**CAPSTONE I (CS 392)**

**LABORATORY MANUAL**

**B.Tech. (CSE/ DS/CYS/iMSC)**

**Semester II**

**L-T-P-C: 2-0-4-4**

**Academic Year: 2024-25**

**A logo of a university

Description automatically generated**

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

|  |  |
| --- | --- |
| Student ID | BT22GDS056 |
| Name | Agnishwar Raychaudhuri |

**NIIT UNIVERSITY, NEEMRANA**

NH-8, Delhi-Jaipur Highway,  
Neemrana, District Alwar (Rajasthan),  
Pin-301705

Website: [www.niituniversity.in](http://www.niituniversity.in)

**VISION OF THE UNIVERSITY**

To be the role model of learning, research, innovation, and sustainability for the knowledge society.

**MISSION OF THE UNIVERSITY**

1. To deliver distinctive education through the pioneering use of technology.
2. To develop superior talent through partnerships with industry and society.
3. To promote research, discovery, and entrepreneurship through collaborative action.
4. To build responsible citizens of the world by instilling a culture of seamlessness in all facets of life.

**VISION OF THE DEPARTMENT**

NIIT University emerged from the visionary aspirations of esteemed intellectuals who sought to revolutionize the Indian education system, with a focus on cultivating original thinkers, innovative problem solvers, and inspiring leaders. Their goal was to foster a dynamic and intellectually stimulating learning environment that seamlessly blends academic theories with real-life experiences. NU's Computer Science and Engineering (CSE) programmes (Undergraduate, Postgraduate, and PhD) are designed to provide a comprehensive understanding of the foundational principles of computing and equip students with the necessary engineering skills for designing, implementing, and utilizing computer systems. This course offers a holistic perspective on the field of computing, encompassing the concepts, principles, and practical applications, allowing students to gain both theoretical knowledge and hands-on experience. By enrolling in our CSE program, students will embark on a journey that not only acquaints them with the intricacies of computing but also nurtures their ability to apply and extend these concepts. Our curriculum is carefully crafted to ensure that students develop a well-rounded skill set, encompassing theoretical understanding, practical skills, and the capacity to apply these principles in real-world scenarios.

**MISSION OF THE DEPARTMENT**

The area of Computer Science and Engineering at NIIT University is dedicated to fulfilling the following mission:

M1: Providing the ideal environment for students to become industry-ready professionals, researchers, and entrepreneurs. We achieve this by offering courses on cutting-edge technology and advanced laboratory courses, ensuring that our students are well-equipped with the knowledge and skills demanded by the industry.

M2: Establishing Centres of Excellence that foster a conducive environment for our faculty to engage in progressive and convergent research themes. Through these centers, we aim to train our students in the latest advancements and emerging technologies, ensuring that they stay at the forefront of the field.

M3: Imparting high-quality experiential learning to our students, enabling them to gain expertise in modern software tools and cater to the real-time requirements of the industry. We focus on providing practical knowledge and hands-on experience to ensure our students are well-prepared for their professional careers.

M4: Instilling problem-solving and team-building skills among our students, emphasizing the importance of lifelong learning, and nurturing a sense of societal and ethical responsibilities. We believe in creating well-rounded professionals who are not only skilled in their domain but also possess the necessary attributes to make a positive impact on society.

M5: Offering continuing education programs in emerging areas of computer science, such as cybersecurity, data science, machine learning, and cloud computing. These programs are designed to provide ongoing learning opportunities for our stakeholders, including professionals already working in the industry, ensuring that they stay updated with the latest advancements.

At NIIT University, we are committed to providing a comprehensive and holistic learning experience to our students, empowering them to excel in their chosen fields and make significant contributions to the ever-evolving world of computer science and engineering.

**PROGRAM OVERVIEW**

The Computer Science and Engineering at NIIT University seems to have a strong focus on providing quality education and preparing students for practical challenges in the field. This approach is valuable in ensuring that students are equipped with the necessary skills and knowledge to contribute to fundamental research in computer science and engineering. By emphasizing real-world challenges, the area likely provides a curriculum that integrates theoretical knowledge with hands-on experience, preparing students for the demands of the industry. The University is actively seeking out students with a passion for innovation and a drive to contribute to the field. This approach can foster an environment that encourages creativity, problem-solving, and critical thinking, which are all essential qualities for success in computer science and engineering.

**PROGRAM OUTCOMES (POs)**

**PO1: Engineering Knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

**PO2: Problem Analysis:** Identify, formulate, research literature, and analyze complex engineering problems reaching substantiated conclusions using the first principles of mathematics, natural sciences, and engineering sciences.

**PO3: Design/Development of Solutions:** Design solutions for complex engineering problems and system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, society, and environmental considerations.

**PO4: Conduct Investigations of Complex Problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

**PO5: Modern Tool Usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

**PO6: The Engineer and Society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

**PO7: Environment and Sustainability:** Understand the impact of the professional engineering solutions in society and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

**PO8: Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

**PO9: Individual and Team Work**: Function effectively as an individual, and as a member or leader in diverse teams, and in multi-disciplinary settings.

**PO10: Communication**: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

**PO11: Project Management and Finance**: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one’s own work, as a member and leader in a team, to manage projects and in multi-disciplinary environments.

**PO12: Life-long Learning**: Recognize the need for and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

**PROGRAM SPECIFIC OUTCOMES (PSOs)**

**PSO 1:** Ability to apply the knowledge of basic and advanced concepts of Computer Science in order to address compelling problems of society and provide an acceptable and practical solution.

**PSO 2:** Ability to understand the mathematical background to solve computational tasks and model solutions to real-life problems.

**PSO 3:** Ability to understand state-of-the-art research problems and identify the research gaps so as to provide innovative and effective methods to fill-up those gaps.

**PSO 4:** Use new technical advancements of Computer Science to produce appreciable contributions in the profession.

**PSO 5:** Ability to analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions.

**PSO 6:** Ability to model data and design the system architecture of high-performance and tolerant systems.

**PSO 7:** Ability to formulate learning solutions for process-centric systems.

**PSO 8:** Ability to achieve a successful career in Industry, in Research Institute, in Academics, and in Entrepreneurship.

**PSO 9:** Ability to publish technical papers in reputed journals and conferences under the guidance of dedicated faculty members.

**PSO 10:** Ability to get involved in innovative, industry-relevant projects jointly with world class engineers in Collaborative Education mode which will provide an environment to convert their innovation into reality.

**PSO 11:** Develop expertise in specialized areas like Data Science, Artificial Intelligence, Cyber Security, Cloud Computing and Big data Engineering.

**PSO 12:** Ability to seamlessly become industry professional after six months Industry Practice in Industry Environment in the final academic year.

**PROGRAMME EDUCATIONAL OBJECTIVES (PEOs)**

**PEO 1:** The Computer Science and Engineering program at NIIT University is designed to empower graduates with the skills and knowledge necessary to achieve the following educational objectives.

**PEO 2:** Thrive as proficient computer professionals: Our graduates will be equipped to work productively as successful computer professionals in a diverse range of career paths. Whether in supportive or leadership roles within multidisciplinary teams, they will possess the competence to contribute effectively. Additionally, they will have the option to pursue higher studies to further enhance their expertise in the field.

**PEO 3:** Demonstrate effective communication and ethical conduct: We emphasize the development of excellent communication skills among our graduates, enabling them to effectively convey ideas and solutions.

**PEO 4:** CSE program instills a strong awareness of societal needs and constraints, empowering graduates to incorporate these considerations into their professional endeavors. With a high regard for ethical responsibilities, they will contribute to the betterment of society through their practice.

**PEO 5:** Embrace lifelong learning for personal and organizational growth: Our program encourages graduates to engage in lifelong learning, ensuring they remain up-to-date in their profession. By nurturing a mindset of continuous growth, they will adapt to evolving technologies and industry trends, fostering personal development and contributing to the growth of the organizations they join. Through their commitment to ongoing learning, our graduates will stay at the forefront of the field.

**GENERAL LABORATORY INSTRUCTIONS**

* + - 1. Students are advised to come to the laboratory at least 5 minutes before (to the starting time), those who come after 5 minutes will not be allowed into the lab.

