnikT1wXgYsOg+OMhuP+IlRH9sENBO0LRn5q+8nbTov4+1p"
crossorigin="anonymous"></script>
</body>
</html>
```

OUTPUT :

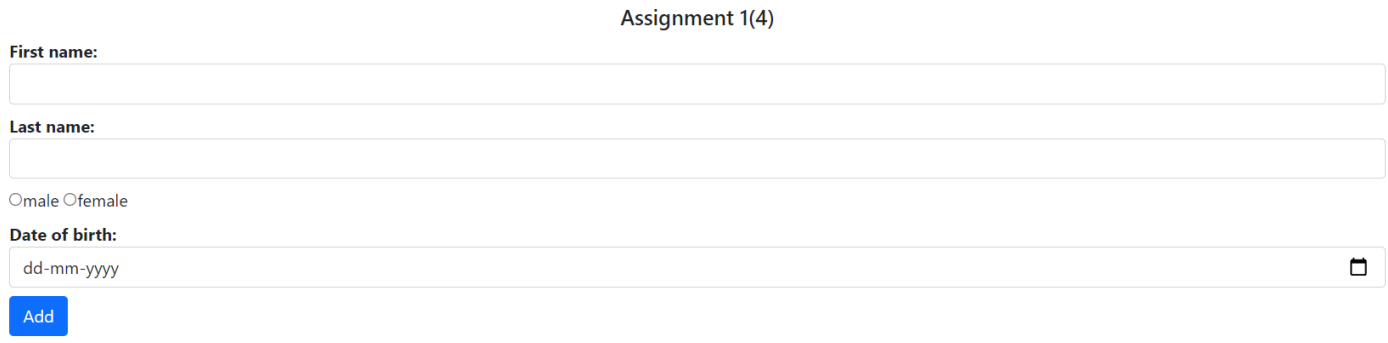


4. Style the below given html form using bootstrap to get the output shown below.

```
<!doctype html>
<html lang="en">
<head>
  <!-- meta tags -->
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">

  <!-- CSS -->
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css"
rel="stylesheet" integrity="sha384-
1BmE4kWBq78iYhFldvKuhfTAU6auU8tT94WrHftjDbrCEXSU1oBoqyl2QvZ6jIW3"
crossorigin="anonymous">

  <title>Hello, world!</title>
</head>
<body>
  <div class="container mt-5">
    <h5 class="text-center">Assignment 1(4)</h5>
    <form action="#">
      <div class="mb-2">
        <label for="first_name" class="fw-bold">First name:</label>
        <input type="text" class="form-control" name="first_name" id="first_name"/>
      </div>
      <div class="mb-2">
        <label for="last_name" class="fw-bold">Last name:</label>
        <input type="text" class="form-control" name="last_name" id="last_name"/>
      </div>
```

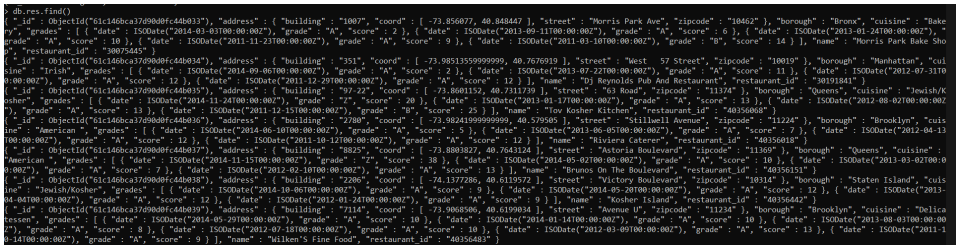
```
        <div class="mb-2">
          <label><input type="radio" name="gender" value="male"/>male</label>
          <label><input type="radio" name="gender" value="female"/>female</label>
        </div>
        <div class="mb-2">
          <label for="birth_date" class="fw-bold">Date of birth:</label>
          <input type="date" class="form-control" name="birth_date" id="birth_date"/>
        </div>
        <input type="submit" class="btn btn-primary" value="Add"/>
      </form>
    </div>
    <!-- JS -->
    <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min.js"
integrity="sha384-
ka7Sk0Gln4gmtz2MlQnikT1wXgYsOg+OMhuP+IlRH9sENBO0LRn5q+8nbTov4+1p"
crossorigin="anonymous"></script>
</body>
</html>
```

OUTPUT :



Tutorial 2

1. Write a MongoDB query to display all the documents in the collection restaurants.



2. Write a MongoDB query to display the fields restaurant_id, name, borough and cuisine for all the documents in the collection restaurant.

3. Write a MongoDB query to display the fields restaurant_id, name, borough and cuisine, but exclude the field _id for all the documents in the collection restaurant.

```
> db.res.find({}, { restaurant_id:1, name:1, cuisine:1, borough:1, _id:0})
{ "borough" : "Bronx", "cuisine" : "Bakery", "name" : "Morris Park Bake Shop", "restaurant_id" : "30075445" }
{ "borough" : "Manhattan", "cuisine" : "Irish", "name" : "Dj Reynolds Pub And Restaurant", "restaurant_id" : "30191841" }
{ "borough" : "Queens", "cuisine" : "Jewish/Kosher", "name" : "Tov Kosher Kitchen", "restaurant_id" : "40356068" }
{ "borough" : "Brooklyn", "cuisine" : "American ", "name" : "Riviera Caterer", "restaurant_id" : "40356018" }
{ "borough" : "Queens", "cuisine" : "American ", "name" : "Brunos On The Boulevard", "restaurant_id" : "40356151" }
{ "borough" : "Staten Island", "cuisine" : "Jewish/Kosher", "name" : "Kosher Island", "restaurant_id" : "40356442" }
{ "borough" : "Brooklyn", "cuisine" : "Delicatessen", "name" : "Wilken'S Fine Food", "restaurant_id" : "40356483" }
{ "borough" : "Brooklyn", "cuisine" : "Ice Cream, Gelato, Yogurt, Ices", "name" : "Taste The Tropics Ice Cream", "restaurant_id" : "40356731" }
{ "borough" : "Brooklyn", "cuisine" : "American ", "name" : "Regina Caterers", "restaurant_id" : "40356649" }
{ "borough" : "Bronx", "cuisine" : "American ", "name" : "Wild Asia", "restaurant_id" : "40357217" }
{ "borough" : "Brooklyn", "cuisine" : "American ", "name" : "C & C Catering Service", "restaurant_id" : "40357437" }
{ "borough" : "Brooklyn", "cuisine" : "Chinese", "name" : "May May Kitchen", "restaurant_id" : "40358429" }
{ "borough" : "Manhattan", "cuisine" : "American ", "name" : "1 East 66Th Street Kitchen", "restaurant_id" : "40359480" }
{ "borough" : "Brooklyn", "cuisine" : "Hamburgers", "name" : "Wendy'S", "restaurant_id" : "30112340" }
{ "borough" : "Manhattan", "cuisine" : "American ", "name" : "Angelika Film Center", "restaurant_id" : "40362274" }
{ "borough" : "Brooklyn", "cuisine" : "Hamburgers", "name" : "White Castle", "restaurant_id" : "40362344" }
{ "borough" : "Manhattan", "cuisine" : "American ", "name" : "P & S Deli Grocery", "restaurant_id" : "40362264" }
{ "borough" : "Queens", "cuisine" : "Chinese", "name" : "Ho Mei Restaurant", "restaurant_id" : "40362432" }
{ "borough" : "Manhattan", "cuisine" : "Turkish", "name" : "The Country Cafe", "restaurant_id" : "40362715" }
{ "borough" : "Brooklyn", "cuisine" : "Caribbean", "name" : "Shashemene Int'L Restaura", "restaurant_id" : "40362869" }
Type "it" for more
```

4. Write a MongoDB query to display the fields restaurant_id, name, borough and zip code, but exclude the field _id for all the documents in the collection restaurant.

```
> db.res.find({}, { restaurant_id:1, name:1, address:{zipcode:1}, borough:1, _id:0})
{ "address" : { "zipcode" : "10462" }, "borough" : "Bronx", "name" : "Morris Park Bake Shop", "restaurant_id" : "30075445" }
{ "address" : { "zipcode" : "10019" }, "borough" : "Manhattan", "name" : "Dj Reynolds Pub And Restaurant", "restaurant_id" : "30191841" }
{ "address" : { "zipcode" : "11374" }, "borough" : "Queens", "name" : "Tov Kosher Kitchen", "restaurant_id" : "40356068" }
{ "address" : { "zipcode" : "11224" }, "borough" : "Brooklyn", "name" : "Riviera Caterer", "restaurant_id" : "40356018" }
{ "address" : { "zipcode" : "11369" }, "borough" : "Queens", "name" : "Brunos On The Boulevard", "restaurant_id" : "40356151" }
{ "address" : { "zipcode" : "10314" }, "borough" : "Staten Island", "name" : "Kosher Island", "restaurant_id" : "40356442" }
{ "address" : { "zipcode" : "11234" }, "borough" : "Brooklyn", "name" : "Wilken'S Fine Food", "restaurant_id" : "40356483" }
{ "address" : { "zipcode" : "11226" }, "borough" : "Brooklyn", "name" : "Taste The Tropics Ice Cream", "restaurant_id" : "40356731" }
{ "address" : { "zipcode" : "11219" }, "borough" : "Brooklyn", "name" : "Regina Caterers", "restaurant_id" : "40356649" }
{ "address" : { "zipcode" : "10460" }, "borough" : "Bronx", "name" : "Wild Asia", "restaurant_id" : "40357217" }
{ "address" : { "zipcode" : "11214" }, "borough" : "Brooklyn", "name" : "C & C Catering Service", "restaurant_id" : "40357437" }
{ "address" : { "zipcode" : "11208" }, "borough" : "Brooklyn", "name" : "May May Kitchen", "restaurant_id" : "40358429" }
{ "address" : { "zipcode" : "10065" }, "borough" : "Manhattan", "name" : "1 East 66Th Street Kitchen", "restaurant_id" : "40359480" }
{ "address" : { "zipcode" : "11225" }, "borough" : "Brooklyn", "name" : "Wendy'S", "restaurant_id" : "30112340" }
{ "address" : { "zipcode" : "10012" }, "borough" : "Manhattan", "name" : "Angelika Film Center", "restaurant_id" : "40362274" }
{ "address" : { "zipcode" : "11205" }, "borough" : "Brooklyn", "name" : "White Castle", "restaurant_id" : "40362344" }
{ "address" : { "zipcode" : "10025" }, "borough" : "Manhattan", "name" : "P & S Deli Grocery", "restaurant_id" : "40362264" }
{ "address" : { "zipcode" : "11368" }, "borough" : "Queens", "name" : "Ho Mei Restaurant", "restaurant_id" : "40362432" }
{ "address" : { "zipcode" : "10005" }, "borough" : "Manhattan", "name" : "The Country Cafe", "restaurant_id" : "40362715" }
{ "address" : { "zipcode" : "11203" }, "borough" : "Brooklyn", "name" : "Shashemene Int'L Restaura", "restaurant_id" : "40362869" }
Type "it" for more
```

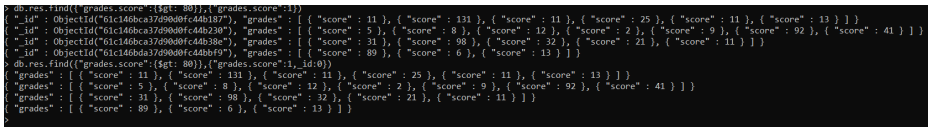
5. Write a MongoDB query to display all the restaurant which is in the borough Bronx.

```
> db.res.find({"borough" : "Bronx"}).limit(5)
{ "_id" : ObjectId("61c146bca37d90d0fc44b017"), "address" : { "building" : "1007", "coord" : [ -73.856077, 40.848447 ], "street" : "Morris Park Ave", "zipcode" : "10462" }, "borough" : "Bronx", "cuisine" : "Bake
ry", "grades" : [ { "date" : ISODate("2014-03-03T00:00:00Z"), "grade" : "A", "score" : 2 }, { "date" : ISODate("2013-09-11T00:00:00Z"), "grade" : "A", "score" : 6 }, { "date" : ISODate("2013-01-24T00:00:00Z"), "
grade" : "A", "score" : 10 }, { "date" : ISODate("2011-11-23T00:00:00Z"), "grade" : "A", "score" : 9 }, { "date" : ISODate("2011-03-10T00:00:00Z"), "grade" : "B", "score" : 14 } ], "name" : "Morris Park Bake Sho
p", "restaurant_id" : "30075445" }
{ "_id" : ObjectId("61c146bca37d90d0fc44b03c"), "address" : { "building" : "2300", "coord" : [ -73.8786113, 40.8502883 ], "street" : "Southern Boulevard", "zipcode" : "10460" }, "borough" : "Bronx", "cuisine" :
"American ", "grades" : [ { "date" : ISODate("2014-05-28T00:00:00Z"), "grade" : "A", "score" : 11 }, { "date" : ISODate("2013-06-19T00:00:00Z"), "grade" : "A", "score" : 4 }, { "date" : ISODate("2012-06-15T00:00
:00Z"), "grade" : "A", "score" : 13 } ], "name" : "Wild Asia", "restaurant_id" : "40357217" }
{ "_id" : ObjectId("61c146bca37d90d0fc44b048"), "address" : { "building" : "1006", "coord" : [ -73.8485687000002, 40.8903781 ], "street" : "East 233 Street", "zipcode" : "10466" }, "borough" : "Bronx", "cuisine
" : "Ice Cream, Gelato, Yogurt, Ices", "grades" : [ { "date" : ISODate("2014-04-24T00:00:00Z"), "grade" : "A", "score" : 10 }, { "date" : ISODate("2013-09-05T00:00:00Z"), "grade" : "A", "score" : 10 }, { "date"
: ISODate("2013-02-21T00:00:00Z"), "grade" : "A", "score" : 9 }, { "date" : ISODate("2012-07-03T00:00:00Z"), "grade" : "A", "score" : 11 }, { "date" : ISODate("2011-07-11T00:00:00Z"), "grade" : "A", "score" : 5
} ], "name" : "Carvel Ice Cream", "restaurant_id" : "40363093" }
{ "_id" : ObjectId("61c146bca37d90d0fc44b04c"), "address" : { "building" : "1236", "coord" : [ -73.8893654, 40.81376179999999 ], "street" : "238 Spofford Ave", "zipcode" : "10474" }, "borough" : "Bronx", "cuisin
e" : "Chinese", "grades" : [ { "date" : ISODate("2013-12-10T00:00:00Z"), "grade" : "A", "score" : 9 }, { "date" : ISODate("2013-01-08T00:00:00Z"), "grade" : "A", "score" : 10 }, { "date" : ISODate("2012-06-12T00
:00:00Z"), "grade" : "B", "score" : 15 } ], "name" : "Happy Garden", "restaurant_id" : "40363289" }
{ "_id" : ObjectId("61c146bca37d90d0fc44b05f"), "address" : { "building" : "277", "coord" : [ -73.8941893, 40.8634684 ], "street" : "East Kingsbridge Road", "zipcode" : "10458" }, "borough" : "Bronx", "cuisine"
: "Chinese", "grades" : [ { "date" : ISODate("2014-03-03T00:00:00Z"), "grade" : "A", "score" : 10 }, { "date" : ISODate("2013-09-26T00:00:00Z"), "grade" : "A", "score" : 10 }, { "date" : ISODate("2013-03-19T00:0
0:00Z"), "grade" : "A", "score" : 10 }, { "date" : ISODate("2012-08-29T00:00:00Z"), "grade" : "A", "score" : 11 }, { "date" : ISODate("2011-08-17T00:00:00Z"), "grade" : "A", "score" : 13 } ], "name" : "Happy Gar
den", "restaurant_id" : "40364296" }
```

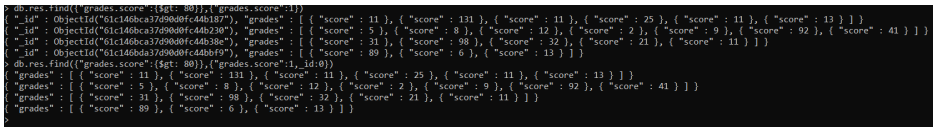
6. Write a MongoDB query to display the first 5 restaurant which is in the borough Bronx.

```
> db.res.find({"borough" : "Bronx"}).limit(5).skip(5)
{ "_id" : ObjectId("61c146bca37d90d0fc44b006"), "address" : { "building" : "658", "coord" : [ -73.81363999999999, 40.82941100000001 ], "street" : "Clarence Ave", "zipcode" : "10465" }, "borough" : "Bronx", "cuis
ine" : "American ", "grades" : [ { "date" : ISODate("2014-06-21T00:00:00Z"), "grade" : "A", "score" : 5 }, { "date" : ISODate("2012-07-11T00:00:00Z"), "grade" : "A", "score" : 10 } ], "name" : "Manhem Club", "re
staurant_id" : "40364363" }
{ "_id" : ObjectId("61c146bca37d90d0fc44b083"), "address" : { "building" : "2222", "coord" : [ -73.84971759999999, 40.8384811 ], "street" : "Haviland Avenue", "zipcode" : "10462" }, "borough" : "Bronx", "cuisine
" : "American ", "grades" : [ { "date" : ISODate("2014-12-18T00:00:00Z"), "grade" : "A", "score" : 7 }, { "date" : ISODate("2014-05-01T00:00:00Z"), "grade" : "B", "score" : 17 }, { "date" : ISODate("2013-03-14T0
0:00:00Z"), "grade" : "A", "score" : 12 }, { "date" : ISODate("2012-09-20T00:00:00Z"), "grade" : "A", "score" : 9 }, { "date" : ISODate("2012-02-08T00:00:00Z"), "grade" : "B", "score" : 19 } ], "name" : "The New
 Starling Athletic Club Of The Bronx", "restaurant_id" : "40364956" }
{ "_id" : ObjectId("61c146bca37d90d0fc44b099"), "address" : { "building" : "72", "coord" : [ -73.92506, 40.8275556 ], "street" : "East  161 Street", "zipcode" : "10451" }, "borough" : "Bronx", "cuisine" : "Ameri
can ", "grades" : [ { "date" : ISODate("2014-04-15T00:00:00Z"), "grade" : "A", "score" : 9 }, { "date" : ISODate("2013-11-14T00:00:00Z"), "grade" : "A", "score" : 4 }, { "date" : ISODate("2013-07-29T00:00:00Z"),
 "grade" : "A", "score" : 10 }, { "date" : ISODate("2012-12-31T00:00:00Z"), "grade" : "B", "score" : 15 }, { "date" : ISODate("2012-05-30T00:00:00Z"), "grade" : "A", "score" : 13 }, { "date" : ISODate("2012-01-0
3T00:00:00Z"), "grade" : "A", "score" : 10 }, { "date" : ISODate("2011-08-15T00:00:00Z"), "grade" : "C", "score" : 37 } ], "name" : "Yankee Tavern", "restaurant_id" : "40365499" }
{ "_id" : ObjectId("61c146bca37d90d0fc44b0ab"), "address" : { "building" : "331", "coord" : [ -73.87786539999999, 40.8724377 ], "street" : "East  204 Street", "zipcode" : "10467" }, "borough" : "Bronx", "cuisine
" : "Irish", "grades" : [ { "date" : ISODate("2014-08-26T00:00:00Z"), "grade" : "A", "score" : 10 }, { "date" : ISODate("2014-03-26T00:00:00Z"), "grade" : "B", "score" : 23 }, { "date" : ISODate("2013-09-11T00:0
0:00Z"), "grade" : "A", "score" : 13 }, { "date" : ISODate("2012-12-18T00:00:00Z"), "grade" : "B", "score" : 27 }, { "date" : ISODate("2011-10-20T00:00:00Z"), "grade" : "A", "score" : 13 } ], "name" : "Mcdwyers
Pub", "restaurant_id" : "40365893" }
{ "_id" : ObjectId("61c146bca37d90d0fc44b0c5"), "address" : { "building" : "5820", "coord" : [ -73.9002615, 40.885186 ], "street" : "Broadway", "zipcode" : "10463" }, "borough" : "Bronx", "cuisine" : "American "
, "grades" : [ { "date" : ISODate("2014-02-26T00:00:00Z"), "grade" : "A", "score" : 5 }, { "date" : ISODate("2013-10-09T00:00:00Z"), "grade" : "B", "score" : 19 }, { "date" : ISODate("2013-05-15T00:00:00Z"), "gr
ade" : "A", "score" : 9 }, { "date" : ISODate("2012-11-20T00:00:00Z"), "grade" : "B", "score" : 18 }, { "date" : ISODate("2011-10-17T00:00:00Z"), "grade" : "A", "score" : 10 }, { "date" : ISODate("2011-06-22T00:
00:00Z"), "grade" : "C", "score" : 35 } ], "name" : "The Punch Bowl", "restaurant_id" : "40366497" }
```

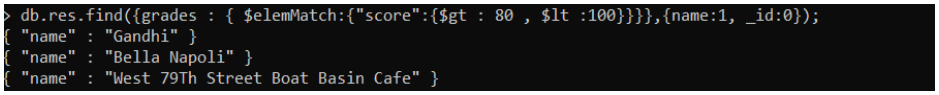
7. Write a MongoDB query to display the next 5 restaurants after skipping first 5 which are in the borough Bronx



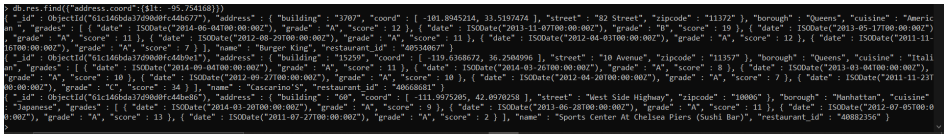
8. Write a MongoDB query to find the restaurants who achieved a score more than 90.



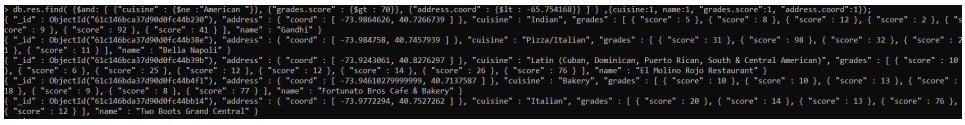
9. Write a MongoDB query to find the restaurants that achieved a score, more than 80 but less than 100



10. Write a MongoDB query to find the restaurants which locate in latitude value less than -95.754168.



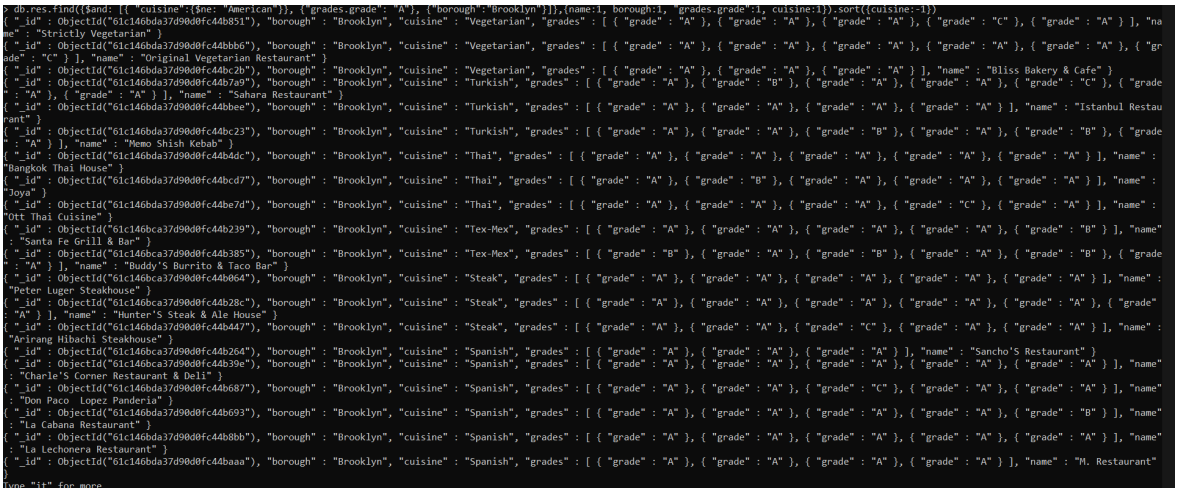
11. Write a MongoDB query to find the restaurants that do not prepare any cuisine of 'American' and their grade score more than 70 and latitude less than -65.754168.



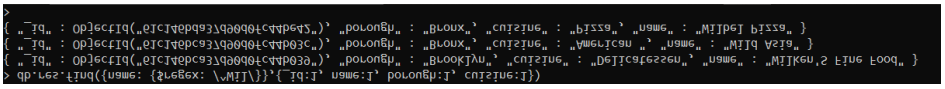
12. Write a MongoDB query to find the restaurants which do not prepare any cuisine of 'American' and achieved a score more than 70 and located in the longitude less than -65.754168.
Note : Do this query without using $and operator.