2. Plan your task properly before the commencement, come prepared to the lab with the synopsis/ program/experiment details.

3. Students should enter the laboratory with:

* + - * 1. Laboratory observation notes with all the details (Problem statement, Aim, Algorithm, Procedure, Program, Expected Output, etc.,) filled in for the lab session.
  1. Laboratory Record updated up to the last session experiments and other utensils (if any) needed in the lab.
  2. Proper Dress code and Identity card.

4. Execute your task in the laboratory, record the results/output, and get certified by the concerned faculty.

5. All the students should be polite and cooperative with the laboratory staff, and must maintain discipline and decency in the laboratory.

6. Students / Faculty must keep their mobile phones in SWITCHED OFF mode during the lab sessions. Misuse of the equipment and misbehavior with the staff will attract severe punishment.

7. Students must get the permission of the faculty in case of any urgency to go out; if anybody is found outside the lab/class without permission during working hours will be treated seriously and punished appropriately.

8. Students should LOG OFF/ SHUT DOWN the computer system before he/she leaves the lab after completing the task (experiment) in all aspects. He/she must ensure the system/seat is kept properly.

**COURSE DESCRIPTION**

By empowering students with the skills necessary to access, understand and evaluate Information, Capstone Project will support students in the achievement of 21st century learning expectations. Students work in teams to develop or implement a real world IT solution

**COURSE OUTCOMES**

|  |  |
| --- | --- |
| **S.No.** | **Description** |
| CO1 | Understand the basics of Web Application and apply them in their project. |
| CO2 | Learn to collaborate development using Version Control. |
| CO3 | Applying the process of Full Stack Development and best practices using cutting Edge Technology stack. |
| CO4 | Apply Continuous Integration and Continuous Deployment |
| CO5 | Understand and apply containerization |

**EVALUATION CRITERIA**

|  |  |
| --- | --- |
| **Component** | **Marks** |
| **Project Presentation 1** | **15** |
| **Lab Assignment** | **10** |
| **Project Presentation 2** | **15** |

**FACULTY DETAILS**

Course In charge: Prof. Manish Hurkat (manish.hurkat@niituniversity.in)

**List of Experiments**

|  |  |  |  |
| --- | --- | --- | --- |
| **Practical No.** | **Aim of Practical** | **COs** | **Page No.** |
| 1 | Collaboration Tool-Git | CO2 | 10 |
| 2 | HTML and CSS | CO1 | 10 |
| 3 | Javascript | CO1 | 10 |
| 4 | BootStrap | CO1 | 19 |
| 5 | NodeJS | CO3 | 19 |
| 6 | ExpressJS | CO3 | 19 |
| 7 | MongoDB | CO3 | 20 |
| 8 | Database Connectivity | CO3 | 27 |
| 9 | Login Exercise | CO3 | 27 |
| 10 | Session and Cookies Demo | CO3 | 28 |
| 11 | Setting up DevOPS for Project | CO4  CO5 | 28 |
| 12 | APIs | CO4  CO5 | 28 |

**Practical 1:** Collaboration Tool-Git:

**Steps of Execution:**

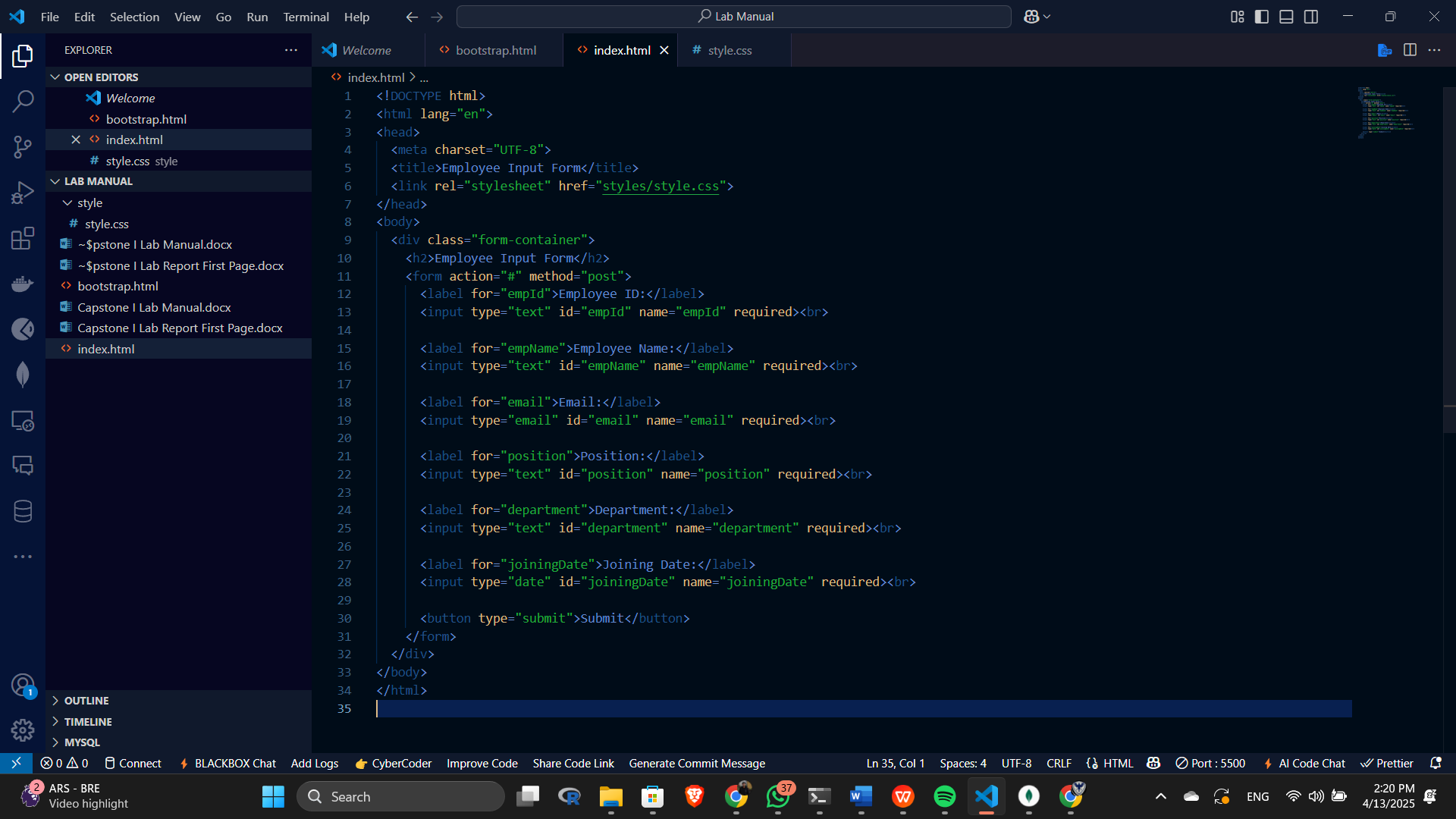
1. Team lead will make Github repository and will send invite to other team members.
2. Team lead will create local repo and then push to remote repo
3. Team members will accept the invite and clone repo on their local machine.
4. All the team members will share file using git / github
5. Team members will update their repository and check the contents shared by others.