```
> db.res.find({"cuisine":{$ne:"American "}, "address.coord":{$lt: -65.754168}, "grades.score": {$gt:70}},{name:1, cuisi
e:1, _id:0})
{ "cuisine" : "Indian", "name" : "Gandhi" }
{ "cuisine" : "Pizza/Italian", "name" : "Bella Napoli" }
{ "cuisine" : "Latin (Cuban, Dominican, Puerto Rican, South & Central American)", "name" : "El Molino Rojo Restaurant"

{ "cuisine" : "Bakery", "name" : "Fortunato Bros Cafe & Bakery" }
{ "cuisine" : "Italian", "name" : "Two Boots Grand Central" }
>
```

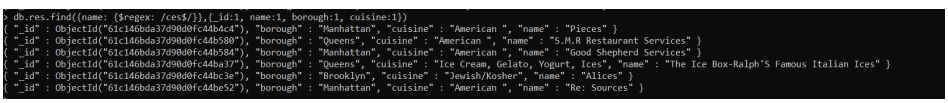
13. Write a MongoDB query to find the restaurants which do not prepare any cuisine of 'American ' and achieved a grade point 'A' not belongs to the borough Brooklyn. The document must be displayed according to the cuisine in descending order.



14. Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those restaurants which contain 'Wil' as first three letters for its name.



15. Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those restaurants which contain 'ces' as last three letters for its name.



16. Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those restaurants which contain 'Reg' as three letters somewhere in its name.

```
> db.res.find({name: {$regex: /Reg/}},{_id:1, name:1, borough:1, cuisine:1})
{ "_id" : ObjectId("61c146bca37d90d0fc44b03b"), "borough" : "Brooklyn", "cuisine" : "American ", "name" : "Regina Caterers" }
{ "_id" : ObjectId("61c146bca37d90d0fc44b134"), "borough" : "Manhattan", "cuisine" : "Café/Coffee/Tea", "name" : "Caffe Reggio" }
{ "_id" : ObjectId("61c146bca37d90d0fc44b23f"), "borough" : "Manhattan", "cuisine" : "American ", "name" : "Regency Hotel" }
{ "_id" : ObjectId("61c146bda37d90d0fc44b560"), "borough" : "Manhattan", "cuisine" : "American ", "name" : "Regency Whist Club" }
{ "_id" : ObjectId("61c146bda37d90d0fc44b640"), "borough" : "Queens", "cuisine" : "American ", "name" : "Rego Park Cafe" }
{ "_id" : ObjectId("61c146bda37d90d0fc44bcaf"), "borough" : "Queens", "cuisine" : "Pizza", "name" : "Regina Pizza" }
{ "_id" : ObjectId("61c146bda37d90d0fc44bee0"), "borough" : "Manhattan", "cuisine" : "American ", "name" : "Regal Entertainment Group" }
>
```

17. Write a MongoDB query to find the restaurants which belong to the borough Bronx and prepared either American or Chinese dish

```
> db.res.find({borough:"Bronx" ,$or:[{cuisine:"American"}, {cuisine:"Chinese"}]}, {name:1,borough:1, cuisine:1, _id:0})
{ "borough" : "Bronx", "cuisine" : "Chinese", "name" : "Happy Garden" }
{ "borough" : "Bronx", "cuisine" : "Chinese", "name" : "Happy Garden" }
{ "borough" : "Bronx", "cuisine" : "Chinese", "name" : "China Wok Ii" }
{ "borough" : "Bronx", "cuisine" : "Chinese", "name" : "Dragon City" }
{ "borough" : "Bronx", "cuisine" : "Chinese", "name" : "Great Wall Restaurant" }
{ "borough" : "Bronx", "cuisine" : "Chinese", "name" : "Hunan Balcony" }
{ "borough" : "Bronx", "cuisine" : "Chinese", "name" : "Lucky House Restaurant" }
{ "borough" : "Bronx", "cuisine" : "Chinese", "name" : "New Wah Kitchen" }
{ "borough" : "Bronx", "cuisine" : "Chinese", "name" : "New Hing Restaurant" }
{ "borough" : "Bronx", "cuisine" : "Chinese", "name" : "Hong Kong Restaurant" }
{ "borough" : "Bronx", "cuisine" : "Chinese", "name" : "Kristy'S Restaurant" }
{ "borough" : "Bronx", "cuisine" : "Chinese", "name" : "East Dynasty" }
{ "borough" : "Bronx", "cuisine" : "Chinese", "name" : "Lin Home Chinese Restaura" }
{ "borough" : "Bronx", "cuisine" : "Chinese", "name" : "Peacock Restaurant" }
{ "borough" : "Bronx", "cuisine" : "Chinese", "name" : "Lin'S Garden" }
{ "borough" : "Bronx", "cuisine" : "Chinese", "name" : "New Rainbow Restaurant" }
>
```

18. Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those restaurants which belong to the borough Staten Island or Queens or Bronxor Brooklyn.

```
> db.res.find({ borough: {$in: [ "Staten Island", "Queens", "Bronxor Brooklyn"]}}, {name:1, borough:1, cuisine:1, _id:0})
{ "borough" : "Queens", "cuisine" : "Jewish/Kosher", "name" : "Tov Kosher Kitchen" }
{ "borough" : "Queens", "cuisine" : "American ", "name" : "Brunos On The Boulevard" }
{ "borough" : "Staten Island", "cuisine" : "Jewish/Kosher", "name" : "Kosher Island" }
{ "borough" : "Queens", "cuisine" : "Chinese", "name" : "Ho Mei Restaurant" }
{ "borough" : "Queens", "cuisine" : "Delicatessen", "name" : "Tony'S Deli" }
{ "borough" : "Staten Island", "cuisine" : "Delicatessen", "name" : "Bagels N Buns" }
{ "borough" : "Queens", "cuisine" : "Bagels/Pretzels", "name" : "Hot Bagels" }
{ "borough" : "Queens", "cuisine" : "American ", "name" : "Snack Time Grill" }
{ "borough" : "Staten Island", "cuisine" : "Ice Cream, Gelato, Yogurt, Ices", "name" : "Carvel Ice Cream" }
{ "borough" : "Queens", "cuisine" : "American ", "name" : "Terminal Cafe/Yankee Clipper" }
{ "borough" : "Staten Island", "cuisine" : "Delicatessen", "name" : "Plaza Bagels & Deli" }
{ "borough" : "Staten Island", "cuisine" : "Delicatessen", "name" : "B & M Hot Bagel & Grocery" }
{ "borough" : "Queens", "cuisine" : "German", "name" : "Gottscheer Hall" }
{ "borough" : "Queens", "cuisine" : "Jewish/Kosher", "name" : "Ben-Best Deli & Restaurant" }
{ "borough" : "Staten Island", "cuisine" : "American ", "name" : "Great Kills Yacht Club" }
{ "borough" : "Queens", "cuisine" : "Delicatessen", "name" : "Sal'S Deli" }
{ "borough" : "Queens", "cuisine" : "Pizza/Italian", "name" : "New Park Pizzeria & Restaurant" }
{ "borough" : "Queens", "cuisine" : "American ", "name" : "Douglaston Club" }
{ "borough" : "Queens", "cuisine" : "Pizza", "name" : "Rizzo'S Fine Pizza" }
{ "borough" : "Staten Island", "cuisine" : "Italian", "name" : "Crystal Room" }
Type "it" for more
```

19. Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those restaurants which are not belonging to the borough Staten Island or Queens or Bronxor Brooklyn.

```
> db.res.find({ borough: {$nin: [ "Staten Island", "Queens", "Bronxor Brooklyn"]}}, {name:1, borough:1, cuisine:1, _id:0})
{ "borough" : "Bronx", "cuisine" : "Bakery", "name" : "Morris Park Bake Shop" }
{ "borough" : "Manhattan", "cuisine" : "Irish", "name" : "Dj Reynolds Pub And Restaurant" }
{ "borough" : "Brooklyn", "cuisine" : "American ", "name" : "Riviera Caterer" }
{ "borough" : "Brooklyn", "cuisine" : "Delicatessen", "name" : "Wilken'S Fine Food" }
{ "borough" : "Brooklyn", "cuisine" : "Ice Cream, Gelato, Yogurt, Ices", "name" : "Taste The Tropics Ice Cream" }
{ "borough" : "Brooklyn", "cuisine" : "American ", "name" : "Regina Caterers" }
{ "borough" : "Bronx", "cuisine" : "American ", "name" : "Wild Asia" }
{ "borough" : "Brooklyn", "cuisine" : "American ", "name" : "C & C Catering Service" }
{ "borough" : "Brooklyn", "cuisine" : "Chinese", "name" : "May May Kitchen" }
{ "borough" : "Manhattan", "cuisine" : "American ", "name" : "1 East 66Th Street Kitchen" }
{ "borough" : "Brooklyn", "cuisine" : "Hamburgers", "name" : "Wendy'S" }
{ "borough" : "Manhattan", "cuisine" : "American ", "name" : "Angelika Film Center" }
{ "borough" : "Brooklyn", "cuisine" : "Hamburgers", "name" : "White Castle" }
{ "borough" : "Manhattan", "cuisine" : "American ", "name" : "P & S Deli Grocery" }
{ "borough" : "Manhattan", "cuisine" : "Turkish", "name" : "The Country Cafe" }
{ "borough" : "Brooklyn", "cuisine" : "Caribbean", "name" : "Shashemene Int'L Restaura" }
{ "borough" : "Manhattan", "cuisine" : "American ", "name" : "Downtown Deli" }
{ "borough" : "Bronx", "cuisine" : "Ice Cream, Gelato, Yogurt, Ices", "name" : "Carvel Ice Cream" }
{ "borough" : "Brooklyn", "cuisine" : "Donuts", "name" : "Dunkin' Donuts" }
{ "borough" : "Brooklyn", "cuisine" : "American ", "name" : "Mejlander & Mulgannon" }
Type "it" for more
```

20. Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those restaurants which prepared dish except 'American' and 'Chinees' or restaurant's name begins with letter 'Wil'.

21. Write a MongoDB query to arrange the name of the restaurants in ascending order along with all the columns

```
> db.res.find({},{"name":1}).sort({"name":1})
{ "_id" : ObjectId("61c146bda37d90d0fc44bcba"), "name" : "(Lewis Drug Store) Locanda Vini E Olii" }
{ "_id" : ObjectId("61c146bca37d90d0fc44b03f"), "name" : "1 East 66Th Street Kitchen" }
{ "_id" : ObjectId("61c146bda37d90d0fc44b853"), "name" : "101 Deli" }
{ "_id" : ObjectId("61c146bda37d90d0fc44b742"), "name" : "101 Restaurant And Bar" }
{ "_id" : ObjectId("61c146bda37d90d0fc44b50a"), "name" : "1020 Bar" }
{ "_id" : ObjectId("61c146bda37d90d0fc44b948"), "name" : "104-01 Foster Avenue Coffee Shop(Ups)" }
{ "_id" : ObjectId("61c146bda37d90d0fc44b9f9"), "name" : "10Th Avenue Pizza & Cafe" }
{ "_id" : ObjectId("61c146bda37d90d0fc44be99"), "name" : "111 Restaurant" }
{ "_id" : ObjectId("61c146bda37d90d0fc44ba69"), "name" : "15 East Restaurant" }
{ "_id" : ObjectId("61c146bca37d90d0fc44b27e"), "name" : "200 Fifth Avenue Restaurant & Sports Bar" }
{ "_id" : ObjectId("61c146bca37d90d0fc44b065"), "name" : "21 Club" }
{ "_id" : ObjectId("61c146bca37d90d0fc44b2ee"), "name" : "2A" }
{ "_id" : ObjectId("61c146bda37d90d0fc44b85f"), "name" : "3 Deli & Grill" }
{ "_id" : ObjectId("61c146bca37d90d0fc44b34c"), "name" : "3 Guys" }
{ "_id" : ObjectId("61c146bda37d90d0fc44b954"), "name" : "3 Guys Resturant" }
{ "_id" : ObjectId("61c146bda37d90d0fc44b8c6"), "name" : "42Nd Street Pizza Diner" }
{ "_id" : ObjectId("61c146bda37d90d0fc44bcd5"), "name" : "44 & X Hell'S Kitchen" }
{ "_id" : ObjectId("61c146bda37d90d0fc44ba65"), "name" : "44 Sw Ristorante & Bar" }
{ "_id" : ObjectId("61c146bca37d90d0fc44b318"), "name" : "5 Burro Cafe" }
{ "_id" : ObjectId("61c146bda37d90d0fc44bcc3"), "name" : "525 Lex Restaurant & Bar" }
Type "it" for more
```

22. Write a MongoDB query to arrange the name of the restaurants in descending along with all the columns.

```
> db.res.find({},{"name":1}).sort({"name":-1})
{ "_id" : ObjectId("61c146bca37d90d0fc44b0f1"), "name" : "Zum Stammtisch" }
{ "_id" : ObjectId("61c146bda37d90d0fc44bc2c"), "name" : "Zum Schneider" }
{ "_id" : ObjectId("61c146bda37d90d0fc44be61"), "name" : "Zorba'S" }
{ "_id" : ObjectId("61c146bda37d90d0fc44bca8"), "name" : "Zebu Grill" }
{ "_id" : ObjectId("61c146bda37d90d0fc44b4ab"), "name" : "Zaro'S Bread Basket" }
{ "_id" : ObjectId("61c146bda37d90d0fc44b5dd"), "name" : "Zaro'S Bread Basket" }
{ "_id" : ObjectId("61c146bda37d90d0fc44b61d"), "name" : "Zaro'S Bread Basket" }
{ "_id" : ObjectId("61c146bda37d90d0fc44b82d"), "name" : "Zaro'S Bread Basket" }
{ "_id" : ObjectId("61c146bda37d90d0fc44bbd8"), "name" : "Zaro'S Bread Basket" }
{ "_id" : ObjectId("61c146bda37d90d0fc44b59e"), "name" : "Zaro'S Bakery" }
{ "_id" : ObjectId("61c146bda37d90d0fc44ba2d"), "name" : "Zaro'S Bakery" }
{ "_id" : ObjectId("61c146bca37d90d0fc44b17b"), "name" : "Zafi'S Luncheonette" }
{ "_id" : ObjectId("61c146bda37d90d0fc44be6e"), "name" : "Yvonne Yvonne Restaurant" }
{ "_id" : ObjectId("61c146bda37d90d0fc44b972"), "name" : "Yura & Company On Madison" }
{ "_id" : ObjectId("61c146bda37d90d0fc44bc2a"), "name" : "Yummy Kitchen" }
{ "_id" : ObjectId("61c146bda37d90d0fc44b66a"), "name" : "Your Bakery" }
{ "_id" : ObjectId("61c146bda37d90d0fc44b6cf"), "name" : "Yonah Shimmels Knishes" }
{ "_id" : ObjectId("61c146bda37d90d0fc44b871"), "name" : "Yolanda Pizzeria Restaurant" }
{ "_id" : ObjectId("61c146bca37d90d0fc44b2d2"), "name" : "Yip'S" }
{ "_id" : ObjectId("61c146bca37d90d0fc44b383"), "name" : "Yen Yen Restaurant" }
Type "it" for more
```

23. Write a MongoDB query to arrange the name of the cuisine in ascending order and for that same cuisine borough should be in descending order

```
> db.res.find({},{"name":1, borough:1, cuisine:1, _id:0}).sort({cuisine:1, borough:-1})
{ "borough" : "Manhattan", "cuisine" : "Afghan", "name" : "Afghan Kebab House" }
{ "borough" : "Manhattan", "cuisine" : "Afghan", "name" : "Khyber Pass" }
{ "borough" : "Manhattan", "cuisine" : "Afghan", "name" : "Afghan Kebab House #1" }
{ "borough" : "Manhattan", "cuisine" : "Afghan", "name" : "Ariana Kebab House" }
{ "borough" : "Queens", "cuisine" : "African", "name" : "Africana Restaurant" }
{ "borough" : "Brooklyn", "cuisine" : "African", "name" : "Madiba" }
{ "borough" : "Bronx", "cuisine" : "African", "name" : "African Terrace" }
{ "borough" : "Bronx", "cuisine" : "African", "name" : "Ebe Ye Yie African Restaurant" }
{ "borough" : "Staten Island", "cuisine" : "American ", "name" : "Great Kills Yacht Club" }
{ "borough" : "Staten Island", "cuisine" : "American ", "name" : "Labetti'S Post # 2159" }
{ "borough" : "Staten Island", "cuisine" : "American ", "name" : "Joyce'S Tavern" }
{ "borough" : "Staten Island", "cuisine" : "American ", "name" : "Li Greci'S Staaten Restaurant" }
{ "borough" : "Staten Island", "cuisine" : "American ", "name" : "Richmond County Country Club" }
{ "borough" : "Staten Island", "cuisine" : "American ", "name" : "South Shore Swimming Club" }
{ "borough" : "Staten Island", "cuisine" : "American ", "name" : "Buddy'S Wonder Bar" }
{ "borough" : "Staten Island", "cuisine" : "American ", "name" : "Colonnade Diner" }
{ "borough" : "Staten Island", "cuisine" : "American ", "name" : "Jody'S Club" }
{ "borough" : "Staten Island", "cuisine" : "American ", "name" : "Perkins Family Restaurant & Bakery" }
{ "borough" : "Staten Island", "cuisine" : "American ", "name" : "Schaffer'S Tavern" }
{ "borough" : "Staten Island", "cuisine" : "American ", "name" : "Golden Dove Diner" }
Type "it" for more
```

24. Find out how many times each cuisine is offered at various restaurants.

```
> db.res.aggregate([{$group:{_id:"$cuisine", count:{$sum:1}}}]).pretty()
{ "_id" : "German", "count" : 9 }
{ "_id" : "French", "count" : 72 }
{ "_id" : "Mexican", "count" : 73 }
{ "_id" : "Not Listed/Not Applicable", "count" : 1 }
{
        "_id" : "Bottled beverages, including water, sodas, juices, etc.",
        "count" : 10
}
{ "_id" : "Chinese", "count" : 115 }
{ "_id" : "Thai", "count" : 14 }
{ "_id" : "Filipino", "count" : 3 }
{ "_id" : "Pakistani", "count" : 1 }
{ "_id" : "Peruvian", "count" : 2 }
{ "_id" : "Pizza", "count" : 270 }
{ "_id" : "Bagels/Pretzels", "count" : 34 }
{ "_id" : "Afghan", "count" : 4 }
{ "_id" : "American ", "count" : 1255 }
{ "_id" : "Indian", "count" : 43 }
{
        "_id" : "Latin (Cuban, Dominican, Puerto Rican, South & Central American)",
        "count" : 115
}
{ "_id" : "Continental", "count" : 8 }
{ "_id" : "Chinese/Japanese", "count" : 1 }
{ "_id" : "Mediterranean", "count" : 16 }
{ "_id" : "Indonesian", "count" : 2 }
Type "it" for more
```

25. Find out how many times each cuisine is offered at various restaurants in descending order.

```
> db.res.aggregate([{$group:{_id:"$cuisine","Total":{$sum:1}}},{$sort:{Total:-1}}])
{ "_id" : "American ", "Total" : 1255 }
{ "_id" : "Italian", "Total" : 325 }
{ "_id" : "Pizza", "Total" : 270 }
{ "_id" : "Café/Coffee/Tea", "Total" : 180 }
{ "_id" : "Hamburgers", "Total" : 159 }
{ "_id" : "Bakery", "Total" : 127 }
{ "_id" : "Chinese", "Total" : 115 }
{ "_id" : "Latin (Cuban, Dominican, Puerto Rican, South & Central American)", "Total" : 115 }
{ "_id" : "Pizza/Italian", "Total" : 106 }
{ "_id" : "Japanese", "Total" : 80 }
{ "_id" : "Irish", "Total" : 79 }
{ "_id" : "Delicatessen", "Total" : 78 }
{ "_id" : "Caribbean", "Total" : 75 }
{ "_id" : "Mexican", "Total" : 73 }
{ "_id" : "French", "Total" : 72 }
{ "_id" : "Jewish/Kosher", "Total" : 60 }
{ "_id" : "Donuts", "Total" : 43 }
{ "_id" : "Indian", "Total" : 43 }
{ "_id" : "Spanish", "Total" : 42 }
{ "_id" : "Seafood", "Total" : 35 }
Type "it" for more
```

26. Which cuisine is highly offered among all restaurants?

```
> db.res.aggregate([{$group:{_id:"$cuisine","Total":{$sum:1}}},{$sort:{Total:-1}},{$limit:1}])
{ "_id" : "American ", "Total" : 1255 }
```

27. Find out the top 5 highly offered cuisines among all restaurants?

```
> db.res.aggregate([{$group:{_id:"$cuisine","Total":{$sum:1}}},{$sort:{Total:-1}},{$limit:5}])
{ "_id" : "American ", "Total" : 1255 }
{ "_id" : "Italian", "Total" : 325 }
{ "_id" : "Pizza", "Total" : 270 }
{ "_id" : "Café/Coffee/Tea", "Total" : 180 }
{ "_id" : "Hamburgers", "Total" : 159 }
>
```

Tutorial 3

1. Create and Emit a custom event that checks whether the age of the person is greater than 18 ornot depending on the date of birth passed to the event.

```
var events = require('events');
var em = new events.EventEmitter();
em.on('Age',function(year){
        C = 2022;
        user = C - year;
        if(user > 18){
            console.log("AGE : "+ user);
            console.log("Eligible");
        }
        else{
            console.log("AGE : "+ user);
            console.log("Not Eligible");
        }
    });
em.emit('Age',2002);
```

Output:

```
PS C:\Users\i\Desktop\Tutorial 03> node index
AGE : 20
Eligible
PS C:\Users\i\Desktop\Tutorial 03> node index
AGE : 17
Not Eligible
PS C:\Users\i\Desktop\Tutorial 03>
```

2. Create a node script that gets the parameters using the GET method from the form.html file and log it on the console.

Code:

Index

.js

```
var http=require('http');
var url=require('url');
var querystring=require('querystring');;
var server=http.createServer((req,res)=>
{
 query=url.parse(req.url).query;
 user_name=querystring.parse(query)['uname'];
 branch_name=querystring.parse(query)['bname'];
```

```
 console.log(user_name);
console.log(branch_name);
}).listen(3000);
```

Index.html

```
<html lang="en">
<head>
 <meta charset="UTF-8">
 <meta http-equiv="X-UA-Compatible" content="IE=edge">
 <meta name="viewport" content="width=device-width, initial-scale=1.0">
 <title>Document</title>
</head>
<body>
 <form method="get" action="http://localhost:3000/submit">
 Name: <input type="text" name="uname"><br><br>
 Branch: <input type="text" name="bname" ><br><br>
 <input type="submit" value="Submit" name="submit">
 </form>
</body>
</html>
```

3. Create a node script that gets the parameters using POST method from the form.html file and log it on console.

Code:
Index
.js

```
var http=require('http')
var querystring=require('querystring')
var server=http.createServer((req,res)=>{
merge_data=''
req.on('data',(maindata)=>{
merge_data+=maindata
 })
 req.on('end',()=>{
 uname=querystring.parse(merge_data)['uname']
 bname=querystring.parse(merge_data)['bname']
 console.log(uname);
 console.log(bname);
 })
}).listen(3000);
```

```html
<html lang="en">
<head>
 <meta charset="UTF-8">
 <meta http-equiv="X-UA-Compatible" content="IE=edge">
 <meta name="viewport" content="width=device-width, initial-scale=1.0">
 <title>Document</title>
</head>
<body>
 <form method="post" action="http://localhost:3000/submit">
 Name: <input type="text" name="uname"><br><br>
 Branch: <input type="text" name="bname"><br><br>
 <input type="submit" value="Submit" name="submit">
 </form>
</body>
</html>
```

4. Create mongodb for students and nodejs that connects to mongodb using mongojs module.(install monojs and nodemon packages). [Student document must contain: s_id,s_name,s_branch, s_city, s_mobilenos, s_add].

Code:

```
var express=require('express')
var mongoose=require('mongoose')
mongoose.connect('mongodb://localhost:27017/students').then(()=>{
 console.log("Database Connected");
 var app=express()
 app.listen(3000,()=>{
 console.log('Server Started');
 })
})
```

Output:

```
PS C:\Users\i\Desktop\Tutorial 03> node index
Database Connected
Server Started
```

Tutorial 4

1. **Create a NodeAPI to get products from the database using mongoose library.**

Code:

Index.js

```
var express=require('express')
var mongoose=require('mongoose')
var route=require('./routes')

mongoose.connect('mongodb://localhost:27017/products').then(()=>{
    console.log("Database Connected");
    var app=express()
    app.use('/api',route)
    app.listen(3000,()=>
    {
        console.log('Server Started on 3000 Port');
    })
}).catch((e)=>{
    console.log(e.toString())
})
```

routes.js

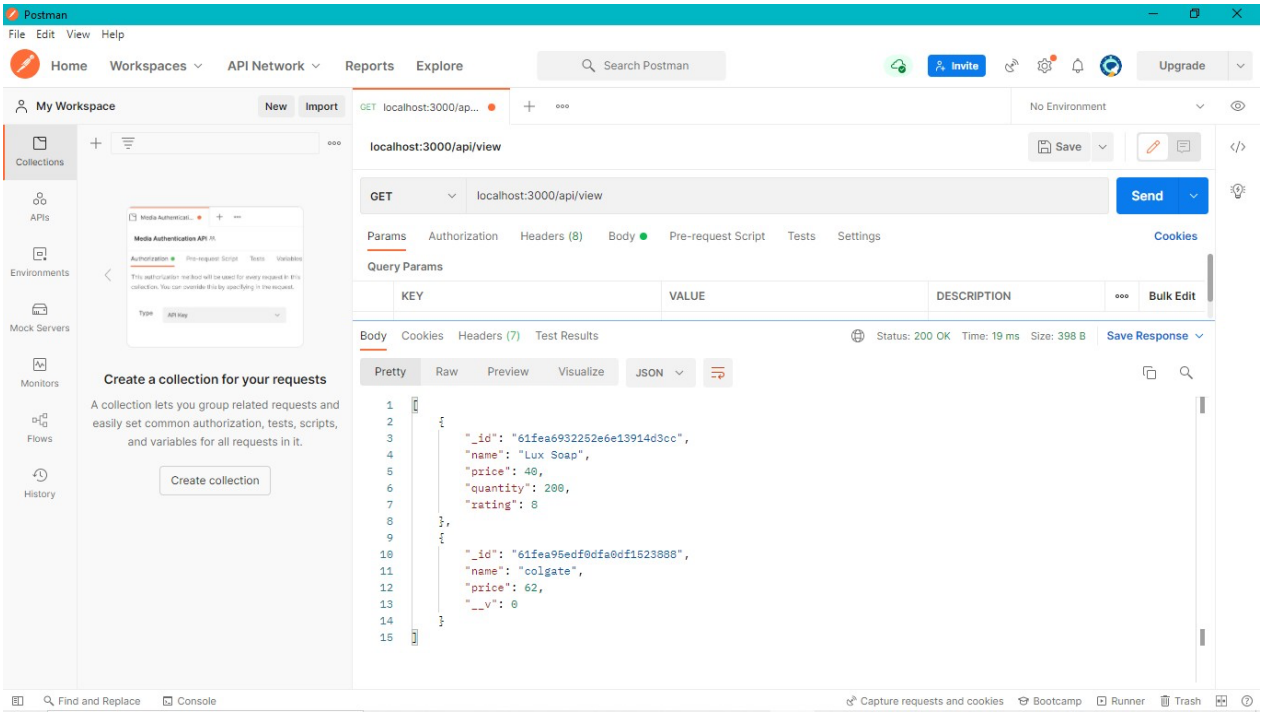
```
var express = require('express');
var router = express.Router();
var product= require('./models/productmodel')


router.get('/view',async(req,res)=>{
    const data = await product.find()
    res.send(data)
    console.log(data)

})
module.exports = router
```

./models/productmodel.js

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

var schema= mongoose.Schema({
    name:String,
    price:Number
```

```
    quantity:Number
    ,rating:Number
})

module.exports = mongoose.model("products",schema)
```

Output:



2. **Create a NodeAPI to insert records into products database using mongooselibrary**.

   Code:

   Index.
   js

```
var express=require('express')
var mongoose=require('mongoose')
var route=require('./routes')
var bodyParser =require('body-parser')

mongoose.connect('mongodb://localhost:27017/products').then(()=>{
    console.log("Database Connected");
    var app=express()
```

```
   app.use(bodyParser.urlencoded({extended:false}))
    app.use('/api',route)
    app.listen(3000,()=>
    {
        console.log('Server Started on 3000 Port');
    })
}).catch((e)=>{
    console.log(e.toString())
})
```

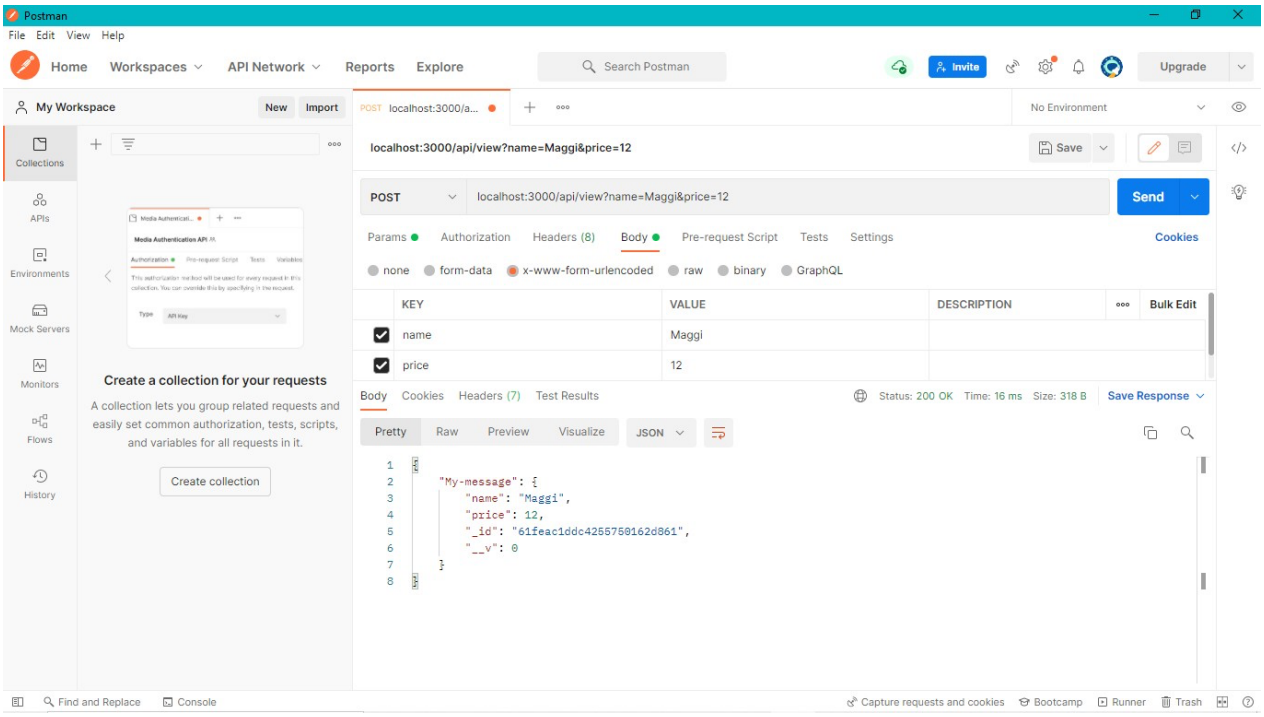
routes.js

```
var express = require('express');
var router = express.Router();
var products= require('./models/productmodel')
router.post("/view",async(req,res)=>{
    const data = new product({
        name:req.body.name,
        price:req.body.price,
        quantity:req.body.quantity,

        rating:req.body.rating

    })

    await data.save((err,msg)=>{
        if(err){
            res.status(500).json({
                "error":err
            })
        }
        else{
            res.status(200).json({
                "My-message":msg
            })
        }
    })
})
module.exports = router
```

./models/productmodel.js

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

var schema= mongoose.Schema({
    name:String,
    price:Number,
    quantity:Number,
    rating:Number
})

module.exports = mongoose.model("products",schema)
```

Output:



3. **Create a NodeAPI to update products from the database using mongoose library.**

Code:

Index

.js

```
var express=require('express')
var mongoose=require('mongoose')
var route=require('./routes')
```

```
mongoose.connect('mongodb://localhost:27017/products').then(()=>{
    console.log("Database Connected");
    var app=express()
    app.use(bodyParser.urlencoded({extended:false}))
    app.use('/api',route)
    app.listen(3000,()=>
    {
        console.log('Server Started on 3000 Port');
    })
}).catch((e)=>{
    console.log(e.toString())
})
```

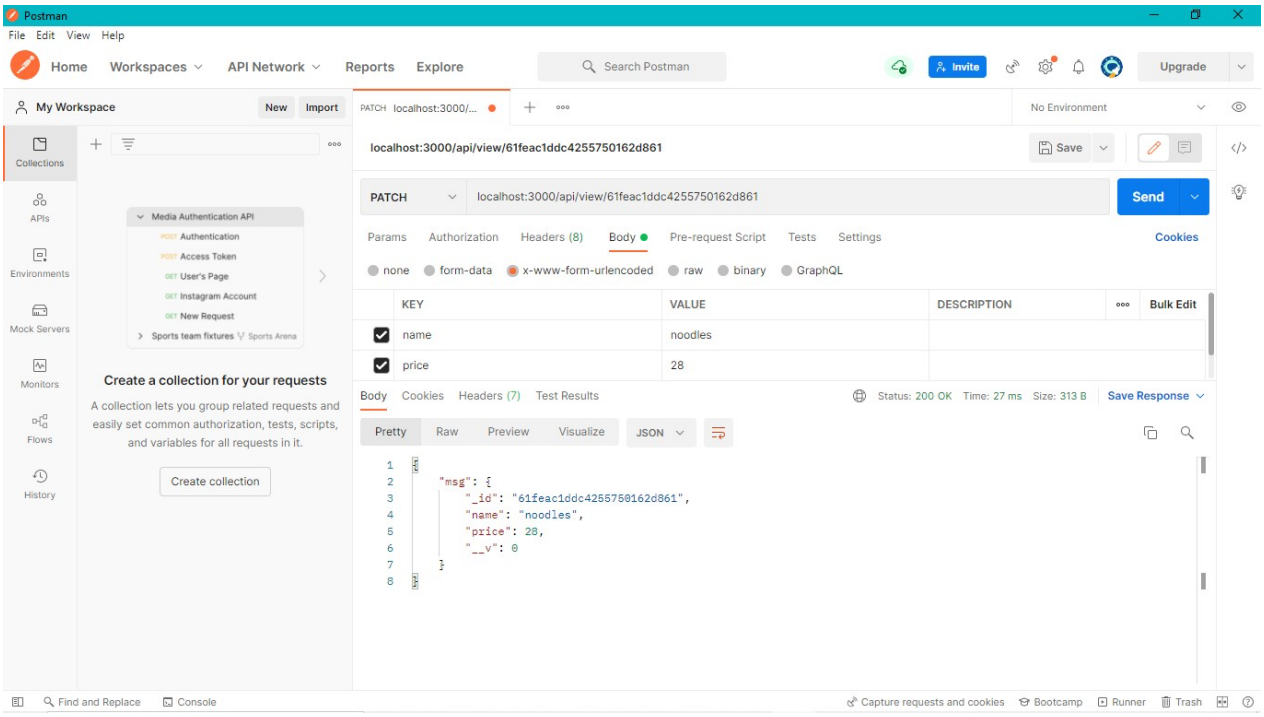
routes.js

```
var express = require('express');
var router = express.Router();
var product= require('./models/productmodel')


router.patch('/view/:id',async (req,res)=>{
    const data = await product.findOne({_id:req.params.id})
    data.name = req.body.name
data.price = req.body.price
data.quantity = req.body.quantity

    data.rating = req.body.rating
    await data.save((err,msg)=>{
        if(err){
            res.status(500).json({
                error:err
            })
        }
        else{
            res.status(200).json({
                msg:msg
            })
        }
    })

})

module.exports = router
```

./models/productmodel.js

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

var schema= mongoose.Schema({
    name:String,
    price:Number,
    quantity:Number,
    rating:Number
})

module.exports = mongoose.model("products",schema)
```

Output:



4. **Create a NodeAPI to delete products from the database using mongoose library.**

Code:

Index.
js

```
var express=require('express')
var mongoose=require('mongoose')
var route=require('./routes')
```

```
mongoose.connect('mongodb://localhost:27017/products').then(()=>{
    console.log("Database Connected");
    var app=express()
    app.use(bodyParser.urlencoded({extended:false}))
    app.use('/api',route)
    app.listen(3000,()=>
    {
        console.log('Server Started on 3000 Port');
    })
}).catch((e)=>{
    console.log(e.toString())
})
```