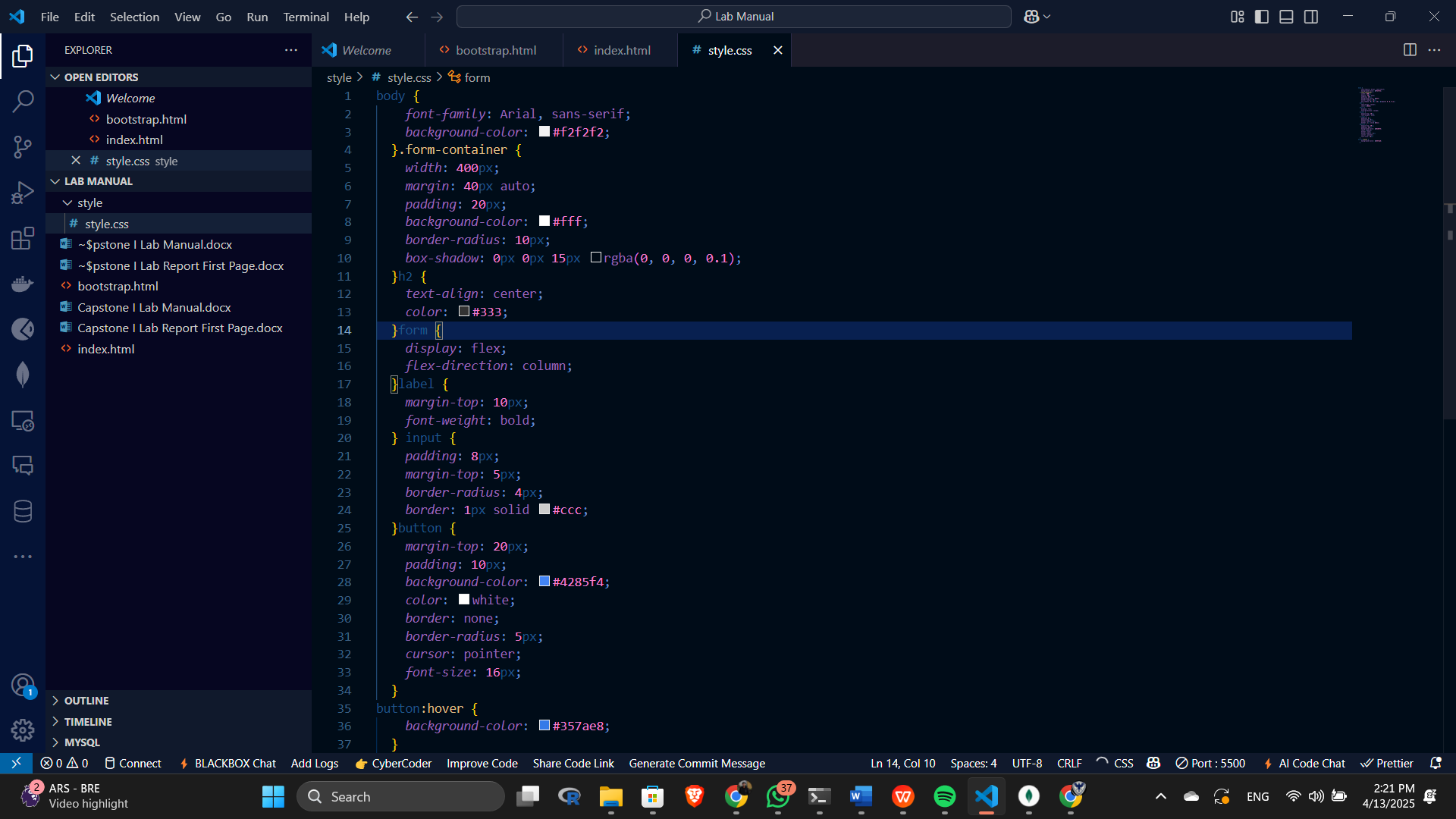
**Practical 2:** HTML and CSS:

**Steps of Execution:**

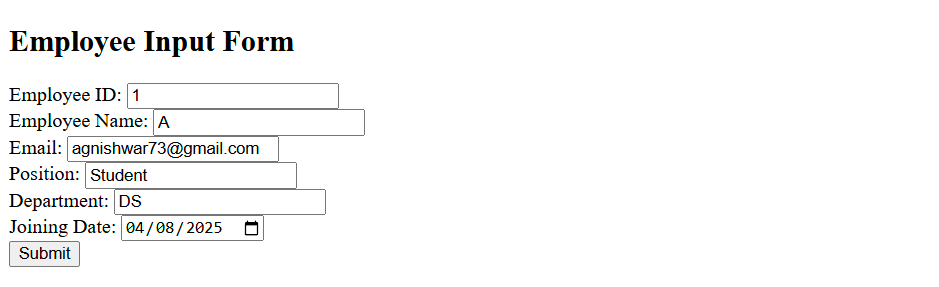
1. Make an input form for an employee.
2. Apply the relevant CSS using external CSS file

**Code**





**Output**



**Practical 3:** Javascript:

Write a JavaScript program to display the current day and time in the following format.  
Today is : Tuesday.   
Current time is : 10 PM : 30 : 38

**JavaScript Code:**

// Get the current date and time

var today = new Date();

// Get the day of the week (0-6, where 0 is Sunday and 6 is Saturday)

var day = today.getDay();

// Array of day names

var daylist = ["Sunday", "Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday"];

// Display the current day

console.log("Today is: " + daylist[day] + ".");

// Get the current hour, minute, and second

var hour = today.getHours();

var minute = today.getMinutes();

var second = today.getSeconds();

// Determine if it's AM or PM

var prepand = (hour >= 12) ? " PM " : " AM ";

// Convert 24-hour format to 12-hour format

hour = (hour >= 12) ? hour - 12 : hour;

// Check for special cases when hour is 0

if (hour === 0 && prepand === ' PM ') {

if (minute === 0 && second === 0) {

hour = 12;

prepand = ' Noon';

} else {

hour = 12;

prepand = ' PM';

}

}

// Check for special cases when hour is 0

if (hour === 0 && prepand === ' AM ') {

if (minute === 0 && second === 0) {

hour = 12;

prepand = ' Midnight';

} else {

hour = 12;

prepand = ' AM';

}

}

// Display the current time

console.log("Current Time: " + hour + prepand + " : " + minute + " : " + second);

Sample Output:

Today is : Tuesday.

Current Time : 10 PM : 30 : 38

Write a JavaScript program to determine whether a given year is a leap year in the Gregorian calendar.

**JavaScript Code:**

// Define a function to check if a given year is a leap year

function leapyear(year) {

// Return true if the year is divisible by 4 but not divisible by 100 unless it's also divisible by 400

return (year % 100 === 0) ? (year % 400 === 0) : (year % 4 === 0);

}

// Test the function with various years and log the results to the console

console.log(leapyear(2016)); // Expected output: true

console.log(leapyear(2000)); // Expected output: true

console.log(leapyear(1700)); // Expected output: false

console.log(leapyear(1800)); // Expected output: false

console.log(leapyear(100)); // Expected output: false

Sample Output:

true

true

false

false

false

Write a JavaScript function that takes a callback and invokes it after a delay of 2 second.

**Sample Solution:**

**JavaScript Code:**

function invokeAfterDelay(callback) {

setTimeout(callback, 2000); // 2000 milliseconds = 2 second

}

function display\_message() {

console.log('Hello!');

}

invokeAfterDelay(display\_message); // Invokes the sayHello function after a 1-second delay

Sample Output:

"Hello!"

Write a JavaScript program that converts a callback-based function to a Promise-based function.

**Promise:**

The Promise object represents the eventual completion (or failure) of an asynchronous operation and its resulting value.

A Promise is a proxy for a value not necessarily known when the promise is created. It allows you to associate handlers with an asynchronous action's eventual success value or failure reason. This lets asynchronous methods return values like synchronous methods: instead of immediately returning the final value, the asynchronous method returns a promise to supply the value at some point in the future.

A Promise is in one of these states:

* pending: initial state, neither fulfilled nor rejected.
* fulfilled: meaning that the operation was completed successfully.
* rejected: meaning that the operation failed.

**Sample Solution:**

**JavaScript Code:**

function callback\_BasedFunction(arg1, arg2, callback) {

// Perform asynchronous operations

// Call the callback with the result or error

setTimeout(() => {

const result = arg1 + arg2;

if (result % 2 !== 0) {

callback(null, result);

} else {

callback(new Error('Result is not odd!'), null);

}

}, 1000);

}

function promisifiedFunction(arg1, arg2) {

return new Promise((resolve, reject) => {

callback\_BasedFunction(arg1, arg2, (error, result) => {

if (error) {

reject(error); // Reject the Promise with the error

} else {

resolve(result); // Resolve the Promise with the result

}

});

});

}

// Usage example:

promisifiedFunction(2, 3)

.then(result => {

console.log('Result:', result);

})

.catch(error => {

console.log('Error:', error.message);

});

promisifiedFunction(2, 4)

.then(result => {

console.log('Result:', result);

})

.catch(error => {

console.log('Error:', error.message);

});

Sample Output:

"Result:"

5

"Error:"

"Result is not odd!"