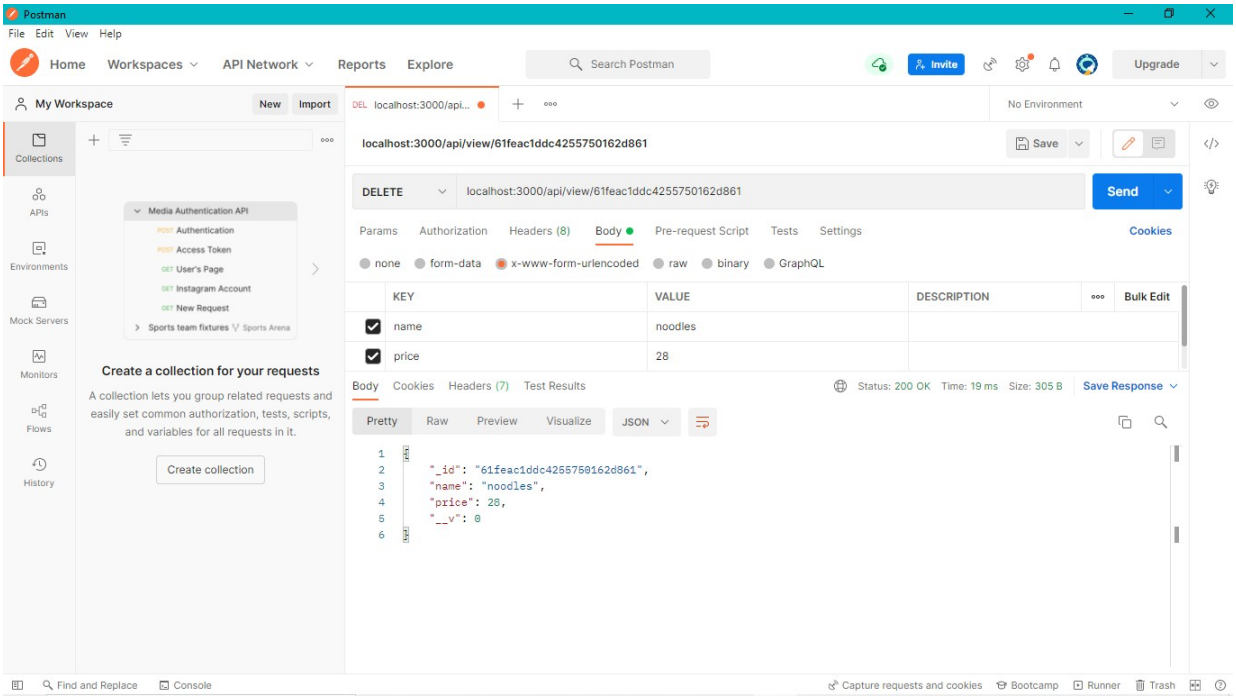
routes.js

```
var express = require('express');
var router = express.Router();
var product= require('./models/productmodel')


router.delete("/view/:id", async (request, response) => {
  const _id = request.params.id;
  const mydata = await product.findByIdAndDelete(_id);
  response.send(mydata);
});
module.exports = router
```

./models/productmodel.js

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

var schema= mongoose.Schema({
    name:String,
    price:Number,
    quantity:Number,
    rating:Number
})

module.exports = mongoose.model("products",schema)
```

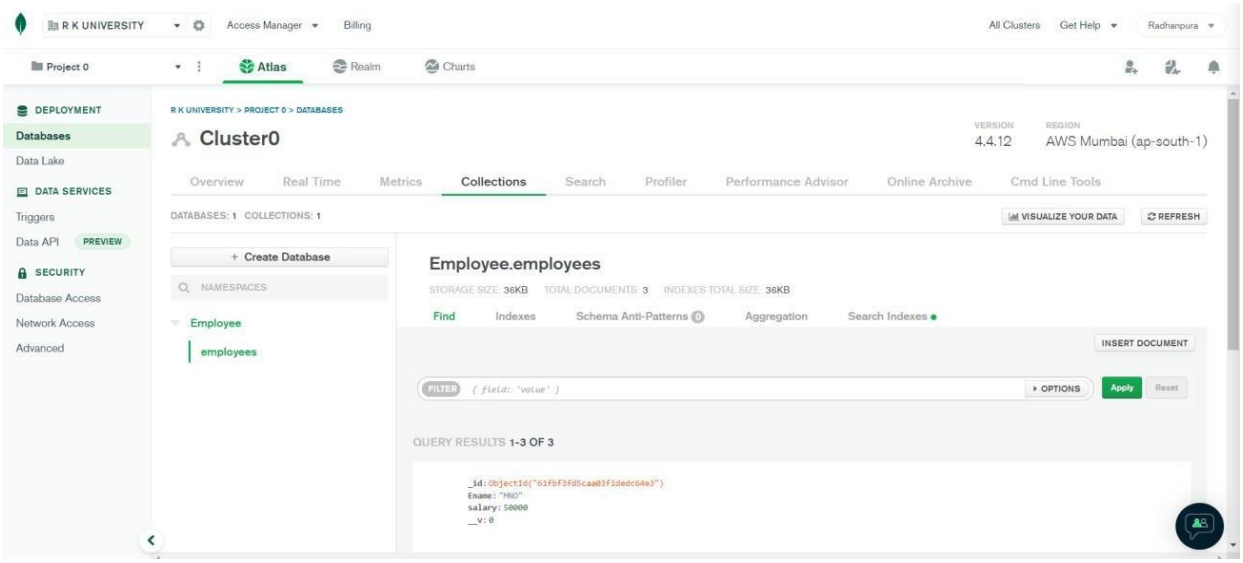
Output:


Postman
File Edit View Help
Home Workspaces API Network Reports Explore
Search Postman
Invite
Upgrade
My Workspace
New Import
Collections
APIs
Environments
Mock Servers
Monitors
Flows
History
Media Authentication API
Authentication
Access Token
User's Page
Instagram Account
New Request
Sports team fixtures
Create a collection for your requests
A collection lets you group related requests and easily set common authorization, tests, scripts, and variables for all requests in it.
Create collection
DEL localhost:3000/api...
No Environment
localhost:3000/api/view/61feac1ddc4255750162d861
Save
DELETE
localhost:3000/api/view/61feac1ddc4255750162d861
Send
Params Authorization Headers (8) Body Pre-request Script Tests Settings
Cookies
none form-data x-www-form-urlencoded raw binary GraphQL
KEY VALUE DESCRIPTION Bulk Edit
name noodles
price 28
Body Cookies Headers (7) Test Results
Status: 200 OK Time: 19 ms Size: 305 B
Save Response
Pretty Raw Preview Visualize JSON
{
"_id": "61feac1ddc4255750162d861",
"name": "noodles",
"price": 28,
"__v": 0
}
Find and Replace
Console
Capture requests and cookies
Bootcamp
Runner
Trash

Tutorial-5

1. Create a database of your choice with at least 2 collections on the mongo cluster.

Output:



2. Create a NodeAPI to perform CRUD operations on the above created collections for thedata stored on mlab cluster.

Code:

index.j

```
var mongoose = require('mongoose')
var express = require('express')
var route = require('./routes')
var bodyParser =require('body-parser')

mongoose.connect('mongodb+srv://KevalSoni:KevalSoni@cluster0.rcfn8.mongodb.net/Em
ployee?retryWrites=true&w=majority').then(()=>{
    console.log('DB Connected')

    app = express();
    app.use(bodyParser.urlencoded({extended:false}))
    app.use('/api',route)
```

s

```
    app.listen((process.env.PORT||3000),()=>{
        console.log('server started')
    })
}).catch((e)=>{
    console.log(e.toString())
})
```

routes.js

```
var express = require('express');
var router = express.Router();
var Employee = require('./Models/Employee')

router.get('/Employee',async(req,res)=>{
    const emp = await Employee.find()
    res.send(emp)
})

router.post("/Employee",async(req,res)=>{
    const emp = new Employee({
        Ename:req.body.Ename,
        salary:req.body.salary
    })

    await emp.save((err,msg)=>{
        if(err){
            res.status(500).json({
                "error":err
            })
        }
        else{
            res.status(200).json({
                "My-message":msg
            })
        }
    })

})

router.patch('/Employee/:id',async(req,res)=>{
    const emp = await Employee.findOne({_id:req.params.id})
    emp.Ename = req.body.Ename
    emp.salary = req.body.salary
    await emp.save((err,msg)=>{
```

```
        if(err){
            res.status(500).json({
                "error":err
            })
        }
        else{
            res.status(200).json({
                "msg":msg
            })
        }
    })
})

router.delete("/Employee/:id",async(req,res)=>{
    const _id = req.params.id;
    const EData = await Employee.findByIdAndDelete(_id);
    res.send(EData);
  });

module.exports = router
```

./Models/Employee.js

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

var employeeSchema = mongoose.Schema({
    Ename:String,
    salary:Number
})

module.exports = mongoose.model("employees",employeeSchema)
```

Output:

```
PS C:\Users\i\Desktop\New folder\EmployeeAPI> node index
DB Connected
server started
```

3. Host the same project on Heroku and need to present the same.

Output:

Tutorial 6

1 Install typescript

Code: npm install -g typescript

OUTPUT :

```
PS D:\RKU-2-11-2020\RKU ALL SEM WORK 2020-21 FOLDER\SEM-6\UI\typescript_tutorials> npm install -g typescript
```

2 Create an arrow function that calculates the sum of n natural numbers. n is passed as a parameter.

Code:

```
var sum = (n)=>{
   return n*(n+1)/2

}

var result = sum(5)

console.log(result)
```

OUTPUT :

```
PS D:\RKU-2-11-2020\RKU ALL SEM WORK 2020-21 FOLDER\SEM-6\UI\typescript_tutorials> tsc t6-2
PS D:\RKU-2-11-2020\RKU ALL SEM WORK 2020-21 FOLDER\SEM-6\UI\typescript_tutorials> node t6-2
15
PS D:\RKU-2-11-2020\RKU ALL SEM WORK 2020-21 FOLDER\SEM-6\UI\typescript_tutorials>
```

3 Create three arrow functions that demonstrate usage of default parameter, optional parameter and rest parameter.

Code:

```
//default parameter
var defaultpara = (a,b=2)=>{
   return a+b
}

var ans = defaultpara(5)
```

```
console.log(ans)

//optional parameter
var optionalpara = (a,b?)=>{
   return a*b
}

var res = optionalpara(5,4)
console.log(res)

//rest parameter
var restpara = (name,...surname)=>{
   return name + " " + surname + " " + ".."
}

var answer = restpara("Kinjal","Shah")
console.log(answer)
```

OUTPUT :

```
PS D:\RKU-2-11-2020\RKU ALL SEM WORK 2020-21 FOLDER\SEM-6\UI\typescript_tutorials> tsc t6-3
PS D:\RKU-2-11-2020\RKU ALL SEM WORK 2020-21 FOLDER\SEM-6\UI\typescript_tutorials> node t6-3
7
20
Kinjal Shah ..
PS D:\RKU-2-11-2020\RKU ALL SEM WORK 2020-21 FOLDER\SEM-6\UI\typescript_tutorials>
```

4 Create an interface called student with name, city, branch properties and display method. Also, create an object to utilize the student interface.

Code:

```
interface Student {
   name:string,
   city:string,
   branch:string
   disp:()=>string
}

var stud : Student = {
   name : "Parmar Parth",
```

```
    city : "Rajkot",
    branch : "CE",
    disp:():string=>{
        return "Hello..."
    }
}

console.log(stud.name)
console.log(stud.city)
console.log(stud.branch)
console.log(stud.disp())
```

5 Demonstrate the usage of single level and multiple inheritance of interface.

Code:

```
interface Person {
    name:string
}

interface Cricketer extends Person{
    area:string
}

interface Jersey extends Cricketer{
    jerno:number
}

var field = <Jersey>{};
field.name = "Dhoni",
field.area = "Batsman"
field.jerno = 7

console.log("Name: "+ field.name)
console.log("Area: "+ field.area)
console.log("JerseyNo: "+ field.jerno)
```

OUTPUT :

```
PS D:\RKU-2-11-2020\RKU ALL SEM WORK 2020-21 FOLDER\SEM-6\UI\typescript_tutorials> tsc t6-5
PS D:\RKU-2-11-2020\RKU ALL SEM WORK 2020-21 FOLDER\SEM-6\UI\typescript_tutorials> node t6-5
Name: Dhoni
Area: Batsman
JerseyNo: 7
PS D:\RKU-2-11-2020\RKU ALL SEM WORK 2020-21 FOLDER\SEM-6\UI\typescript_tutorials>
```

6 Define a class Clock with three private integer data members hour, min and sec. Define a no
argument constructor to initialize time value to 12:00:00. Define a three argument constructor to initialize the time.
Define a methods to

a. Increment time to the next second.
b. Display the time value.
c. Return the hour (getHour():number)
d. Return the minute (getMinute():number)
e. Return the seconds (getSeconds():number)

Code:

```
class clock{
    private hour:number;
    private min:number;
    private sec:number;

    constructor(){
        this.hour = 12;
        this.min = 0;
        this.sec = 0;
    }

    incrementSecs(){
        return this.sec++;
    }

    getHour(){
        return this.hour;
    }

    getMinute(){
```

```
        return this.min;
    }

    getSecond(){
        return this.sec;
    }

    displayTime(){
        return this.hour + this.min + this.sec;
    }
}

var show = new clock();
show.incrementSecs();
show.getHour();
show.getMinute();
show.getSecond();
show.displayTime();

console.log(show)
```

OUTPUT :

```
PS D:\RKU-2-11-2020\RKU ALL SEM WORK 2020-21 FOLDER\SEM-6\UI\typescript_tutorials> tsc t6-6
PS D:\RKU-2-11-2020\RKU ALL SEM WORK 2020-21 FOLDER\SEM-6\UI\typescript_tutorials> node t6-6
clock { hour: 12, min: 0, sec: 1 }
PS D:\RKU-2-11-2020\RKU ALL SEM WORK 2020-21 FOLDER\SEM-6\UI\typescript_tutorials>
```

Tutorial 7

Create a component that displays data the from the live API created using node(refer prev tutorials).

Make sure to use following things:

ngFor
bootstrap (any)
Pagination
Sorting
Searching
Service
Interface

Code:-

```
import{ NgModule} from'@angular/core';
import{ BrowserModule} from'@angular/platform-browser';


import{ AppRoutingModule} from'./app-routing.module';
import{ AppComponent} from'./app.component';
import{ PostsComponent} from'./posts/posts.component';
import{ HttpClientModule} from'@angular/common/http';
import{NgxPaginationModule} from'ngx-pagination';
import{ OrderModule} from'ngx-order-pipe';


@NgModule({
  declarations:[
    AppComponent,
    PostsComponent
  ],
  imports:[
    BrowserModule,
    AppRoutingModule,
    HttpClientModule,
```

```
    NgxPaginationModule,
    OrderModule
  ],
  providers:[],
  bootstrap:[AppComponent]
})
exportclassAppModule{ }
```

```
import{ Injectable} from'@angular/core';
import{HttpClient} from'@angular/common/http';



@Injectable({
  providedIn:'root'
})
exportclassPostService{


  url= "https://jsonplaceholder.typicode.com/posts"


  constructor(private_http:HttpClient) { }


  getData(){
    returnthis._http.get(this.url);
  }
}
```

```
import{ Component, OnInit} from'@angular/core';
```

```
import{ PostService} from'../Service/post.service';


@Component({
  selector:'app-posts',
  templateUrl:'./posts.component.html',
  styleUrls:['./posts.component.css']
})
exportclassPostsComponentimplementsOnInit{


  data:any= []
  p: number= 1;


  key= 'id';
  reverse:boolean=false;


  constructor(private_postService:PostService) { }


  ngOnInit(): void{
    this._postService.getData().subscribe(res=>{
      this.data=res;
      console.log(this.data)
    })
  }


  sort(key:string){
    this.key= key;
    this.reverse=!this.reverse;
  }


}
```

```
<tableclass="table">
  <thead>
    <th(click)="sort('id')">Id</th>
    <th(click)="sort('title')">Title</th>
    <th(click)="sort('body')">Body</th>
  </thead>
  <tbody>
    <tr*ngFor="let item of data | paginate: { itemsPerPage: 10,
currentPage: p } | orderBy:key:reverse">
      <td>{{item.id}}</td>
      <td>{{item.title}}</td>
      <td>{{item.body}}</td>
    </tr>
  </tbody>
</table>
<pagination-controls(pageChange)="p = $event"></pagination-controls>
```

Output:-

Id	Title	Body
1	sunt aut facere repellat provident occaecati excepturi optio reprehenderit	quia et suscipit suscipit recusandae consequuntur expedita et cum reprehenderit molestiae ut ut quas totam nostrum rerum est autem sunt rem eveniet architecto
2	qui est esse	est rerum tempore vitae sequi sint nihil reprehenderit dolor beatae ea dolores neque fugiat blanditiis voluptate porro vel nihil molestiae ut reiciendis qui aperiam non debitis possimus qui neque nisi nulla
3	ea molestias quasi exercitationem repellat qui ipsa sit aut	et iusto sed quo iure voluptatem occaecati omnis eligendi aut ad voluptatem doloribus vel accusantium quis pariatur molestiae porro eius odio et labore et velit aut
4	eum et est occaecati	ullam et saepe reiciendis voluptatem adipisci sit amet autem assumenda provident rerum culpa quis hic commodi nesciunt rem tenetur doloremque ipsam iure quis sunt voluptatem rerum illo velit
5	nesciunt quas odio	repudiandae veniam quaerat sunt sed alias aut fugiat sit autem sed est voluptatem omnis possimus esse voluptatibus quis est aut tenetur dolor neque
6	dolorem eum magni eos aperiam quia	ut aspernatur corporis harum nihil quis provident sequi mollitia nobis aliquid molestiae perspiciatis et ea nemo ab reprehenderit accusantium quas voluptate dolores velit et doloremque molestiae
7	magnam facilis autem	dolore placeat quibusdam ea quo vitae magni quis enim qui quis quo nemo aut saepe quidem repellat excepturi ut quia sunt ut sequi eos ea sed quas
8	dolorem dolore est ipsam	dignissimos aperiam dolorem qui eum facilis quibusdam animi sint suscipit qui sint possimus cum quaerat magni maiores excepturi ipsam ut commodi dolor voluptatum modi aut vitae
9	nesciunt iure omnis dolorem tempora et accusantium	consectetur animi nesciunt iure dolore enim quia ad veniam autem ut quam aut nobis et est aut quod aut provident voluptas autem voluptas
10	optio molestias id quia eum	quo et expedita modi cum officia vel magni doloribus qui repudiandae vero nisi sit quos veniam quod sed accusamus veritatis error

Id	Title	Body
10	optio molestias id quia eum	quo et expedita modi cum officia vel magni doloribus qui repudiandae vero nisi sit quos veniam quod sed accusamus veritatis error
9	nesciunt iure omnis dolorem tempora et accusantium	consectetur animi nesciunt iure dolore enim quia ad veniam autem ut quam aut nobis et est aut quod aut provident voluptas autem voluptas
8	dolorem dolore est ipsam	dignissimos aperiam dolorem qui eum facilis quibusdam animi sint suscipit qui sint possimus cum quaerat magni maiores excepturi ipsam ut commodi dolor voluptatum modi aut vitae
7	magnam facilis autem	dolore placeat quibusdam ea quo vitae magni quis enim qui quis quo nemo aut saepe quidem repellat excepturi ut quia sunt ut sequi eos ea sed quas
6	dolorem eum magni eos aperiam quia	ut aspernatur corporis harum nihil quis provident sequi mollitia nobis aliquid molestiae perspiciatis et ea nemo ab reprehenderit accusantium quas voluptate dolores velit et doloremque molestiae
5	nesciunt quas odio	repudiandae veniam quaerat sunt sed alias aut fugiat sit autem sed est voluptatem omnis possimus esse voluptatibus quis est aut tenetur dolor neque
4	eum et est occaecati	ullam et saepe reiciendis voluptatem adipisci sit amet autem assumenda provident rerum culpa quis hic commodi nesciunt rem tenetur doloremque ipsam iure quis sunt voluptatem rerum illo velit
3	ea molestias quasi exercitationem repellat qui ipsa sit aut	et iusto sed quo iure voluptatem occaecati omnis eligendi aut ad voluptatem doloribus vel accusantium quis pariatur molestiae porro eius odio et labore et velit aut
2	qui est esse	est rerum tempore vitae sequi sint nihil reprehenderit dolor beatae ea dolores neque fugiat blanditiis voluptate porro vel nihil molestiae ut reiciendis qui aperiam non debitis possimus qui neque nisi nulla
1	sunt aut facere repellat provident occaecati excepturi optio reprehenderit	quia et suscipit suscipit recusandae consequuntur expedita et cum reprehenderit molestiae ut ut quas totam nostrum rerum est autem sunt rem eveniet architecto

Id	Title	Body
11	et ea vero quia laudantium autem	delectus reiciendis molestiae occaecati non minima eveniet qui voluptatibus accusamus in eum beatae sit vel qui neque voluptates ut commodi qui incidunt ut animi commodi
12	in quibusdam tempore odit est dolorem	itaque id aut magnam praesentium quia et ea odit et ea voluptas et sapiente quia nihil amet occaecati quia id voluptatem incidunt ea est distinctio odio
13	dolorum ut in voluptas mollitia et saepe quo animi	aut dicta possimus sint mollitia voluptas commodi quo doloremque iste corrupti reiciendis voluptatem eius rerum sit cumque quod eligendi laborum minima perferendis recusandae assumenda consectetur porro architecto ipsum ipsam
14	voluptatem eligendi optio	fuga et accusamus dolorum perferendis illo voluptas non doloremque neque facere ad qui dolorum molestiae beatae sed aut voluptas totam sit illum
15	eveniet quod temporibus	reprehenderit quos placeat velit minima officia dolores impedit repudiandae molestiae nam voluptas recusandae quis delectus officiis harum fugiat vitae
16	sint suscipit perspiciatis velit dolorum rerum ipsa laboriosam odio	suscipit nam nisi quo aperiam aut asperiores eos fugit maiores voluptatibus quia voluptatem quis ullam qui in alias quia est consequatur magni mollitia accusamus ea nisi voluptate dicta
17	fugit voluptas sed molestias voluptatem provident	eos voluptas et aut odit natus earum aspernatur fuga molestiae ullam deserunt ratione qui eos qui nihil ratione nemo velit ut aut id quo
18	voluptate et itaque vero tempora molestiae	eveniet quo quis laborum totam consequatur non dolor ut et est repudiandae est voluptatem vel debitis et magnam
19	adipisci placeat illum aut reiciendis qui	illum quis cupiditate provident sit magnam ea sed aut omnis veniam maiores ullam consequatur atque adipisci quo iste expedita sit quos voluptas
20	doloribus ad provident suscipit at	qui consequuntur ducimus possimus quisquam amet similique suscipit porro ipsam amet eos veritatis officiis exercitationem vel fugit aut necessitatibus totam omnis rerum consequatur expedita quidem cumque explicabo

« Previous 1 2 3 4 5 ... 10 Next »

Id	Title	Body
41	non est facere	molestias id nostrum excepturi molestiae dolore omnis repellendus quaerat saepe consectetur iste quaerat tenetur asperiores accusamus ex ut nam quidem est ducimus sunt debitis saepe
42	commodi ullam sint et excepturi error explicabo praesentium voluptas	odio fugit voluptatum ducimus earum autem est incidunt voluptatem odit reiciendis aliquam sunt sequi nulla dolorem non facere repellendus voluptates quia ratione harum vitae ut
43	eligendi iste nostrum consequuntur adipisci praesentium sit beatae perferendis	similique fugit est illum et dolorum harum et voluptate eaque quidem exercitationem quos nam commodi possimus cum odio nihil nulla dolorum exercitationem magnam ex et a et distinctio debitis
44	optio dolor molestias sit	temporibus est consectetur dolore et libero debitis vel velit laboriosam quia ipsum quibusdam qui itaque fuga rem aut ea et iure quam sed maxime ut distinctio quae
45	ut numquam possimus omnis eius suscipit laudantium iure	est natus reiciendis nihil possimus aut provident ex et dolor repellat pariatur est nobis rerum repellendus dolorem autem
46	aut quo modi neque nostrum ducimus	voluptatem quisquam iste voluptatibus natus officiis facilis dolorem quis quas ipsam vel et voluptatum in aliquid
47	quibusdam cumque rem aut deserunt	voluptatem assumenda ut qui ut cupiditate aut impedit veniam occaecati nemo illum voluptatem laudantium molestiae beatae rerum ea iure soluta nostrum eligendi et voluptate
48	ut voluptatem illum ea doloribus itaque eos	voluptates quo voluptatem facilis iure occaecati vel assumenda rerum officia et illum perspiciatis ab deleniti laudantium repellat ad ut et autem reprehenderit
49	laborum non sunt aut ut assumenda perspiciatis voluptas	inventore ab sint natus fugit id nulla sequi architecto nihil quaerat eos tenetur in in eum veritatis non quibusdam officiis aspernatur cumque aut commodi aut
50	repellendus qui recusandae incidunt voluptates tenetur qui omnis exercitationem	error suscipit maxime adipisci consequuntur recusandae voluptas eligendi et est et voluptates quia distinctio ab amet quaerat molestiae et vitae adipisci impedit sequi nesciunt quis consectetur

« Previous 1 ... 4 5 6 ... 10 Next »

Activity 4 Bootstrap Portfolio

```
<!DOCTYPE
html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-
scale=1.0">
  <link rel="stylesheet" href="css\bootstrap.min.css">
  <link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/4.7.0/css/font-awesome.min.css">

  <script src="js\bootstrap.min.js"></script>
  <title>Document</title>
</head>
<body>

 <div class="container">
  <div class="row">
  <div class="col-md-3 col-lg-3">
<img  src="image\p3.png" class="img-thumbnail"/>

  </div>
  <div class="col-md-9 col-lg-9">
<div class="d-flex flex-column  ">
 <div class="bg-dark d-flex text-white p-md-4 align-items-center
justify-content-between ">
  <div><h1 class="display-6">CRYPTO PUNK</h1></div>
  <a href="https://www.facebook.com/"><div><i class="fa fa-
facebook text-white"></i></div></a>
  <a href="https://www.instagram.com/"><div><i class="fa fa-
instagram text-white"></i></div></a>
  <a href="https://www.twitter.com/"><div><i class="fa fa-
twitter text-white"></i></div></a>
```

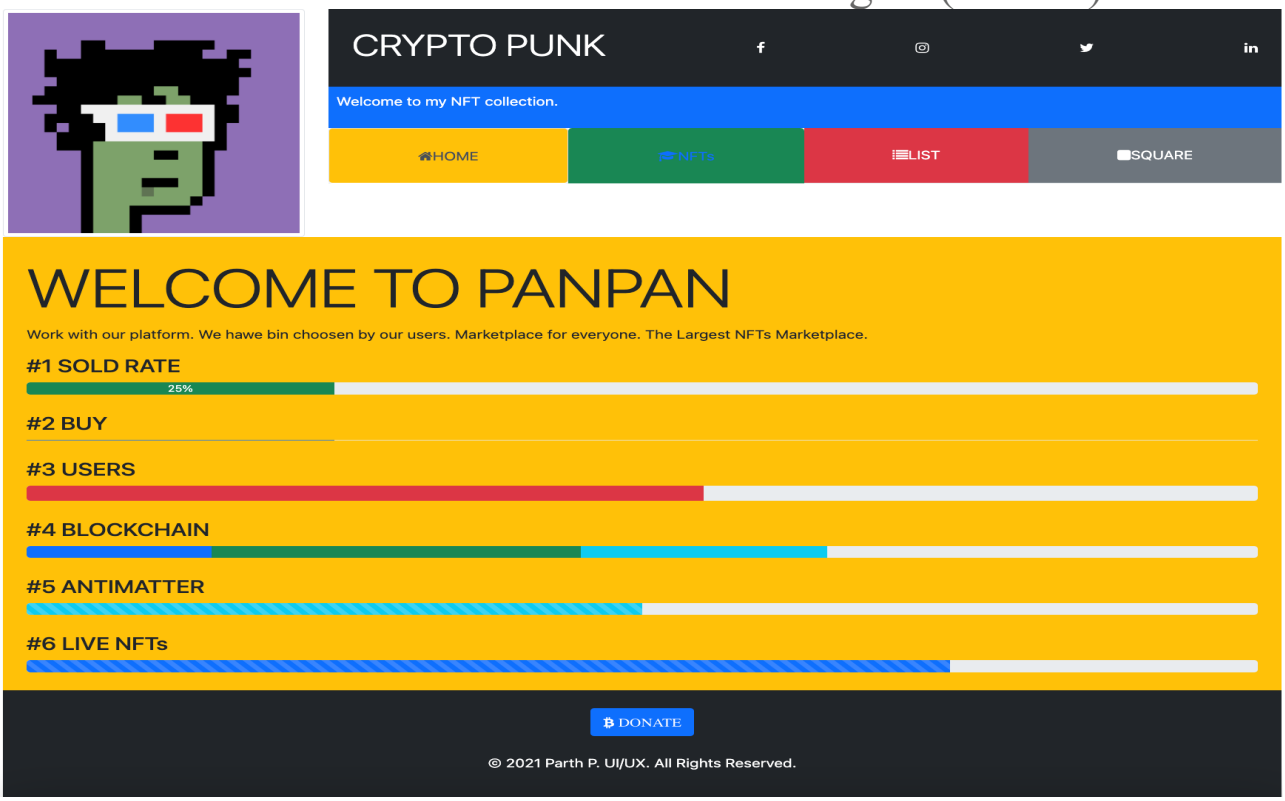
```
    <a href="https://www.linkedin.com/"><div><i class="fa fa-linkedin text-white"></i></div></a>

  </div>
  <div class="bg-primary text-white p-md-2  "><p>Welcome to my NFT collection.</p>

  </div>
  <div class="d-flex text-white text-center nav nav-tabs" id="nav-tab" role="tablist">
    <div class=" btn bg-warning p-md-4  flex-fill nav-link active" id="nav-home-tab" data-bs-toggle="tab" data-bs-target="#nav-home" type="button" role="tab" aria-controls="nav-home" aria-selected="true" ><i class="fa fa-home"></i>HOME</div>
    <div class="bg-success p-md-4  flex-fill nav-link" id="nav-profile-tab" data-bs-toggle="tab" data-bs-target="#nav-profile" type="button" role="tab" aria-controls="nav-profile" aria-selected="false"><i class="fa fa-graduation-cap"></i>NFTs</div>
    <div class="bg-danger p-md-4  flex-fill"><i class="fa fa-list"></i>LIST</div>
    <div class="bg-secondary p-md-4  flex-fill"><i class="fa fa-square"></i>SQUARE</div>
  </div>
</div>

    </div>
  </div>

  <div class="tab-content border-5" id="nav-tabContent">

    <div class="bg-warning p-4 container  tab-pane fade show active" id="nav-home" role="tabpanel" aria-labelledby="nav-home-tab" ><h4 class=" display-2">WELCOME TO PANPAN</h4>
```

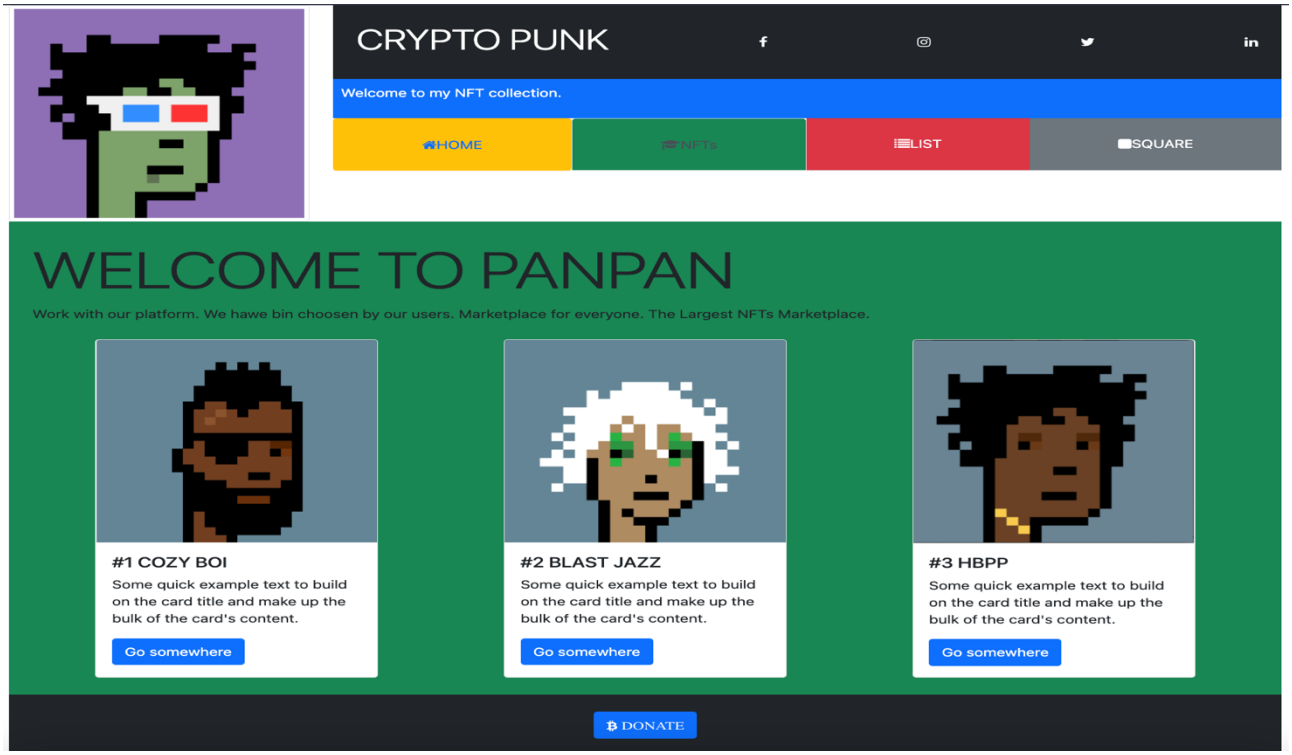
```
<p>Work with our platform. We hawe bin choosen by our users. Marketplace for everyone. The Largest NFTs Marketplace.</p>

<h4>#1 SOLD RATE</h4>
<div class="progress">

  <div class="progress-bar bg-success" role="progressbar" style="width: 25%;" aria-valuenow="25" aria-valuemin="0" aria-valuemax="100">25%</div>
</div><br>
<h4>#2 BUY </h4>
<div class="progress" style="height: 1px;">
  <div class="progress-bar" role="progressbar" style="width: 25%;" aria-valuenow="25" aria-valuemin="0" aria-valuemax="100"></div>
</div><br>
<h4>#3 USERS</h4>
<div class="progress " style="height: 20px;">
  <div class="progress-bar bg-danger" role="progressbar" style="width: 55%;" aria-valuenow="55" aria-valuemin="0" aria-valuemax="100"></div>
</div><br>
<h4>#4 BLOCKCHAIN</h4>
<div class="progress">
  <div class="progress-bar" role="progressbar" style="width: 15%" aria-valuenow="15" aria-valuemin="0" aria-valuemax="100"></div>
  <div class="progress-bar bg-success" role="progressbar" style="width: 30%" aria-valuenow="30" aria-valuemin="0" aria-valuemax="100"></div>
  <div class="progress-bar bg-info" role="progressbar" style="width: 20%" aria-valuenow="20" aria-valuemin="0" aria-valuemax="100"></div>
</div><br>
<h4>#5 ANTIMATTER</h4>
<div class="progress">
  <div class="progress-bar progress-bar-striped bg-info" role="progressbar" style="width: 50%" aria-valuenow="50" aria-valuemin="0" aria-valuemax="100"></div>
</div><br>
```

```
                    <h4>#6 LIVE NFTs</h4>
                    <div class="progress">
                      <div class="progress-bar progress-bar-striped progress-bar-animated" role="progressbar" aria-valuenow="75" aria-valuemin="0" aria-valuemax="100" style="width: 75%"></div>
                    </div>


                  </div>


                  <div class="bg-success p-4 container  tab-pane fade" id="nav-profile" role="tabpanel" aria-labelledby="nav-profile-tab" ><h4 class=" display-2">WELCOME TO PANPAN</h4>

                    <p>Work with our platform. We hawe bin choosen by our users. Marketplace for everyone. The Largest NFTs Marketplace.</p>

                    <div class="d-flex justify-content-around mt-4 "><div class="card mr-4" style="width: 18rem;">
                      <img src="image\p1.jpeg" class="card-img-top" alt="...">
                      <div class="card-body">
                        <h5 class="card-title">#1 COZY BOI</h5>
                        <p class="card-text">Some quick example text to build on the card title and make up the bulk of the card's content.</p>
                        <a href="#" class="btn btn-primary">Go somewhere</a>
                      </div>
                    </div><div class="card" style="width: 18rem;">
                      <img src="image\p2.png" class="card-img-top" alt="...">
                      <div class="card-body">
                        <h5 class="card-title">#2 BLAST JAZZ</h5>
                        <p class="card-text">Some quick example text to build on the card title and make up the bulk of the card's content.</p>
                        <a href="#" class="btn btn-primary">Go somewhere</a>
                      </div>
                    </div>
                      <div class="card" style="width: 18rem;">
```

```
                    <img src="image\p4.png" class="card-img-top" alt="...">
                    <div class="card-body">
                     <h5 class="card-title">#3 HBPP</h5>
                      <p class="card-text">Some quick example text to build on
                    the card title and make up the bulk of the card's content.</p>
                     <a href="#" class="btn btn-primary">Go somewhere</a>
                    </div>
                   </div>
                  </div>
                 </div>


                 </div>
                 <footer class=" bg-dark p-4 text-center  text-white"><button
                 class="btn bg-primary text-white"> <i class="fa fa-bitcoin
                 "> DONATE</i> </button>
                 <br><br>
                 <p>
                 © 2021 Parth P. UI/UX. All Rights Reserved.</p>
                 </footer>
                 </div>
                 </body>
                 </html>
```

OUTPUT:


CRYPTO PUNK
Welcome to my NFT collection.
HOME
NFTs
LIST
SQUARE
WELCOME TO PANPAN
Work with our platform. We hawe bin choosen by our users. Marketplace for everyone. The Largest NFTs Marketplace.
#1 SOLD RATE
25%
#2 BUY
#3 USERS
#4 BLOCKCHAIN
#5 ANTIMATTER
#6 LIVE NFTs
DONATE
© 2021 Parth P. UI/UX. All Rights Reserved.


CRYPTO PUNK
Welcome to my NFT collection.
HOME
NFTs
LIST
SQUARE
WELCOME TO PANPAN
Work with our platform. We hawe bin choosen by our users. Marketplace for everyone. The Largest NFTs Marketplace.
#1 COZY BOI
Some quick example text to build on the card title and make up the bulk of the card's content.
Go somewhere
#2 BLAST JAZZ
Some quick example text to build on the card title and make up the bulk of the card's content.
Go somewhere
#3 HBPP
Some quick example text to build on the card title and make up the bulk of the card's content.
Go somewhere
DONATE

Hands on for Querying Documents

1: Find all the result of the given collection.

2: Show the result in pretty format?

3: Get only MongoDB data as a output.

4: Get only MongoDB data as a output with only name field.

5: Get the MongoDB data without _ID field in it.

6: set the filter to “active:true” and get only the first field with “active:true” value.

7: Do the same as 5question but with different method.

8: Do the same as 5question but this time, get the 2nd field with active:true by skipping the 1st field.