Write a JavaScript a function that makes an HTTP GET request and returns a Promise that resolves with the response data.

**Sample Solution:**

**JavaScript Code:**

function make\_Get\_Request(url) {

return new Promise((resolve, reject) => {

fetch(url)

.then(response => {

if (!response.ok) {

throw new Error(`HTTP error! Status: ${response.status}`);

}

return response.json();

})

.then(data => resolve(data))

.catch(error => reject(error));

});

}

make\_Get\_Request('https://example.com/data')

.then(data => {

console.log('Response data: ' + data);

})

.catch(error => {

console.log('Error: ' + error.message);

});

Sample Output:

"Error:"

"Failed to fetch"

Write a JavaScript function that takes an array of URLs and downloads the contents of each URL in parallel using Promises.

**Sample Solution:**

**JavaScript Code:**

function downloadContents(urls) {

const promises = urls.map(url => {

return new Promise((resolve, reject) => {

fetch(url)

.then(response => {

if (!response.ok) {

throw new Error(`HTTP error! Status: ${response.status}`);

}

return response.text();

})

.then(data => resolve(data))

.catch(error => reject(error));

});

});

return Promise.all(promises);

}

// Usage example:

const urls = [

'https://jsonplaceholder.typicode.com/posts/1',

'https://jsonplaceholder.typicode.com/posts/2',

'https://jsonplaceholder.typicode.com/posts/3'

];

downloadContents(urls)

.then(contents => {

console.log('Downloaded contents:', contents);

})

.catch(error => {

console.log('Error:', error.message);

});

Sample Output:

"Downloaded contents:"

["{

\"userId\": 1,

\"id\": 1,

\"title\": \"sunt aut facere repellat provident occaecati excepturi optio reprehenderit\",

\"body\": \"quia et suscipit\nsuscipit recusandae consequuntur expedita et cum\nreprehenderit molestiae ut ut quas totam\nnostrum rerum est autem sunt rem eveniet architecto\"

}", "{

\"userId\": 1,

\"id\": 2,

\"title\": \"qui est esse\",

\"body\": \"est rerum tempore vitae\nsequi sint nihil reprehenderit dolor beatae ea dolores neque\nfugiat blanditiis voluptate porro vel nihil molestiae ut reiciendis\nqui aperiam non debitis possimus qui neque nisi nulla\"

}", "{

\"userId\": 1,

\"id\": 3,

\"title\": \"ea molestias quasi exercitationem repellat qui ipsa sit aut\",

\"body\": \"et iusto sed quo iure\nvoluptatem occaecati omnis eligendi aut ad\nvoluptatem doloribus vel accusantium quis pariatur\nmolestiae porro eius odio et labore et velit aut\"

}"]

Write a JavaScript program to sort the items of an array.

*Sample array* : var arr1 = [ -3, 8, 7, 6, 5, -4, 3, 2, 1 ];  
*Sample Output* : -4,-3,1,2,3,5,6,7,8

**Visual Presentation:**



**Sample Solution:**

**JavaScript Code:**

// Declare and initialize the original array

var arr1 = [-3, 8, 7, 6, 5, -4, 3, 2, 1];

// Declare an empty array to store the sorted result

var arr2 = [];

// Initialize variables to track the minimum, maximum, and position in the original array

var min = arr1[0];

var pos;

var max = arr1[0];

// Find the maximum value in the original array

for (i = 0; i < arr1.length; i++) {

if (max < arr1[i]) max = arr1[i];

}

// Selection sort algorithm: Iterate over the original array

for (var i = 0; i < arr1.length; i++) {

// Iterate over the original array to find the minimum element

for (var j = 0; j < arr1.length; j++) {

// Check if the element is not marked as processed (not equal to "x")

if (arr1[j] != "x") {

// Find the minimum element and its position in the original array

if (min > arr1[j]) {

min = arr1[j];

pos = j;

}

}

}

// Store the minimum element in the sorted array

arr2[i] = min;

// Mark the minimum element as processed by replacing it with "x" in the original array

arr1[pos] = "x";

// Reset min to the maximum value for the next iteration

min = max;

}

// Output the sorted array

console.log(arr2);

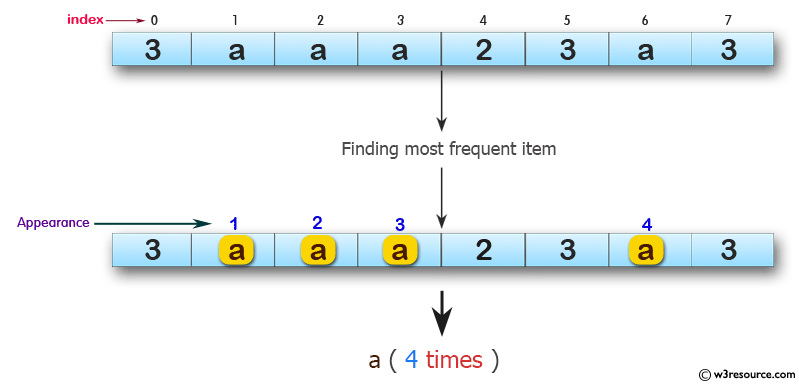
Sample Output:

[-4,-3,1,2,3,5,6,7,8]

Write a JavaScript program to find the most frequent item in an array.

*Sample array*: var arr1=[3, 'a', 'a', 'a', 2, 3, 'a', 3, 'a', 2, 4, 9, 3];  
*Sample Output*: a ( 5 times )

**Visual Presentation:**



**Sample Solution:**

**JavaScript Code:**

// Declare and initialize the original array

var arr1 = [3, 'a', 'a', 'a', 2, 3, 'a', 3, 'a', 2, 4, 9, 3];

// Initialize variables to track the most frequent item, its frequency, and the current item's frequency

var mf = 1;

var m = 0;

var item;

// Iterate through the array to find the most frequent item

for (var i = 0; i < arr1.length; i++) {

// Nested loop to compare the current item with others in the array

for (var j = i; j < arr1.length; j++) {

// Check if the current item matches with another item in the array

if (arr1[i] == arr1[j])

m++;

// Update the most frequent item and its frequency if the current item's frequency is higher

if (mf < m) {

mf = m;

item = arr1[i];

}

}

// Reset the current item's frequency for the next iteration

m = 0;