1)

```
> db.student.find()
{ "_id" : ObjectId("61ba0f0ccbe57f17ac0ceb62"), "name" : "harsh", "branch" : "CE", "height" : "6'2", "status" : "A" }
{ "_id" : ObjectId("61ba0f1fcbe57f17ac0ceb63"), "name" : "harsh ved", "branch" : "CE", "height" : "5'2", "status" : "NA" }
{ "_id" : ObjectId("61ba0f37cbe57f17ac0ceb64"), "name" : "parth parmar", "branch" : "IT", "height" : "5'11", "status" : "NA" }
{ "_id" : ObjectId("61ba0f53cbe57f17ac0ceb65"), "name" : "dhairya upadhyay", "branch" : "IT", "height" : "4'11", "status" : "A" }
```

2)

```
> db.student.find().pretty()
{
        "_id" : ObjectId("61ba0f0ccbe57f17ac0ceb62"),
        "name" : "harsh",
        "branch" : "CE",
        "height" : "6'2",
        "status" : "A"
}
{
        "_id" : ObjectId("61ba0f1fcbe57f17ac0ceb63"),
        "name" : "harsh ved",
        "branch" : "CE",
        "height" : "5'2",
        "status" : "NA"
}
{
        "_id" : ObjectId("61ba0f37cbe57f17ac0ceb64"),
        "name" : "parth parmar",
        "branch" : "IT",
        "height" : "5'11",
        "status" : "NA"
}
{
        "_id" : ObjectId("61ba0f53cbe57f17ac0ceb65"),
        "name" : "dhairya upadhyay",
        "branch" : "IT",
        "height" : "4'11",
        "status" : "A"
}
```

3)

```
> db.student.find({name:"harsh"}).pretty()
{
        "_id" : ObjectId("61ba0f0ccbe57f17ac0ceb62"),
        "name" : "harsh",
        "branch" : "CE",
        "height" : "6'2",
        "status" : "A"
}
```

4)

```
> db.student.find({}, {name:1, _id:0}).pretty()
{ "name" : "harsh" }
{ "name" : "harsh ved" }
{ "name" : "parth parmar" }
{ "name" : "dhairya upadhyay" }
>
```

5)

```
> db.student.find({status:"A"}).limit(1).pretty()
{
        "_id" : ObjectId("61ba0f0ccbe57f17ac0ceb62"),
        "name" : "harsh",
        "branch" : "CE",
        "height" : "6'2",
        "status" : "A"
```

6)

```
> db.student.find({status:"A"}).pretty()
{
        "_id" : ObjectId("61ba0f0ccbe57f17ac0ceb62"),
        "name" : "harsh",
        "branch" : "CE",
        "height" : "6'2",
        "status" : "A"
}
{
        "_id" : ObjectId("61ba0f53cbe57f17ac0ceb65"),
        "name" : "dhairya upadhyay",
        "branch" : "IT",
        "height" : "4'11",
        "status" : "A"
}
```

7)

```
> db.student.find({status:"A"}).limit(1).skip(1).pretty()
{
        "_id" : ObjectId("61ba0f53cbe57f17ac0ceb65"),
        "name" : "dhairya upadhyay",
        "branch" : "IT",
        "height" : "4'11",
        "status" : "A"
}
```

Hands on with Aggregation & Grouping

1. List the category of books

```
> db.book.aggregate([{$group:{_id:"$Category"}}])
{ "_id" : "programming" }
{ "_id" : "fiction" }
{ "_id" : "history" }
{ "_id" : "novel" }
{ "_id" : "gk" }
```

2. find the total count of documents for each category

```
> db.book.aggregate([{$group:{_id:"$Category", "Total": {$sum:1}}}
{ "_id" : "gk", "Total" : 1 }
{ "_id" : "programming", "Total" : 2 }
{ "_id" : "fiction", "Total" : 1 }
{ "_id" : "history", "Total" : 1 }
{ "_id" : "novel", "Total" : 1 }
```

3. find the documents having copies greater than 30 according to the categories

```
> db.book.aggregate([{$match:{qty:{$gt:30}}},{$group:{_id:"$Category", "Total": {$sum:1},"minimum":{$min:"$qty"}}}])
{ "_id" : "history", "Total" : 1, "minimum" : 35 }
{ "_id" : "novel", "Total" : 1, "minimum" : 35 }
{ "_id" : "programming", "Total" : 1, "minimum" : 33 }
>
```

4. find the documents by grouping according to isbn number and sort by the same

```
> db.book.aggregate([{$group:{_id:"$ISBN"}},{$sort:{_id:1}}])
{ "_id" : 1234 }
{ "_id" : 12354 }
>
```

5. find the documents by grouping to categories and sorting it by category name only for those documents having available status true

```
> db.book.aggregate([{$match:{status:"A"}},{$group:{_id:"$Category"}},{$sort:{_id:1}}])
{ "_id" : "fiction" }
{ "_id" : "novel" }
{ "_id" : "programming" }
>
```

6. find the average no. of books based on the categories.

```
> db.book.aggregate([{$group:{_id:"$Category","average":{$avg:"$qty"}}}])
{ "_id" : "history", "average" : 35 }
{ "_id" : "gk", "average" : 30 }
{ "_id" : "fiction", "average" : 22 }
{ "_id" : "programming", "average" : 28 }
{ "_id" : "novel", "average" : 35 }
```

7. find the min and max no. of books based on the categories.

```
> db.book.aggregate([{$group:{_id:"$Category","minimum":{$min:"$qty"}}}])
{ "_id" : "history", "minimum" : 35 }
{ "_id" : "gk", "minimum" : 30 }
{ "_id" : "fiction", "minimum" : 22 }
{ "_id" : "programming", "minimum" : 23 }
{ "_id" : "novel", "minimum" : 22 }
>
```

8. find the total count of documents for each category

```
> db.book.aggregate([{$group:{_id:"$Category","Total count":{$sum:1}}}])
{ "_id" : "history", "Total count" : 4 }
{ "_id" : "gk", "Total count" : 4 }
{ "_id" : "fiction", "Total count" : 4 }
{ "_id" : "programming", "Total count" : 8 }
{ "_id" : "novel", "Total count" : 5 }
>
```

9. find the total count of not available books for each category.

```
> db.book.aggregate([{$match:{status:"NA"}},{$group:{_id:"$Category","Total count":{$sum:1}}}])
{ "_id" : "history", "Total count" : 2 }
{ "_id" : "gk", "Total count" : 2 }
{ "_id" : "programming", "Total count" : 2 }
>
```

10. find the total no. of copies of books for each category.

```
> db.book.aggregate([{$group:{_id:"$Category","total copies":{$sum:"$qty"}}}])
{ "_id" : "history", "total copies" : 105 }
{ "_id" : "gk", "total copies" : 90 }
{ "_id" : "fiction", "total copies" : 66 }
{ "_id" : "programming", "total copies" : 168 }
{ "_id" : "novel", "total copies" : 127 }
>
```

11. Sort the above output in descending order of copies.

```
> db.book.aggregate([{$group:{_id:"$Category","total copies":{$sum:"$qty"}}},{$sort:{_id:-1}}])
{ "_id" : "programming", "total copies" : 168 }
{ "_id" : "novel", "total copies" : 127 }
{ "_id" : "history", "total copies" : 105 }
{ "_id" : "gk", "total copies" : 90 }
{ "_id" : "fiction", "total copies" : 66 }
>
```

12. find the documents by grouping category and isbn number along with the count of each group.

```
> db.book.aggregate([{$group:{_id:{"Category":"$Category","ISBN":"$ISBN"},"total":{$sum:1}}}])
{ "_id" : { "Category" : "history", "ISBN" : 1234 }, "total" : 4 }
{ "_id" : { "Category" : "fiction", "ISBN" : 1234 }, "total" : 4 }
{ "_id" : { "Category" : "programming", "ISBN" : 1234 }, "total" : 4 }
{ "_id" : { "Category" : "programming", "ISBN" : 12354 }, "total" : 4 }
{ "_id" : { "Category" : "novel", "ISBN" : 1234 }, "total" : 5 }
{ "_id" : { "Category" : "gk", "ISBN" : 1234 }, "total" : 4 }
>
```

13. find the documents by grouping category and isbn number along with the count of each group. But the output should be displayed as follows:

Expected output:

{ "Category" : "history", "ISBN" : 32145, "total_docs" : 1 }
{ "Category" : "history", "ISBN" : 1234, "total_docs" : 2 }
{ "Category" : "fiction", "ISBN" : 1234, "total_docs" : 1 }
{ "Category" : "programming", "ISBN" : 1234, "total_docs" : 1 }
{ "Category" : "programming", "ISBN" : 12354, "total_docs" : 1 }
{ "Category" : "gk", "ISBN" : 1234, "total_docs" : 1 }
{ "Category" : "novel", "ISBN" : 1234, "total_docs" : 1 }

```
> db.book.aggregate([{$group:{_id :{"Category":"$Category","ISBN":"$ISBN","Total docs" : {$sum :1}}},{$project:{"Category" : "$_id.Category" ,"ISBN" : "$_id.ISBN", "Total_
docs" : "$_id.Total docs",_id : 0}}])
{ "Category" : "programming", "ISBN" : 1234, "Total_docs" : 1 }
{ "Category" : "fiction", "ISBN" : 1234, "Total_docs" : 1 }
{ "Category" : "programming", "ISBN" : 12354, "Total_docs" : 1 }
{ "Category" : "history", "ISBN" : 1234, "Total_docs" : 1 }
{ "Category" : "gk", "ISBN" : 1234, "Total_docs" : 1 }
{ "Category" : "novel", "ISBN" : 1234, "Total_docs" : 1 }
>
```

14. find the documents by grouping category and isbn number along with the total no of copies for each group. Display the group having highest copies.

Expected output:

{ "Category" : "history", "ISBN" : 32145, "total_copies" : 50 }

```
> db.book.aggregate([{$group:{_id :{"Category":"$Category","ISBN":"$ISBN","total_copies" : {$sum :"$qty"}}},{$project:{"Category" : "$_id.Category" ,"ISBN" : "$_id.ISBN","
total_copies" : "$_id.total_copies",_id : 0}},{$sort : {"total_copies" : -1}},{$limit : 1}])
{ "Category" : "history", "ISBN" : 1234, "total_copies" : 35 }
>
```

Hands-On for Querying with filters

1. Create a Database named "Stocks"

```
> use stocks
switched to db stocks
```

2. Create an empty collection named "inventory"

```
> db.createCollection("inventory")
{ "ok" : 1 }
```

3. Insert below records in the inventory collection all together.

```
db.inventory.insertMany([
   { item: "journal", qty: 25, size: { h: 14, w: 21, uom: "cm" }, status: "A" },
   { item: "notebook", qty: 50, size: { h: 8.5, w: 11, uom: "in" }, status: "A" },
   { item: "paper", qty: 100, size: { h: 8.5, w: 11, uom: "in" }, status: "D" },
   { item: "planner", qty: 75, size: { h: 22.85, w: 30, uom: "cm" }, status: "D" },
   { item: "postcard", qty: 45, size: { h: 10, w: 15.25, uom: "cm" }, status: "A" }
]);
```

```
> db.inventory.insertMany([
... { item: "journal", qty: 25, size: { h: 14, w: 21, uom: "cm"}, status: "A"},
... { item: "notebook", qty: 50, size: { h: 8.5, w: 11, uom: "in"}, status: "A"},
... { item: "paper", qty: 100, size: { h: 8.5, w: 11, uom: "in"}, status: "D"},
... { item: "planner", qty: 75, size: { h: 22.85, w: 30, uom: "cm"}, status: "D"},
... { item: "postcard", qty: 45, size: { h: 10, w: 15.25, uom: "cm"}, status: "A"}]
)
{
        "acknowledged" : true,
        "insertedIds" : [
                ObjectId("61ba26f1cbe57f17ac0ceb66"),
                ObjectId("61ba26f1cbe57f17ac0ceb67"),
                ObjectId("61ba26f1cbe57f17ac0ceb68"),
                ObjectId("61ba26f1cbe57f17ac0ceb69"),
                ObjectId("61ba26f1cbe57f17ac0ceb6a")
```

4. Select all documents from inventory collection

```
> db.inventory.find({})
{ "_id" : ObjectId("61ba26f1cbe57f17ac0ceb66"), "item" : "journal", "qty" : 25, "size" : { "h" : 14, "w" : 21, "uom" : "cm" }, "status" : "A" }
{ "_id" : ObjectId("61ba26f1cbe57f17ac0ceb67"), "item" : "notebook", "qty" : 50, "size" : { "h" : 8.5, "w" : 11, "uom" : "in" }, "status" : "A" }
{ "_id" : ObjectId("61ba26f1cbe57f17ac0ceb68"), "item" : "paper", "qty" : 100, "size" : { "h" : 8.5, "w" : 11, "uom" : "in" }, "status" : "D" }
{ "_id" : ObjectId("61ba26f1cbe57f17ac0ceb69"), "item" : "planner", "qty" : 75, "size" : { "h" : 22.85, "w" : 30, "uom" : "cm" }, "status" : "D" }
{ "_id" : ObjectId("61ba26f1cbe57f17ac0ceb6a"), "item" : "postcard", "qty" : 45, "size" : { "h" : 10, "w" : 15.25, "uom" : "cm" }, "status" : "A" }
```

5. selects from the inventory collection all documents where the status equals "D"

```
> db.inventory.find({status: "D"}).pretty()
{
        "_id" : ObjectId("61ba26f1cbe57f17ac0ceb68"),
        "item" : "paper",
        "qty" : 100,
        "size" : {
                "h" : 8.5,
                "w" : 11,
                "uom" : "in"
        },
        "status" : "D"
}
{
        "_id" : ObjectId("61ba26f1cbe57f17ac0ceb69"),
        "item" : "planner",
        "qty" : 75,
        "size" : {
                "h" : 22.85,
                "w" : 30,
                "uom" : "cm"
        },
        "status" : "D"
}
```

6. Retrieves all documents from the inventory collection where status equals either "A" or "D".

```
> db.inventory.find({status:{$in:["A","D"]}}).pretty()
{
        "_id" : ObjectId("61ba26f1cbe57f17ac0ceb66"),
        "item" : "journal",
        "qty" : 25,
        "size" : {
                "h" : 14,
                "w" : 21,
                "uom" : "cm"
        },
        "status" : "A"
}
{
        "_id" : ObjectId("61ba26f1cbe57f17ac0ceb67"),
        "item" : "notebook",
        "qty" : 50,
        "size" : {
                "h" : 8.5,
                "w" : 11,
                "uom" : "in"
        },
        "status" : "A"
}
```

```
{
        "_id" : ObjectId("61ba26f1cbe57f17ac0ceb68"),
        "item" : "paper",
        "qty" : 100,
        "size" : {
                "h" : 8.5,
                "w" : 11,
                "uom" : "in"
        },
        "status" : "D"
}
{
        "_id" : ObjectId("61ba26f1cbe57f17ac0ceb69"),
        "item" : "planner",
        "qty" : 75,
        "size" : {
                "h" : 22.85,
                "w" : 30,
                "uom" : "cm"
        },
        "status" : "D"
}
{
        "_id" : ObjectId("61ba26f1cbe57f17ac0ceb6a"),
        "item" : "postcard",
        "qty" : 45,
        "size" : {
                "h" : 10,
                "w" : 15.25,
                "uom" : "cm"
        },
        "status" : "A"
}
```


7. Retrieve documents in the inventory collection where the status equals "A" **and** qty is less than ($lt (Links to an external site.)Links to an external site.) 30

```
> db.inventory.find({status:"A",qty:{$lt:30}}).pretty()
{
        "_id" : ObjectId("61bbfc8b72c4f1b3675588d1"),
        "item" : "journal",
        "qty" : 25,
        "size" : {
                "h" : 14,
                "w" : 21,
                "uom" : "cm"
        },
        "status" : "A"
}
```

8. Retrieve all documents in the collection where the status equals "A" **or** qty is less than ($lt (Links to an external site.)Links to an external site.) 30

```
> db.inventory.find({$or:[{status:"A",qty:{$lt:30}}]}).pretty()
...
...
>  db.inventory.find({$or:[{status:"A"},{qty:{$lt:30}}]}).pretty()
{
        "_id" : ObjectId("61bbfc8b72c4f1b3675588d1"),
        "item" : "journal",
        "qty" : 25,
        "size" : {
                "h" : 14,
                "w" : 21,
                "uom" : "cm"
        },
        "status" : "A"
}
{
        "_id" : ObjectId("61bbfc8b72c4f1b3675588d2"),
        "item" : "notebook",
        "qty" : 50,
        "size" : {
                "h" : 8.5,
                "w" : 11,
                "uom" : "in"
        },
        "status" : "A"
}
{
        "_id" : ObjectId("61bbfc8b72c4f1b3675588d5"),
        "item" : "postcard",
        "qty" : 45,
        "size" : {
                "h" : 10,
                "w" : 15.25,
                "uom" : "cm"
        },
        "status" : "A"
}
```

9. Update qty to 25 where item is notebook

```
> db.inventory.updateOne({ item: "notebook"}, {$set: {qty: 25}})
{ "acknowledged" : true, "matchedCount" : 1, "modifiedCount" : 1 }
```

10. Update all status to 'B' where status is 'D'

```
> db.inventory.updateMany({status: "D"}, {$set: {status: "B"}})
{ "acknowledged" : true, "matchedCount" : 2, "modifiedCount" : 2 }
```

11. Write a query to demonstrate usage of upsert.

```
> db.inventory.updateOne({item: "postcard"}, {$set: {qty: 46}}, {upsert: true})
{ "acknowledged" : true, "matchedCount" : 1, "modifiedCount" : 1 }
```

12. Delete the document where qty is 100

```
> db.inventory.remove({ qty: 100})
WriteResult({ "nRemoved" : 1 })
```

13. Delete all the documents from inventory collection.

```
> db.inventory.remove({})
WriteResult({ "nRemoved" : 4 })
```

14. Rename the collection.

```
> db.inventory.renameCollection("renamed")
{ "ok" : 1 }
```

15. Delete the database.

```
> db.dropDatabase()
{ "ok" : 1 }
```

Regex

1. find the documents whose description starts with "M".

```
> db.products.find({description: {$regex: /^M/}})
{ "_id" : 102, "sku" : "xyz456", "description" : "Many spaces before   line" }
{ "_id" : 103, "sku" : "XYZ789", "description" : "Multiple\nline description" }
>
```

2. find the documents whose description end with "n".

```
> db.products.find({description: {$regex: /n$/}})
{ "_id" : 103, "sku" : "XYZ789", "description" : "Multiple\nline description" }
>
```

3. find the documents whose description contains "line" word".

```
> db.products.find({description: {$regex: /line/}})
{ "_id" : 100, "sku" : "abc123", "description" : "Single line description." }
{ "_id" : 101, "sku" : "abc789", "description" : "First line\nSecond line" }
{ "_id" : 102, "sku" : "xyz456", "description" : "Many spaces before   line" }
{ "_id" : 103, "sku" : "XYZ789", "description" : "Multiple\nline description" }
>
```

4. find the documents whose description contains second character "i".

```
> db.products.find({description: {$regex: /^.i/}})
{ "_id" : 100, "sku" : "abc123", "description" : "Single line description." }
{ "_id" : 101, "sku" : "abc789", "description" : "First line\nSecond line" }
>
```

5. find the documents where "sku" fields contains "xyz", ignore the case sensitivity.

```
> db.products.find({sku: {$regex: /xyz/, $options:"i"}})
{ "_id" : 102, "sku" : "xyz456", "description" : "Many spaces before   line" }
{ "_id" : 103, "sku" : "XYZ789", "description" : "Multiple\nline description" }
>
```

6. find the documents where any line from the description starts with 'S..

```
> db.products.find({description:{$regex: /^S/,$options:"M"}})
{ "_id" : 100, "sku" : "abc123", "description" : "Single line description." }
```

Practice Basic CRUD

7. Create a database named "Students".

OUTPUT :

```
> use students
switched to db students
>
```

8. Create an empty collection named "studentData"

OUTPUT :

```
> db.createCollection("studentData")
{ "ok" : 1 }
```

9. Insert only one record with appropriate fields.

OUTPUT :

```
> db.studentData.insert({name:"Lucifer"})
WriteResult({ "nInserted" : 1 })
```

10. Try to insert 5 records together using single query

OUTPUT :

```
> db.studentData.insertMany([{lastname:"versace"},{age:"21"},{number:"9900999900"},{bg:"o+"},{height:"6'2"}
{
        "acknowledged" : true,
        "insertedIds" : [
                ObjectId("61b2c4e452d3a64e76c42c60"),
                ObjectId("61b2c4e452d3a64e76c42c61"),
                ObjectId("61b2c4e452d3a64e76c42c62"),
                ObjectId("61b2c4e452d3a64e76c42c63"),
                ObjectId("61b2c4e452d3a64e76c42c64")
        ]
}
```

11. Fetch all the documents in formatted manner.

OUTPUT :

```
> db.studentData.find().pretty()
{ "_id" : ObjectId("61b2c26e52d3a64e76c42c5f"), "name" : "Lucifer" }
{ "_id" : ObjectId("61b2c4e452d3a64e76c42c60"), "lastname" : "versace" }
{ "_id" : ObjectId("61b2c4e452d3a64e76c42c61"), "age" : "21" }
{ "_id" : ObjectId("61b2c4e452d3a64e76c42c62"), "number" : "9900999900" }
{ "_id" : ObjectId("61b2c4e452d3a64e76c42c63"), "bg" : "o+" }
{ "_id" : ObjectId("61b2c4e452d3a64e76c42c64"), "height" : "6'2" }
```

12. Fetch the document based on the filter criteria.

OUTPUT :

```
> db.studentData.find({name:"Lucifer"})
{ "_id" : ObjectId("61b2c26e52d3a64e76c42c5f"), "name" : "Lucifer" }
>
```

13. Delete one document from the "studentData".

OUTPUT :

```
> db.studentData.deleteOne({bg:"o+"})
{ "acknowledged" : true, "deletedCount" : 1 }
>
```

14. Delete the documents based on the filter criteria.

OUTPUT :

```
> db.studentData.deleteOne({bg:"o+"})
{ "acknowledged" : true, "deletedCount" : 1 }
>
```

15. Delete all the documents from the "studentData: collection

OUTPUT :

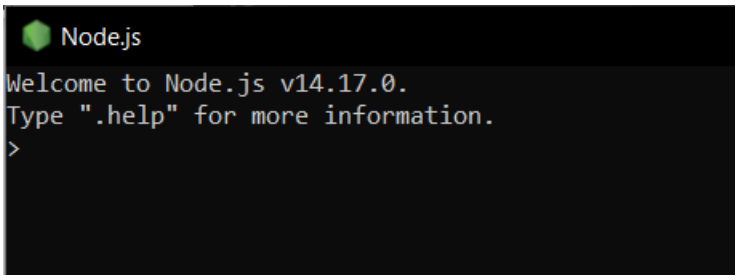
```
> db.studentData.deleteMany({name:"Lucifer"},{name:"harsh"})
{ "acknowledged" : true, "deletedCount" : 1 }
>
```