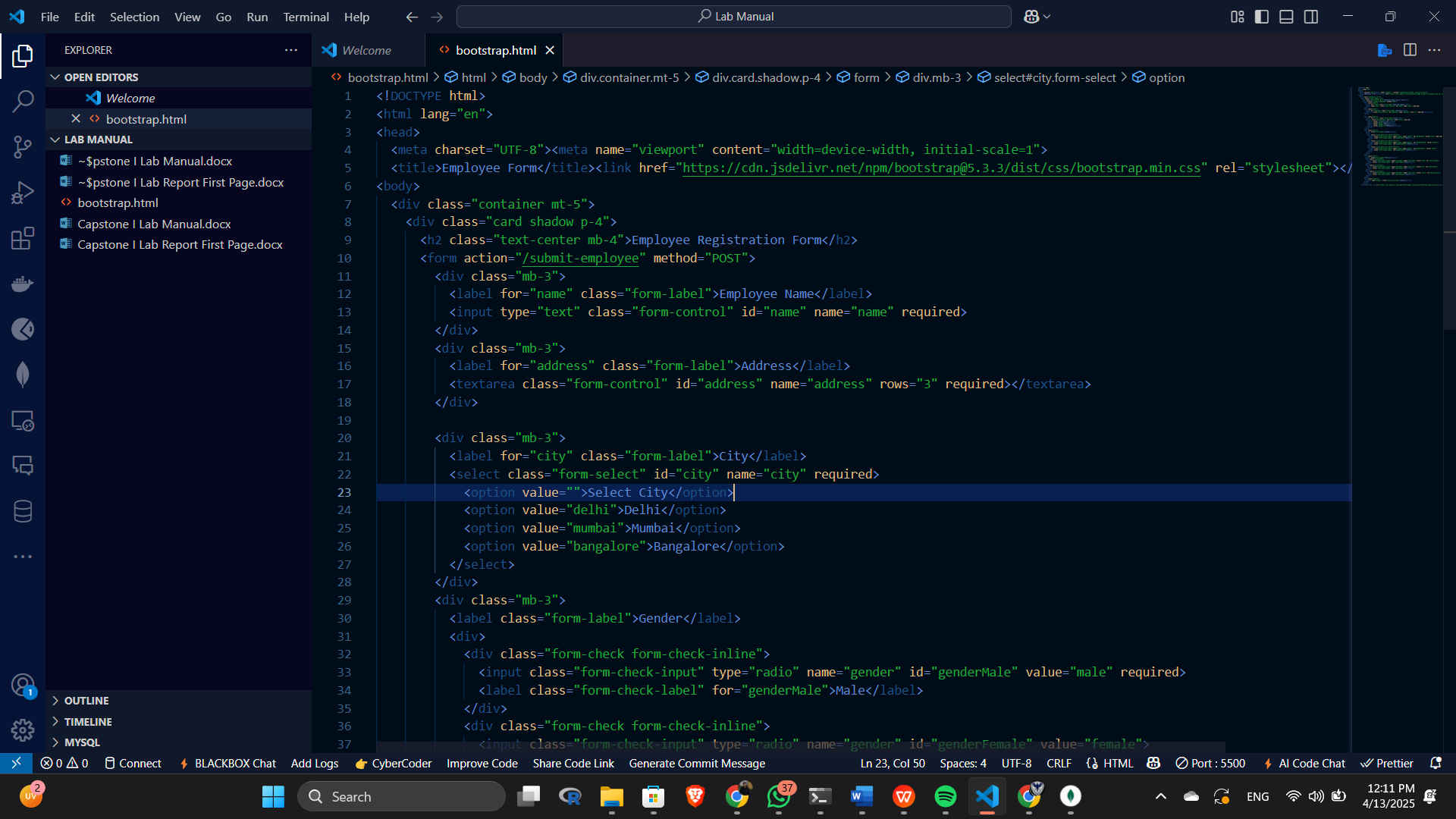
}

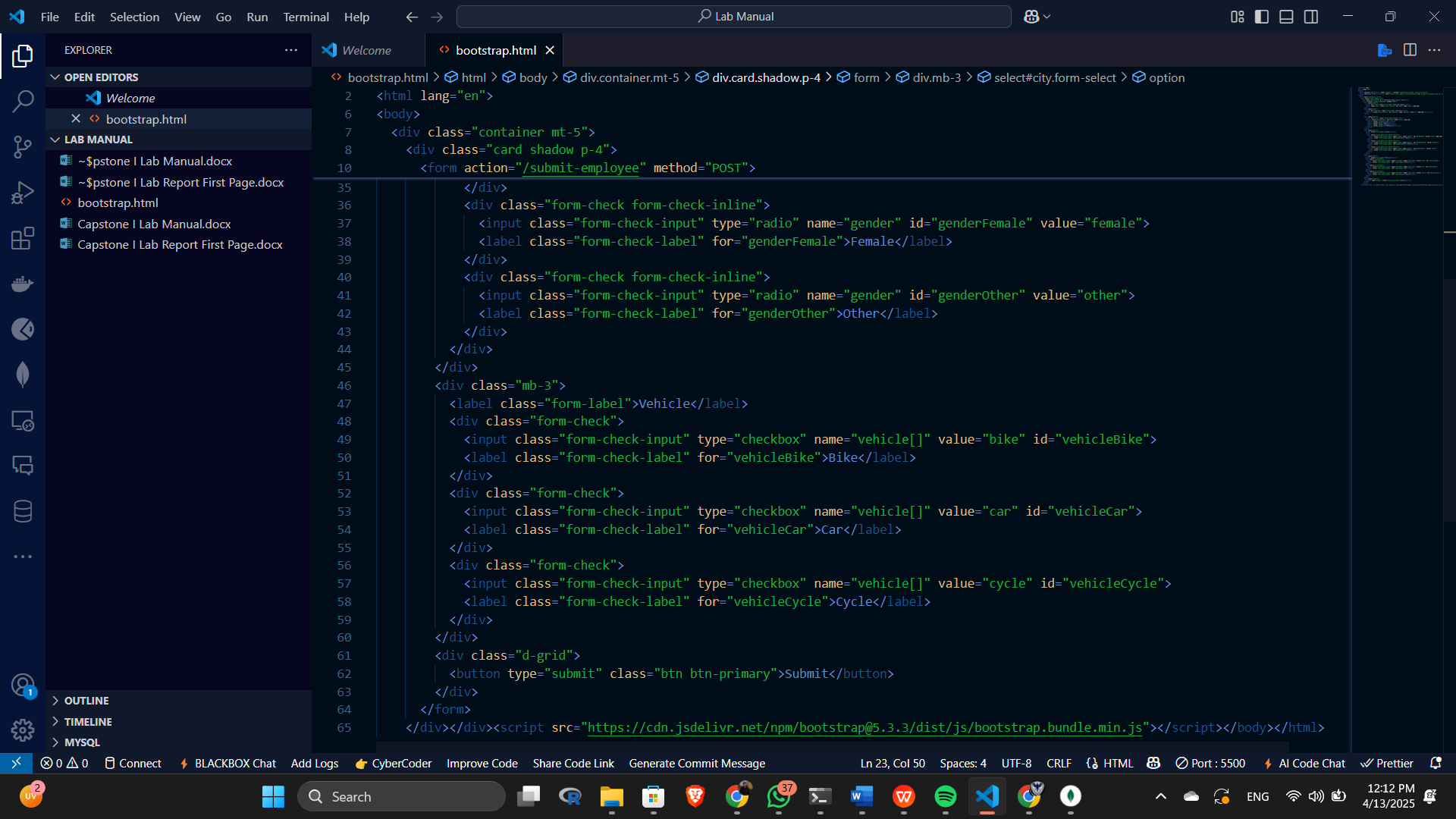
// Output the most frequent item and its frequency

console.log(item + " ( " + mf + " times ) ");

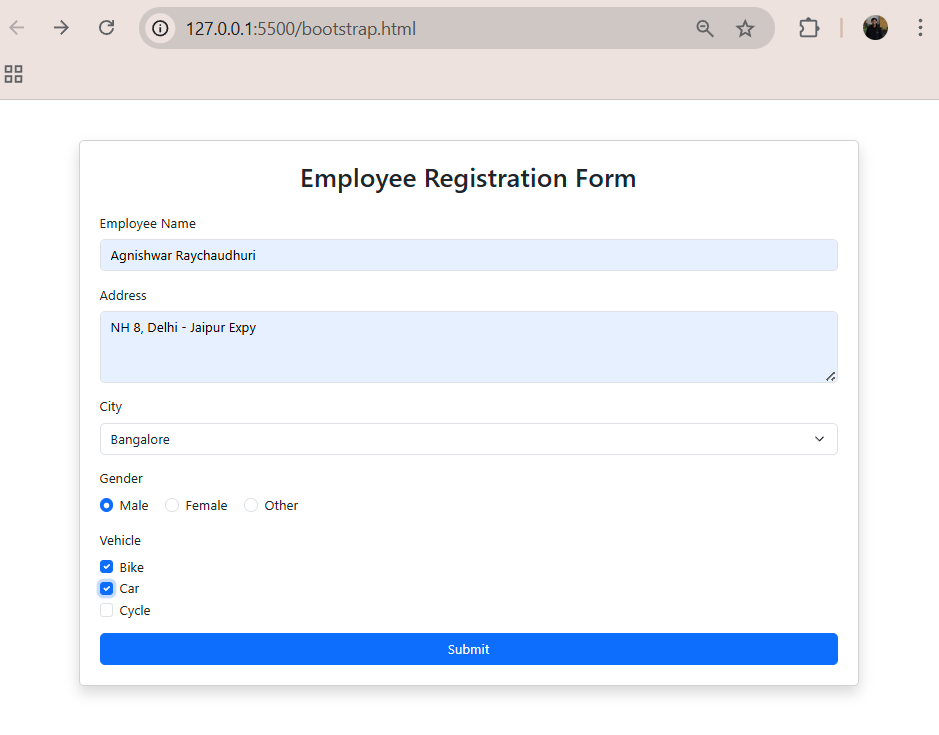
**Practical 4:** BootStrap :

**Code**





**Output**



**Practical 5:** NodeJS :

Write NodeJS code using HTTP module

Using NodeJS copy a file

Make use of URL module using NodeJS

Show event handling with NodeJS

Do CRUD operation using NodeJS with MySQL

**Practical 6:** ExpressJS :

Do routing with ExpressJS

Make use of GET, POST, PUT and DELETE HTTP methods with ExpressJS

Use middleware in ExpressJS

Create response using ejs in ExpressJS

Use static files in ExpressJS

Handle Form data in ExpressJS

**Practical 7:** MongoDB :

**Structure of 'restaurants' collection:**

{

"address": {

"building": "1007",

"coord": [ -73.856077, 40.848447 ],

"street": "Morris Park Ave",

"zipcode": "10462"

},

"borough": "Bronx",

"cuisine": "Bakery",

"grades": [

{ "date": { "$date": 1393804800000 }, "grade": "A", "score": 2 },

{ "date": { "$date": 1378857600000 }, "grade": "A", "score": 6 },

{ "date": { "$date": 1358985600000 }, "grade": "A", "score": 10 },

{ "date": { "$date": 1322006400000 }, "grade": "A", "score": 9 },

{ "date": { "$date": 1299715200000 }, "grade": "B", "score": 14 }

],

"name": "Morris Park Bake Shop",

"restaurant\_id": "30075445"

}

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

IMG_256

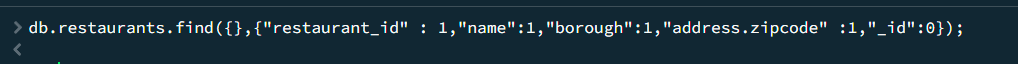
2. Write a MongoDB query to display the fields restaurant\_id, name, borough and cuisine for all the documents in the collection restaurant.

IMG_257

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.

IMG_258

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.



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

IMG_260

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

IMG_261

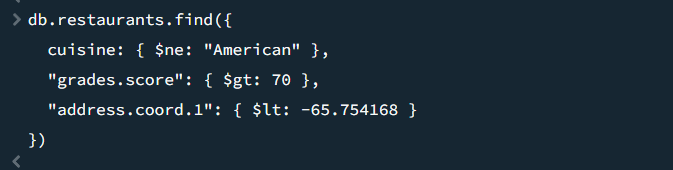
7.Write a MongoDB query to display the next 5 restaurants after skipping first 5 which are in the borough Bronx.

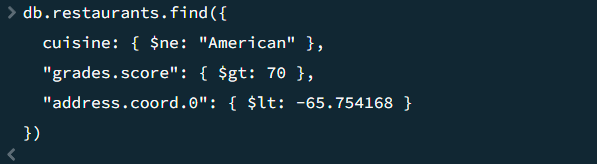
IMG_262

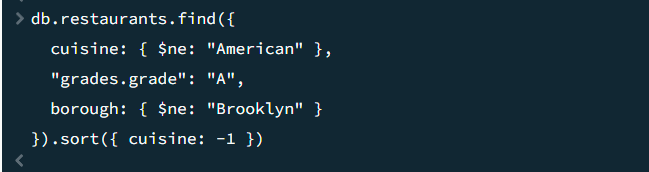
8. Write a MongoDB query to find the restaurants who achieved a score more than 90.   

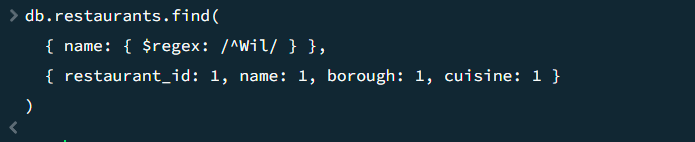

9. Write a MongoDB query to find the restaurants that achieved a score, more than 80 but less than 100.   

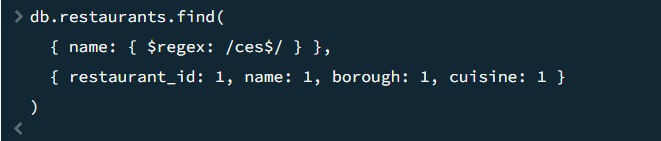

10. Write a MongoDB query to find the restaurants which locate in latitude value less than -95.754168.  

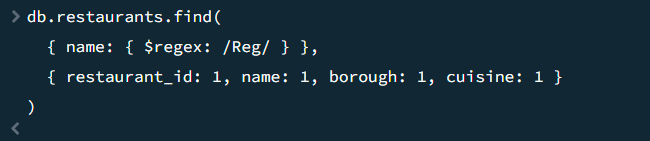

11. Write a MongoDB query to find the restaurants that do not prepare any cuisine of 'American' and their grade score more than 70 and latitude less than -65.754168.   


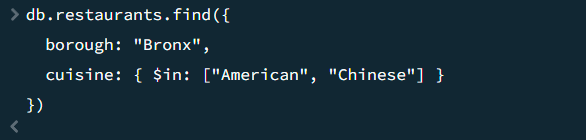
12. Write a MongoDB query to find the restaurants which do not prepare any cuisine of 'American' and achieved a score more than 70 and located in the longitude less than -65.754168.  
Note : Do this query without using $and operator.   


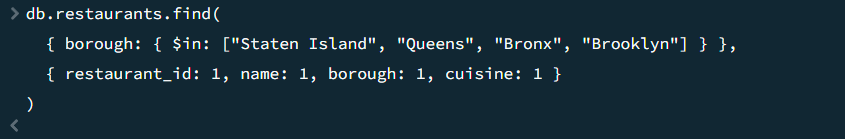
13. Write a MongoDB query to find the restaurants which do not prepare any cuisine of 'American' and achieved a grade point 'A' not belongs to the borough Brooklyn. The document must be displayed according to the cuisine in descending order.   


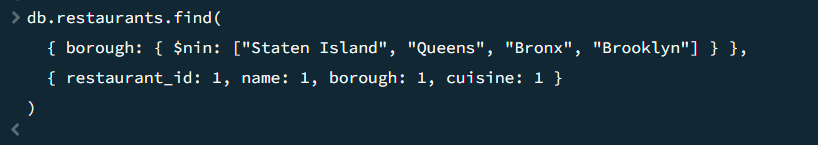
14. Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those restaurants which contain 'Wil' as first three letters for its name. 

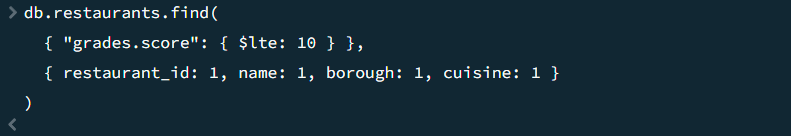
15. Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those restaurants which contain 'ces' as last three letters for its name.   


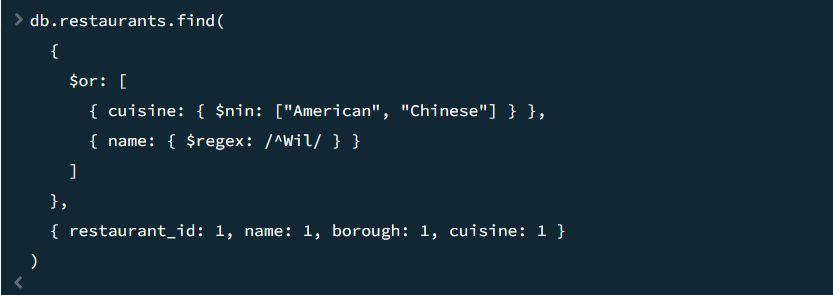
16. Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those restaurants which contain 'Reg' as three letters somewhere in its name.   