16. Drop the collection.

OUTPUT :

```
> db.studentData.drop()
true
>
```

Hands on with Node.js

1. Install Nodejs to set up the project environment.
2. 

2. Create a function expression in node js to count the number of items in an array.

```
var arr = [8,56,9,5,8,7,2,3,6,9,4,4];
var count = 0;
arr.forEach(element => {
    count++;
});

console.log("Count is  : ", count);
```

```
PROBLEMS    OUTPUT    TERMINAL    DEBUG CONSOLE
C:\Program Files\nodejs\node.exe .\Array.js
Count is  :  12
```

3. Create a module that contains three functions and export the same using nodejs.

```
var name = "Harsh Bhatt";

var sum = (a,b)=>{
   return a+b;
}

var div = (c,d)=>{
   return c/d;
}

module.exports={name, sum, div};
```

4. Create a module to import the above created function and utilize it.

```
var  test = require('./index')

console.log(test.sum(10,20))
console.log(test.name)
console.log(test.div(30,6))
```

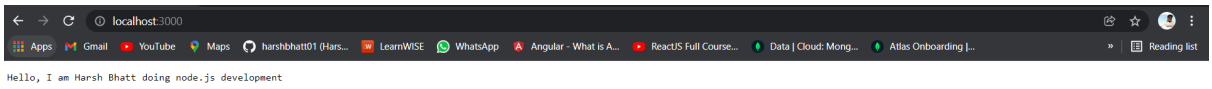
Output:

```
C:\Program Files\nodejs\node.exe .\second.js
30
Harsh Bhatt
5
```

5. Create a server using http module and listen to port 3000.

```
var http = require('http')

var server = http.createServer(function(req,res){

        res.writeHead(200,{"Content-Type":"text.html"});
        res.write("Hello, I am Harsh Bhatt doing node.js development");
        res.end();

}).listen(3000);

console.log("server is running at port 3000");
```

Output:



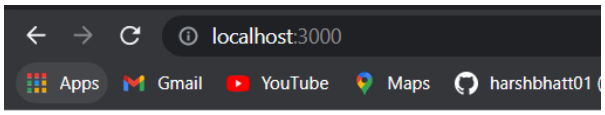
6. Display three different users(Student, admin and default) html pages as per the requested url.

```
var http = require('http');

var server = http.createServer(function(req,res){

// code to run the default page
   if(req.url== '/'){
   res.writeHead(200,{"Content-Type": "text.html"});
   res.write("harsh created a home(default) page");
   res.end();
   }

// code to run student page
   else if(req.url=='/student'){
   res.writeHead(200,{"Content-Type": "text.html"});
   res.write("harsh created a student page");
```

```
    res.end();

    }
// code to run admin page
    else if(req.url=='/admin'){
        res.writeHead(200,{"Content-Type": "text.html"});
        res.write("harsh created a admin page");
        res.end();

    }
// else than above three it will give the command as 'invalid'
    else
        res.end('invalid')

}).listen(3000);

console.log("server running on port 3000");
```

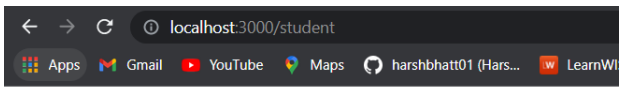
Output:

Default:



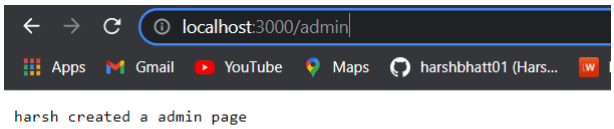
harsh created a home(default) page

Student:



harsh created a student page

Admin:



7. Create and import the class type object to display the full name of the user.

```
var name = {
      fname: "Hi",
      lname: " I am Harsh"
};
module.exports = {name};
```

```
var imp = require("./index.js");

console.log(imp.name.fname +
    imp.name.lname);
```

Output:

```
PS D:\6th sem\UI\node,js\practice jsons> node second
Hi I am Harsh
```

Practice with NavBar and Components

Create an angular project with three components namely - Home, Student and Result. Also, Add the navigation bar to the project such that it is accessible from all the components. Provide appropriate routing and static content to each component.

Code:

Index.html

```
<!doctype html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <title>Demo</title>
  <base href="/">
  <meta name="viewport"content="width=device-width, initial-scale=1">
  <link rel="icon"type="image/x-icon"href="favicon.ico">
  <link
rel="stylesheet"href="https://cdn.jsdelivr.net/npm/bootstrap@4.6.1/dist/css/bootstrap.
min.css">
  <script
src="https://cdn.jsdelivr.net/npm/jquery@3.5.1/dist/jquery.slim.min.js"></script>
  <script
src="https://cdn.jsdelivr.net/npm/popper.js@1.16.1/dist/umd/popper.min.js"></script
>
  <script
src="https://cdn.jsdelivr.net/npm/bootstrap@4.6.1/dist/js/bootstrap.bundle.min.js"></
script>
</head>
<body>
  <app-root></app-root>
</body>
</html>
```

App-routing.module.ts

```
import{ NgModule} from'@angular/core';
import{ RouterModule, Routes} from'@angular/router';
import{ HomeComponent} from'./home/home.component';
import{ ResultComponent} from'./result/result.component';
import{ StudentComponent} from'./student/student.component';


constroutes: Routes= [
  {path:'home',component:HomeComponent},
```

```
  {path:'student',component:StudentComponent},
  {path:'result',component:ResultComponent}
];


@NgModule({
  imports:[RouterModule.forRoot(routes)],
  exports:[RouterModule]
})
exportclassAppRoutingModule{ }
```

Navbar.html

```
<nav class="navbar navbar-expand-md bg-dark navbar-dark">
   <a class="navbar-brand"href="#">Navbar</a>
   <button class="navbar-toggler"type="button"data-
toggle="collapse"data-target="#collapsibleNavbar">
     <span class="navbar-toggler-icon"></span>
   </button>
   <div class="collapse navbar-collapse"id="collapsibleNavbar">
    <ul class="navbar-nav">
      <li class="nav-item">
       <a class="nav-link"routerLink="home"href="#">Home</a>
      </li>
      <li class="nav-item">
       <a class="nav-link"routerLink="student"href="#">Student</a>
      </li>
      <li class="nav-item">
       <a class="nav-link" routerLink="result"href="#">Result</a>
      </li>
    </ul>
   </div>
  </nav>
```

Home.html

```
<h1 class="text-center m-5">Welcome to my Homepage</h1>
```

Student.html

```
<div class="container">
   <h2 class="text-center m-4">Student details</h2>
   <div class="card-columns">
    <div *ngFor="let stud of students; let i = index"class="card bg-info">
      <div [routerLink]="['/studentdetails',{id:i}]"class="card-header text-
uppercase text-center">
         <h4>{{stud.erno}}</h4>
      </div>
      <div class="card-body text-center">
       <img [src]="stud.profile"class="rounded-circle">
       <h5 class="card-text">Name: {{stud.name}}</h5>
       <h5 class="card-text">DOB: {{stud.DOB}}</h5>
       <h5 class="card-text">Department: {{stud.department}}</h5>
      </div>
    </div>
   </div>
 </div>
```

Student.ts

```
import { Component, OnInit } from '@angular/core';

@Component({
  selector: 'app-student',
  templateUrl: './student.component.html',
  styleUrls: ['./student.component.css']
})
export class StudentComponent implements OnInit {

  students:any[]=[

{id:1,erno:"20SOECE15056",name:"ParthParmar",DOB:"04/09/2001",department:
"Computer",profile:"../assets/images/2.jpg"},
    {id:2,erno:"20SOECE15044",name:"Shyam
Patel",DOB:"23/10/2000",department:"Computer",profile:"../assets/images/2.jpg"},
    {id:3,erno:"20SOECE19058",name:"Shiv
Prajapati",DOB:"04/11/2001",department:"Computer",profile:"../assets/images/2.jp
g"},
```

```
    {id:4,erno:"20SOECE10012",name:"Devansh
Prajapati",DOB:"17/07/2002",department:"Computer",profile:"../assets/images/2.jp
g"},
    {id:5,erno:"20SOECE15039",name:"Raj
Patel",DOB:"26/01/2000",department:"Computer",profile:"../assets/images/2.jpg"},
    {id:6,erno:"20SOECE11035",name:"Dev
Patel",DOB:"25/11/2001",department:"Computer",profile:"../assets/images/2.jpg"},

  ]

  constructor() { }

  ngOnInit(): void {
  }

}
```

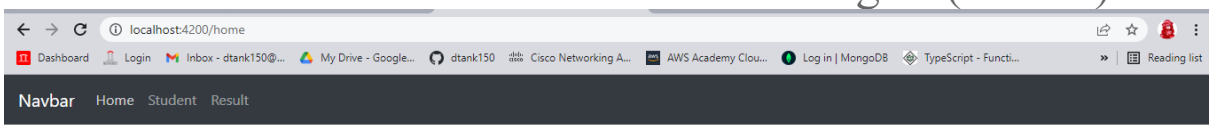
Student.css

```
img{
    width:200px;
    height:200px;
}
```

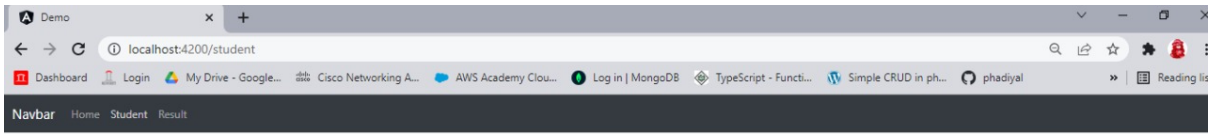
Result.html

```
<h2 class="text-center m-5">Result Display</h2>
```

Output:-

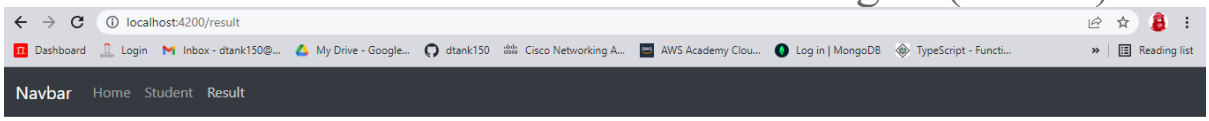


Welcome to my Homepage



Student details

20SOECE15056	20SOECE19058	20SOECE15039
Name: Yash Parajapati DOB: 04/09/2001 Department: Computer	Name: Shiv Prajapati DOB: 04/11/2001 Department: Computer	Name: Raj Patel DOB: 26/01/2000 Department: Computer



Result Display

Practice with events and string interpolation

1. Create a variable, array, function, object and an array of objects in the class file and utilize the same in html to display the data.

2. Demonstrate the usage of following events with appropriate example in the existing project:

- **Click**
- **MouseOver**
- **MouseLeave**
- **Keyup**
- **Keydown**
- **Keyup + enter**

Also, Provide the text box to enter username and display it in alert box when user Clicks on "Show" button.

Code:

Result.component.ts

```
import { Component, OnInit } from '@angular/core';

@Component({
  selector: 'app-result',
  templateUrl: './result.component.html',
  styleUrls: ['./result.component.css']
})
export class ResultComponent implements OnInit {

  names=['Yash','Shyam','Shiv','Devansh'];
  subjects=['Compter']
  marks=['99']


  constructor() { }

  ngOnInit(): void {
  }

  show(name:any){
    console.log(name.value)
  }
  func(subject:any){
    console.log(subject.value)
  }
  display(mark:any){
    console.log(mark.value)
  }
  submit(name:any,mark:any,subject:any){
    console.log(name.value)
    console.log(subject.value)
    console.log(mark.value)
  }
  over(){
    console.log("Mouseover called");
  }
```

```
  leave(){
    console.log("Mouseleave called");
  }

}
```

Result.component.html

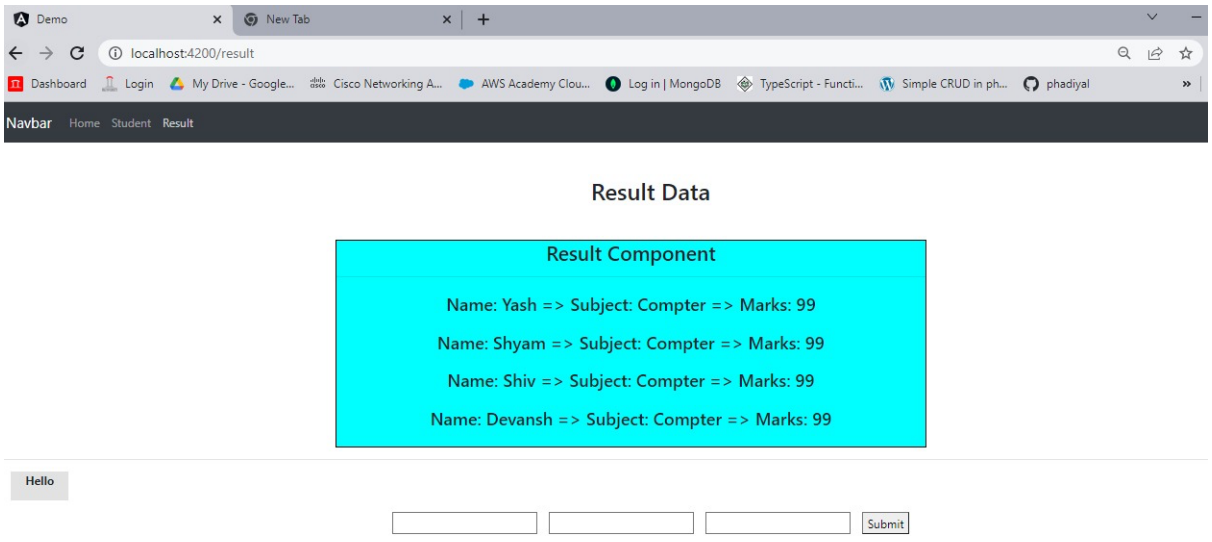
```
<h2 class="text-center m-5">Result Data</h2>
<div class="center">
   <h3 class="text-center">Result Component</h3>
   <hr>
   <h4 *ngFor="let name of names"class="text-center m-4">
    <h4 *ngFor="let subject of subjects">
   <h4 *ngFor="let mark of marks">
      Name: {{name}}  =>  Subject: {{subject}}  =>  Marks: {{mark}}
   </h4>
   </h4>
   </h4>

</div>
<hr>
<div class="col-md-12 text-center">
   <div(mouseover)='over()'(mouseleave)='leave()'style="height:40px;
width:80px; background:#e2e2e2"class="m-2 text-
center"><b>Hello</b></div>
   <input type="text"#mark(keyup)="display(mark)"class="m-2">
   <input type="text"#subject(keydown)="func(subject)"class="m-2">
   <input type="text"#name(keyup.enter)="show(name)"class="m-2">
   <button type="button"(click)="submit(name,mark,subject)"class="m-
2">Submit</button>
 </div>
```

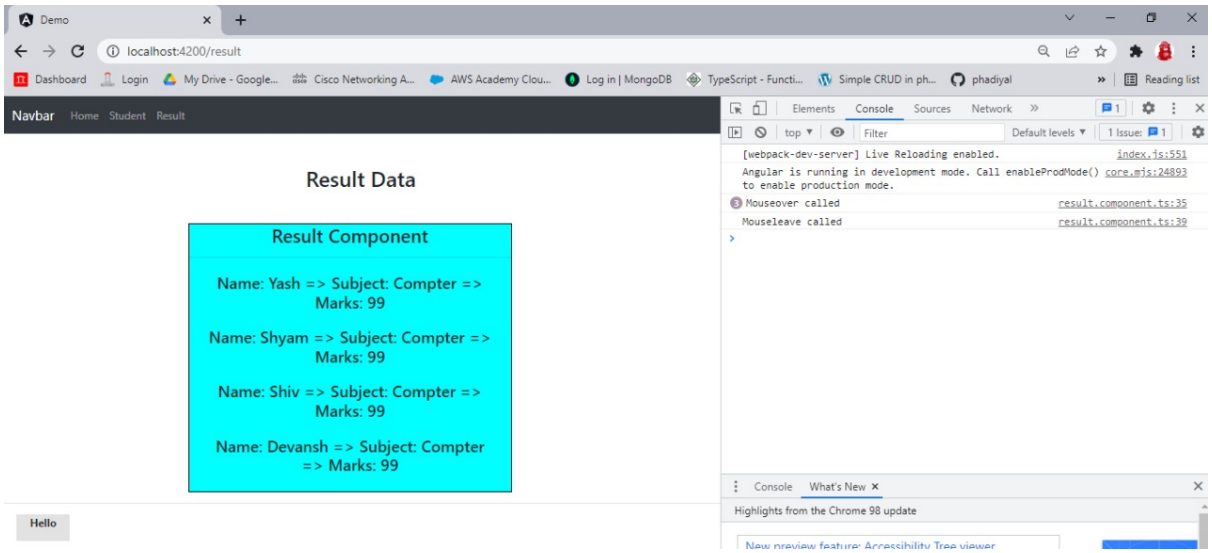
Result.component.css

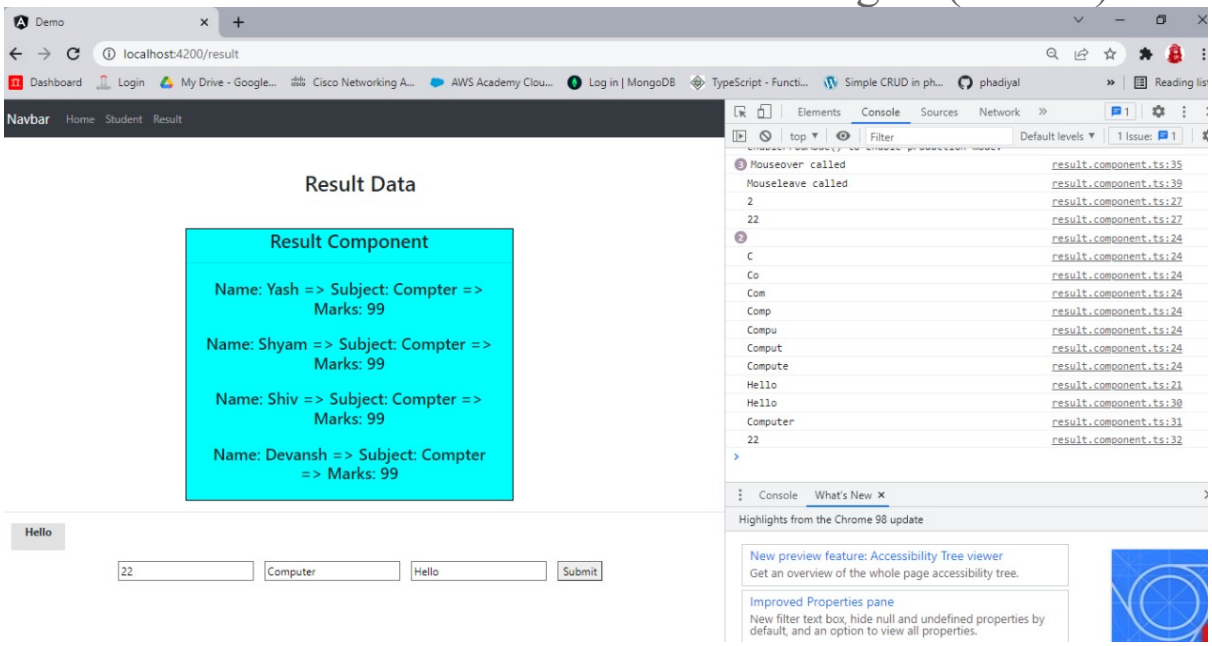
```
.center{
   border: 1pxsolidblack;
   background-color: aqua;
```

```
    width: 45%;
    margin-left: 26%;
}
```

Output:-







Practice for Property Binding in Angular

Display the list of students with its profile image, erno, Full Name, DOB, Department. Designing should be proper using card deck and card of BS4 (Bootstrap 4) or ngBootStrap.