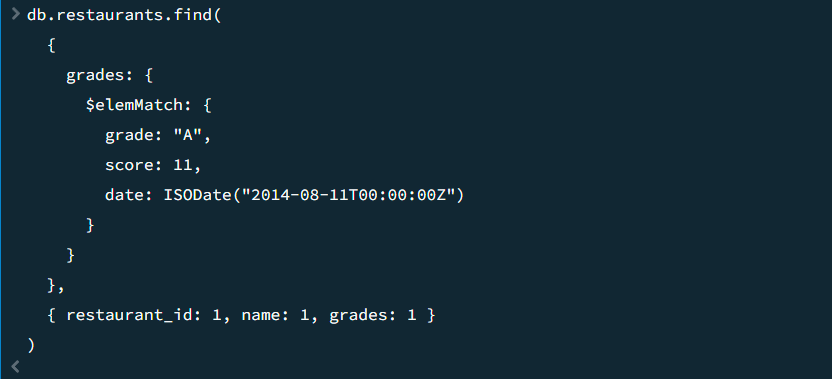
17. Write a MongoDB query to find the restaurants which belong to the borough Bronx and prepared either American or Chinese dish.   


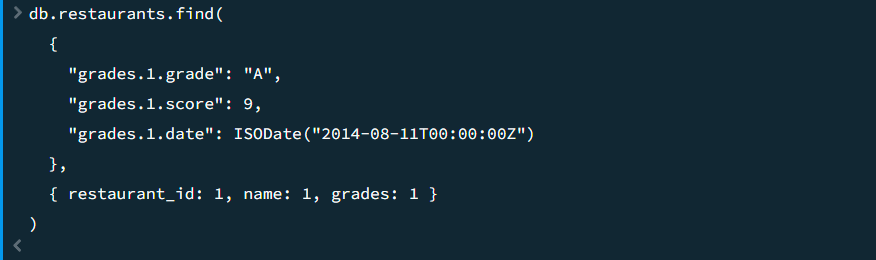
18. Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those restaurants which belong to the borough Staten Island or Queens or Bronxor Brooklyn.   


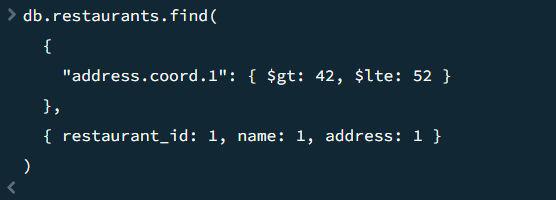
19. Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those restaurants which are not belonging to the borough Staten Island or Queens or Bronxor Brooklyn.   


20. Write a MongoDB query to find the restaurant Id, name, borough and cuisine for those restaurants which achieved a score which is not more than 10.   


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


22. Write a MongoDB query to find the restaurant Id, name, and grades for those restaurants which achieved a grade of "A" and scored 11 on an ISODate "2014-08-11T00:00:00Z" among many of survey dates..   


23. Write a MongoDB query to find the restaurant Id, name and grades for those restaurants where the 2nd element of grades array contains a grade of "A" and score 9 on an ISODate "2014-08-11T00:00:00Z".   


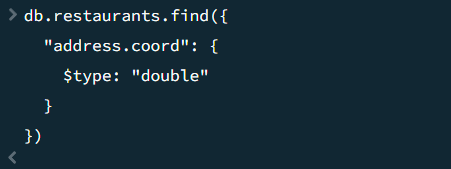
24. Write a MongoDB query to find the restaurant Id, name, address and geographical location for those restaurants where 2nd element of coord array contains a value which is more than 42 and upto 52..   


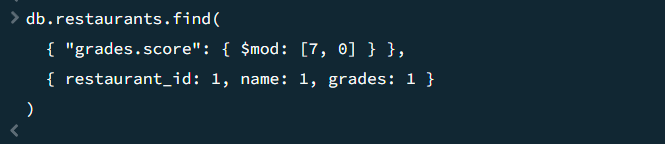
25. Write a MongoDB query to arrange the name of the restaurants in ascending order along with all the columns.   

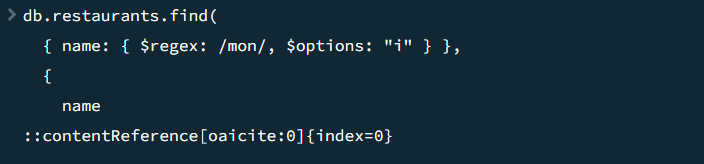

26. Write a MongoDB query to arrange the name of the restaurants in descending along with all the columns.   

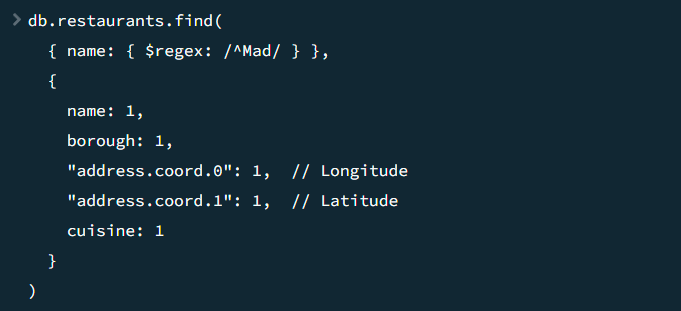

27. Write a MongoDB query to arranged the name of the cuisine in ascending order and for that same cuisine borough should be in descending order.   

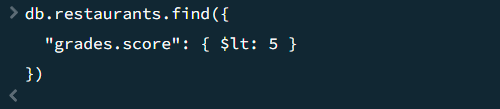

28. Write a MongoDB query to know whether all the addresses contains the street or not.   

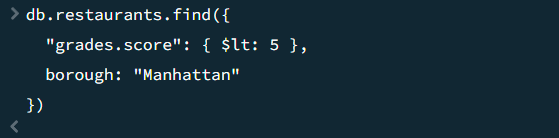

29. Write a MongoDB query which will select all documents in the restaurants collection where the coord field value is Double.   


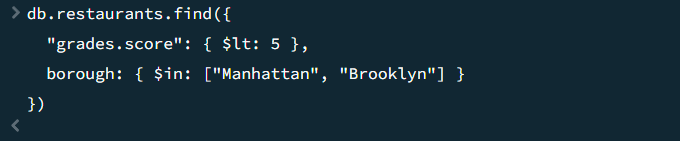
30. Write a MongoDB query which will select the restaurant Id, name and grades for those restaurants which returns 0 as a remainder after dividing the score by 7.   


31. Write a MongoDB query to find the restaurant name, borough, longitude and attitude and cuisine for those restaurants which contains 'mon' as three letters somewhere in its name.   


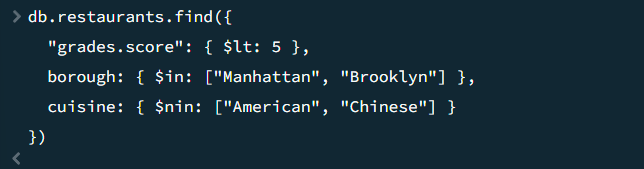
32. Write a MongoDB query to find the restaurant name, borough, longitude and latitude and cuisine for those restaurants which contain 'Mad' as first three letters of its name.   


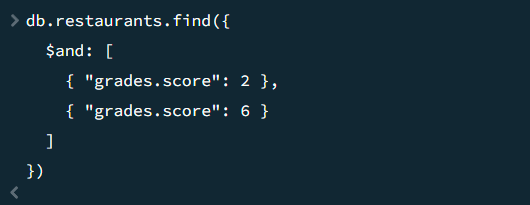
33. Write a MongoDB query to find the restaurants that have at least one grade with a score of less than 5.   