Code:

NavBar:

Navbar.component.html

```
<nav class="navbar navbar-expand-md bg-primary navbar-dark">
    <a class="navbar-brand" href="#">20SOECE13047</a>
    <button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#collapsibleNavbar">
      <span class="navbar-toggler-icon"></span>
    </button>
    <div class="collapse navbar-collapse" id="collapsibleNavbar">
      <ul class="navbar-nav nav-pills card-header-pills">
        <li class="nav-item ">
          <a class="nav-link" routerLink="/home"   href="#">Home</a>
        </li>
        <li class="nav-item">
          <a class="nav-link active bg-warning" routerLink="/student"
href="#">Student</a>
        </li>
```

Student:

Student.component.html

```
<div class="container">
<div class="row col-md-3">
  <div class="card-columns m-3">
      <div *ngFor="let stud of students; let i = index" class="card">
        <div class="card-body text-center">
            <img [src]="stud.display">
        </div>
```

```
            <hr>
            <div class="text bg-primary text-white p-2">
                <h5 class="card-text">Name: {{stud.name}}</h5>
                <h5 class="card-text">DOB: {{stud.DOB}}</h5>
                <h5 class="card-text">Department: {{stud.Department}}</h5>
            </div>
          </div>
      </div>
```

Student.component.css

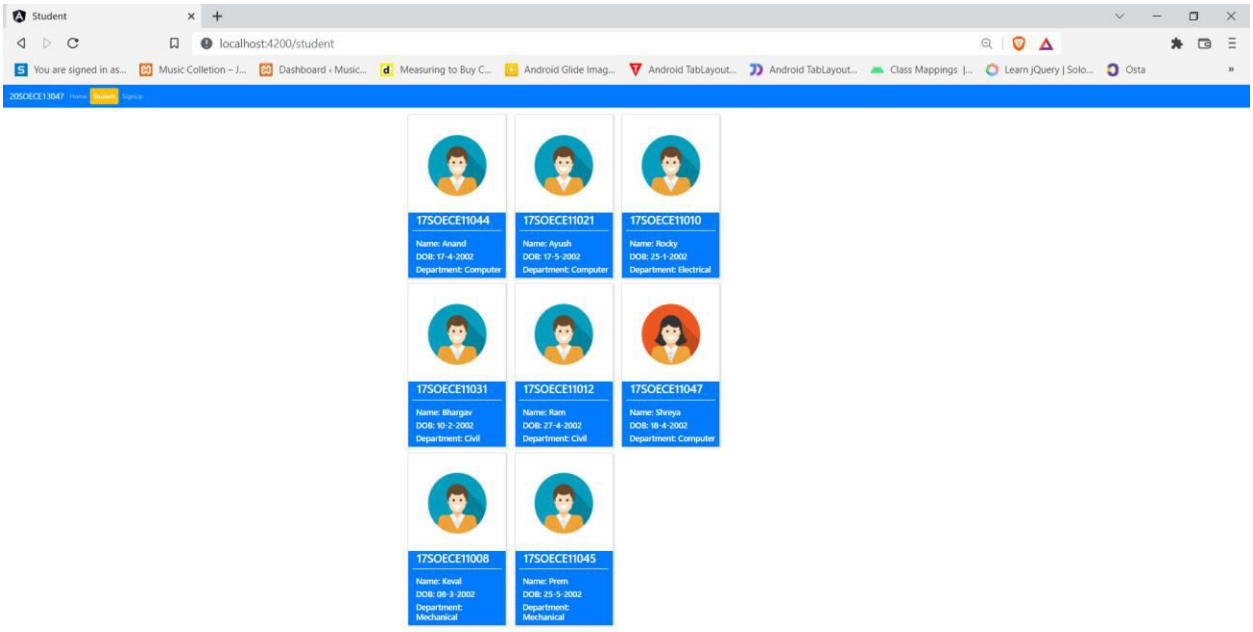
```
img{
    height: 200px;
    width: 200px;
}
hr{
    border-top: 2px solid white;
    margin: 10px;
    text-align: justify;
}
.text{
    margin-left: 10px;
}
h3{
    margin-left: 20px;
```

Student.component.ts

```
import { Component, OnInit } from '@angular/core';


@Component({
  selector: 'app-student',
  templateUrl: './student.component.html',
```

```
export class StudentComponent implements OnInit {



  students:any[]=[
    {id:1,display:"../assets/images/1.png",name:"Anand",EnrolmentNo:"17SOECE1
1044 ",DOB:"17-4-2002",Department:"Computer"},
    {id:2,display:"../assets/images/1.png",name:"Bhargav",EnrolmentNo:"17SOECE
110 31",DOB:"10-2-2002",Department:"Civil"},
    {id:3,display:"../assets/images/1.png",name:"Keval",EnrolmentNo:"17SOECE11
008 ",DOB:"08-3-2002",Department:"Mechanical"},
    {id:4,display:"../assets/images/1.png",name:"Ayush",EnrolmentNo:"17SOECE1
1021 ",DOB:"17-5-2002",Department:"Computer"},
    {id:5,display:"../assets/images/1.png",name:"Ram",EnrolmentNo:"17SOECE110
12", DOB:"27-4-2002",Department:"Civil"},
     {id:6,display:"../assets/images/1.png",name:"Prem",EnrolmentNo:"17SOECE11
     045"
,DOB:"25-5-2002",Department:"Mechanical"},
    {id:7,display:"../assets/images/1.png",name:"Rocky",EnrolmentNo:"17SOECE1
1010 ",DOB:"25-1-2002",Department:"Electrical"},
    {id:8,display:"../assets/images/3.jpg",name:"Shreya",EnrolmentNo:"17SOECE1
104 7",DOB:"18-4-2002",Department:"Computer"},
  ]

  constructor() { }

  ngOnInit(): void {
  }

}
```

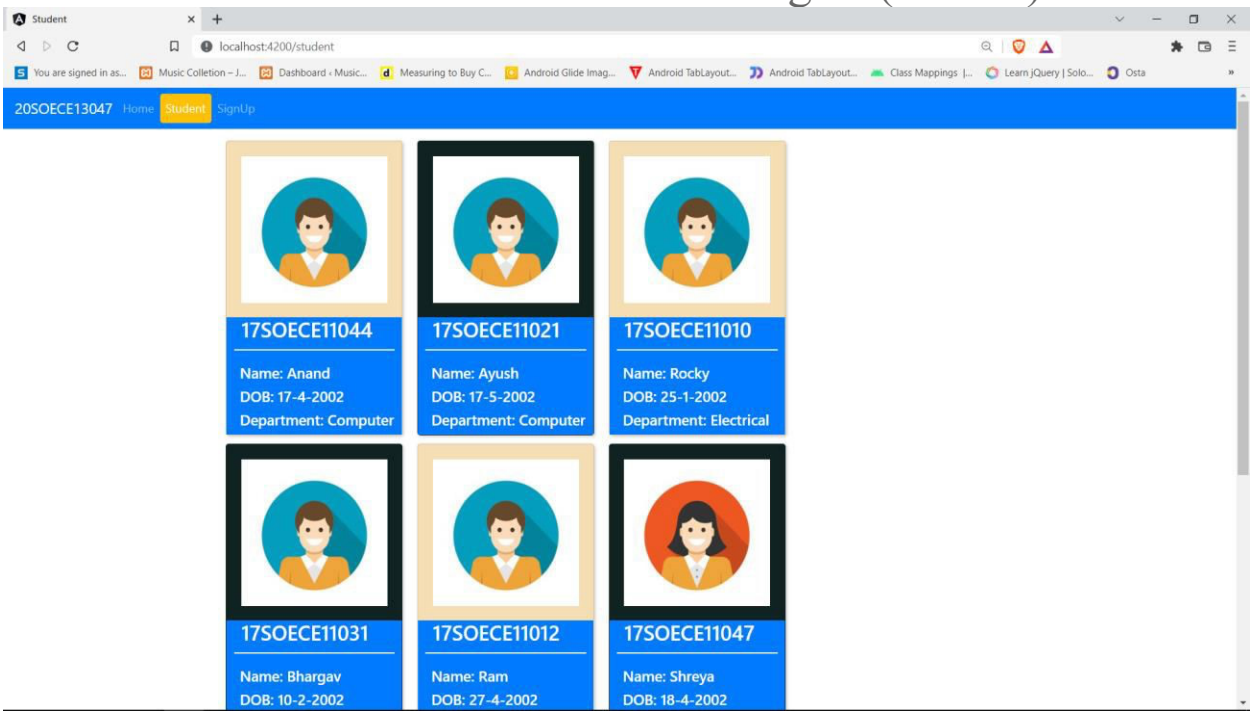
Output:



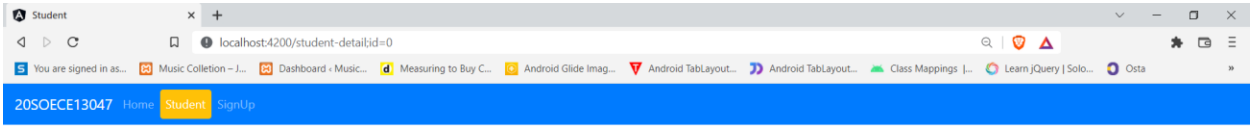
Practice with ngIf and routes with Parameter

1. Modify the existing project such that odd and even cards of students should have different style binding (use *ngIf and [ngStyle].

Output:

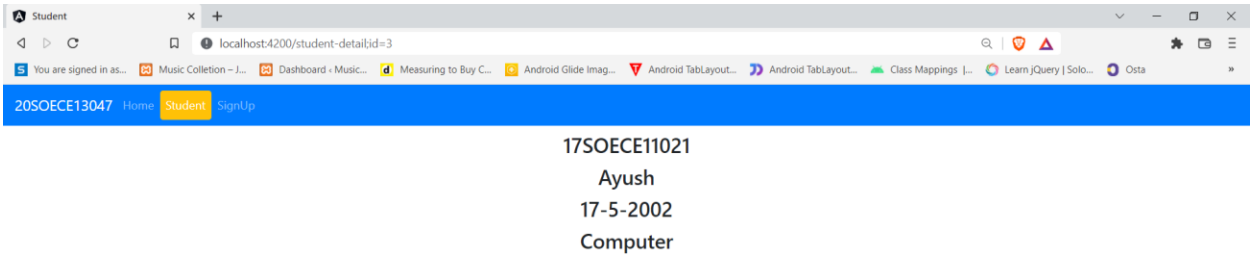


Even:



17SOECE11044
Anand
17-4-2002
Computer

Odd:



Code:

Studen t:

student.component.css:-

```css
img{
    height: 200px;
    width: 200px;
}
hr{
    border-top: 2px solid white;
    margin: 10px;
    text-align: justify;
}
.text{
    margin-left: 10px;
}
h3{
```

```
.even {
    background-color: wheat;
  }

.odd {
```

student.component.html:-

```
<div class="container">
<div class="row col-md-3">
  <div class="card-columns m-3">
      <div *ngFor="let stud of students; index as i; odd as isOdd; even as
isEven;"
      [ngClass]="{even: isEven, odd: isOdd}" class="card">
        <div class="card-body text-center">
            <img [src]="stud.display">
        </div>
          <div [routerLink]="['/student-detail',{id:i}]" class="text-Uppercase
bg-primary text-white">
            <h3>{{stud.EnrolmentNo}}</h3>
            <hr>
            <div class="text bg-primary text-white p-2">
                <h5 class="card-text">Name: {{stud.name}}</h5>
                <h5 class="card-text">DOB: {{stud.DOB}}</h5>
```

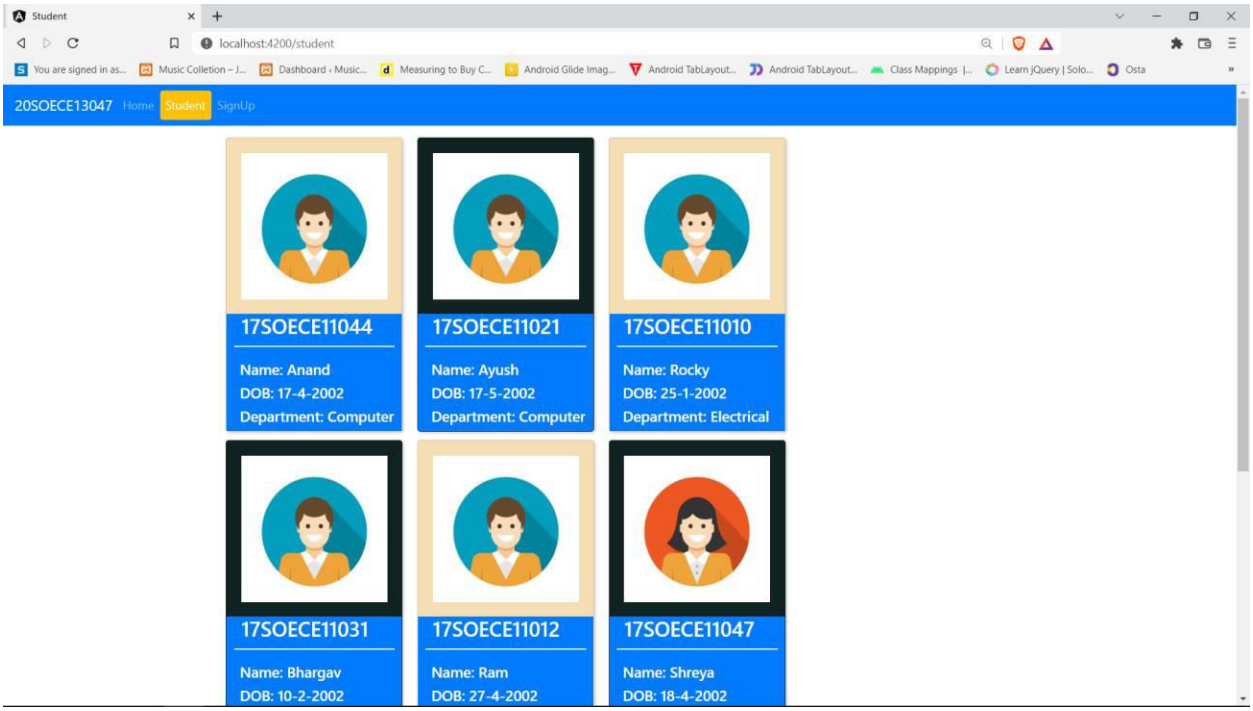
student.component.ts:-

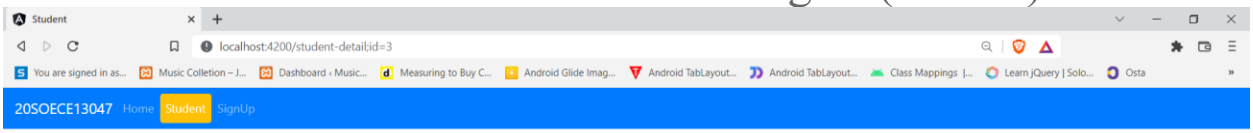
```
import { Component, OnInit } from '@angular/core';


@Component({
  selector: 'app-student',
```

```
})
export class StudentComponent implements OnInit {

    visible = true;

    students:any[]=[
      {id:1,display:"../assets/images/1.png",name:"Anand",EnrolmentNo:"17SOECE1
1044 ",DOB:"17-4-2002",Department:"Computer"},
      {id:2,display:"../assets/images/1.png",name:"Bhargav",EnrolmentNo:"17SOECE
110 31",DOB:"10-2-2002",Department:"Civil"},
      {id:3,display:"../assets/images/1.png",name:"Keval",EnrolmentNo:"17SOECE11
008 ",DOB:"08-3-2002",Department:"Mechanical"},
      {id:4,display:"../assets/images/1.png",name:"Ayush",EnrolmentNo:"17SOECE1
1021 ",DOB:"17-5-2002",Department:"Computer"},
      {id:5,display:"../assets/images/1.png",name:"Ram",EnrolmentNo:"17SOECE110
12", DOB:"27-4-2002",Department:"Civil"},
        {id:6,display:"../assets/images/1.png",name:"Prem",EnrolmentNo:"17SOECE11
        045"
,DOB:"25-5-2002",Department:"Mechanical"},
      {id:7,display:"../assets/images/1.png",name:"Rocky",EnrolmentNo:"17SOECE1
1010 ",DOB:"25-1-2002",Department:"Electrical"},
      {id:8,display:"../assets/images/3.jpg",name:"Shreya",EnrolmentNo:"17SOECE1
104 7",DOB:"18-4-2002",Department:"Computer"},
    ]

    constructor() { }

    ngOnInit(): void {
    }

}
```

2. Provide routes with parameter. When any card of student is clicked from "studentList" component, display the details of that particular student in newly created "student Component".

Output:-


Student
localhost:4200/student
20SOECE13047 Home Student SignUp
17SOECE11044
Name: Anand
DOB: 17-4-2002
Department: Computer
17SOECE11021
Name: Ayush
DOB: 17-5-2002
Department: Computer
17SOECE11010
Name: Rocky
DOB: 25-1-2002
Department: Electrical
17SOECE11031
Name: Bhargav
DOB: 10-2-2002
17SOECE11012
Name: Ram
DOB: 27-4-2002
17SOECE11047
Name: Shreya
DOB: 18-4-2002



17SOECE11021
Ayush
17-5-2002
Computer

Code:

student-detail:

student-detail.component.css:-

```
h3{
    text-align: center;
    margin-top: 10px;
```

student-detail.component.html:-

```
<h3>{{student.EnrolmentNo}}</h3>
<h3>{{student.name}}</h3>
<h3>{{student.DOB}}</h3>
```

student-detail.component.ts:-

```
import { Component, OnInit } from '@angular/core';
import { ActivatedRoute } from '@angular/router';

@Component({
  selector: 'app-student-detail',
  templateUrl: './student-detail.component.html',
  styleUrls: ['./student-detail.component.css']
})
export class StudentDetailComponent implements OnInit {

  students:any[]=[
   {id:1,display:"../assets/images/1.png",name:"Anand",EnrolmentNo:"17SOECE1
```

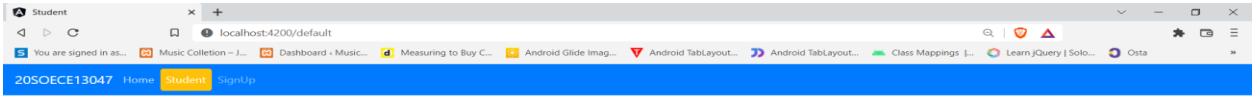
1044 ",DOB:"17-4-2002",Department:"Computer"},
{id:2,display:"../assets/images/1.png",name:"Bhargav",EnrolmentNo:"17SOECE
110 31",DOB:"10-2-2002",Department:"Civil"},
{id:3,display:"../assets/images/1.png",name:"Keval",EnrolmentNo:"17SOECE11
008 ",DOB:"08-3-2002",Department:"Mechanical"},
{id:4,display:"../assets/images/1.png",name:"Ayush",EnrolmentNo:"17SOECE1
1021 ",DOB:"17-5-2002",Department:"Computer"},
{id:5,display:"../assets/images/1.png",name:"Ram",EnrolmentNo:"17SOECE110
12", DOB:"27-4-2002",Department:"Civil"},
{id:6,display:"../assets/images/1.png",name:"Prem",EnrolmentNo:"17SOECE11
045"
,DOB:"25-5-2002",Department:"Mechanical"},
{id:7,display:"../assets/images/1.png",name:"Rocky",EnrolmentNo:"17SOECE1
1010 ",DOB:"25-1-2002",Department:"Electrical"},
{id:8,display:"../assets/images/3.jpg",name:"Shreya",EnrolmentNo:"17SOECE1
104 7",DOB:"18-4-2002",Department:"Computer"},
]

```
constructor(private _activatedRoute:ActivatedRoute) { }

student:any=null;

ngOnInit(): void {
  this._activatedRoute.params.subscribe(param=>{


    this.student=this.students[param['id']]

  })
}
```

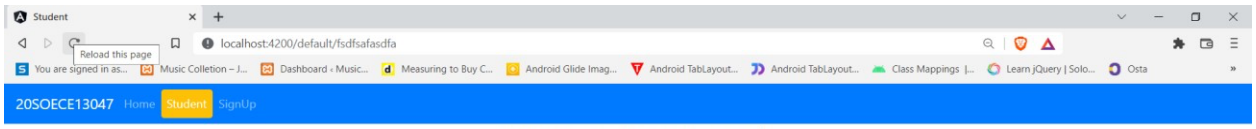
app-routing.module.ts:-

```
import { NgModule } from '@angular/core';

import { RouterModule, Routes } from '@angular/router';

import { DefaultComponent } from './default/default.component';
import { NotfoundComponent } from './notfound/notfound.component';
import { SignupComponent } from './signup/signup.component';
import { StudentDetailComponent } from './student-detail/student-
detail.component';

import { StudentComponent } from './student/student.component';

const routes: Routes = [


  {path:"default",component:DefaultComponent},

  {path:"student",component:StudentComponent},

  {path:"student-detail",component:StudentDetailComponent},

  {path:"student-detail/:id",component:StudentDetailComponent},

  {path:'signup',component:SignupComponent},

  {path:'',redirectTo:'default',pathMatch:'full'},

  {path:'**',component:NotfoundComponent}


];
```

3. Provide default and 404NotFound routes for the application. Output:



Welcome Student!.



404 Page Not Found

Code: default:

default.component.css:-

```
h1{
    margin-top: 20%;
    text-align: center;
```

default.component.html:-

```
<h1>
    Welcome Student!.
```

notFound:
notfound.component.css:-

```
h1{
    text-align:center;
```

notfound.component.html:-

```
<h1>
    404 Page Not Found
```