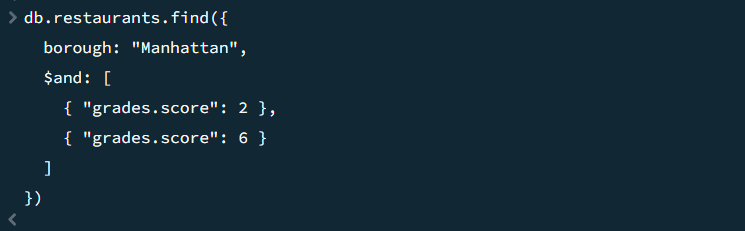
34. Write a MongoDB query to find the restaurants that have at least one grade with a score of less than 5 and that are located in the borough of Manhattan.   


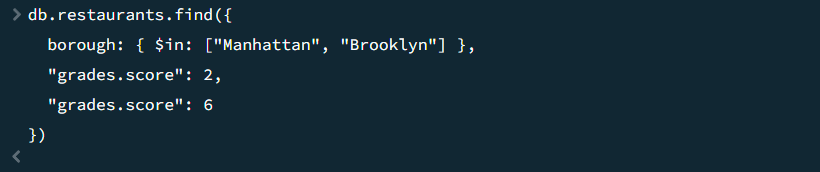
35. Write a MongoDB query to find the restaurants that have at least one grade with a score of less than 5 and that are located in the borough of Manhattan or Brooklyn.   


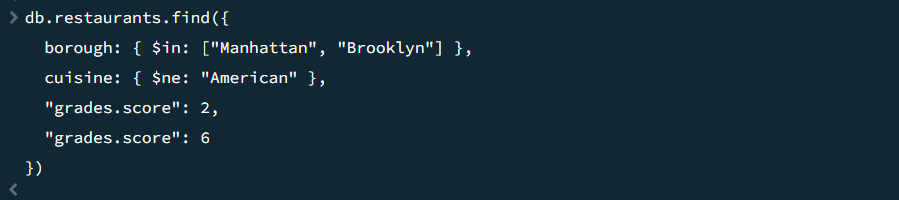
36. Write a MongoDB query to find the restaurants that have at least one grade with a score of less than 5 and that are located in the borough of Manhattan or Brooklyn, and their cuisine is not American.   

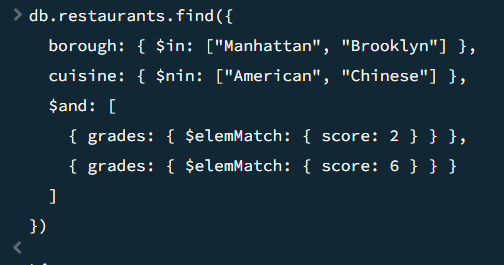

37. Write a MongoDB query to find the restaurants that have at least one grade with a score of less than 5 and that are located in the borough of Manhattan or Brooklyn, and their cuisine is not American or Chinese.   


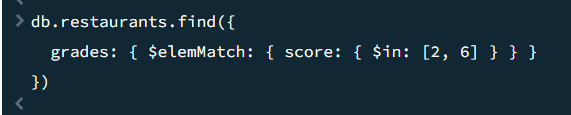
38. Write a MongoDB query to find the restaurants that have a grade with a score of 2 and a grade with a score of 6.   


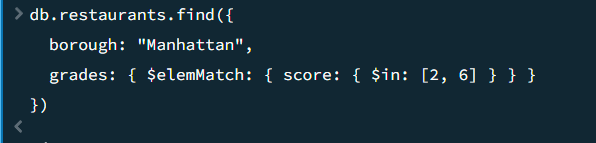
39. Write a MongoDB query to find the restaurants that have a grade with a score of 2 and a grade with a score of 6 and are located in the borough of Manhattan.   


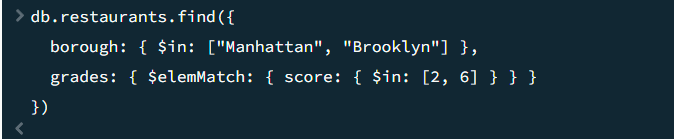
40. Write a MongoDB query to find the restaurants that have a grade with a score of 2 and a grade with a score of 6 and are located in the borough of Manhattan or Brooklyn.   


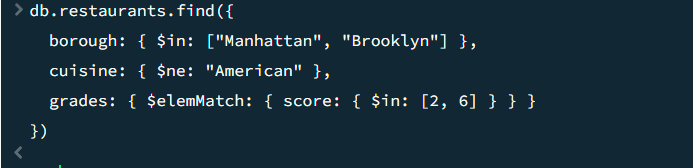
41. Write a MongoDB query to find the restaurants that have a grade with a score of 2 and a grade with a score of 6 and are located in the borough of Manhattan or Brooklyn, and their cuisine is not American.   


42. Write a MongoDB query to find the restaurants that have a grade with a score of 2 and a grade with a score of 6 and are located in the borough of Manhattan or Brooklyn, and their cuisine is not American or Chinese.   
 

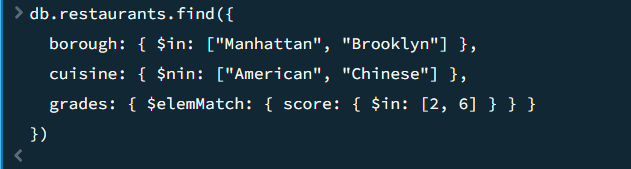
43. Write a MongoDB query to find the restaurants that have a grade with a score of 2 or a grade with a score of 6.   
 

44. Write a MongoDB query to find the restaurants that have a grade with a score of 2 or a grade with a score of 6 and are located in the borough of Manhattan.   


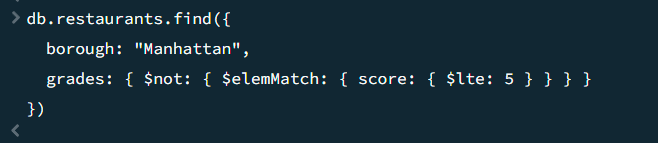
45. Write a MongoDB query to find the restaurants that have a grade with a score of 2 or a grade with a score of 6 and are located in the borough of Manhattan or Brooklyn.   


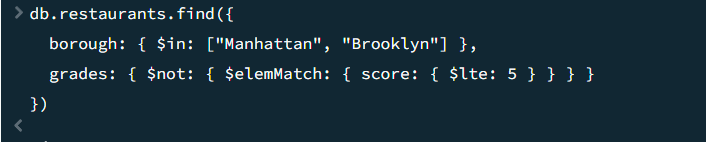
46. Write a MongoDB query to find the restaurants that have a grade with a score of 2 or a grade with a score of 6 and are located in the borough of Manhattan or Brooklyn, and their cuisine is not American. 

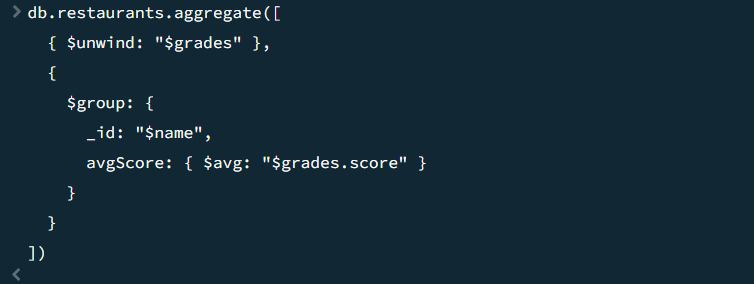
47. Write a MongoDB query to find the restaurants that have a grade with a score of 2 or a grade with a score of 6 and are located in the borough of Manhattan or Brooklyn, and their cuisine is not American or Chinese.

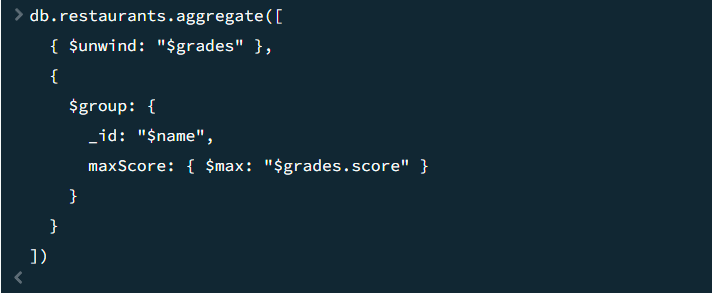
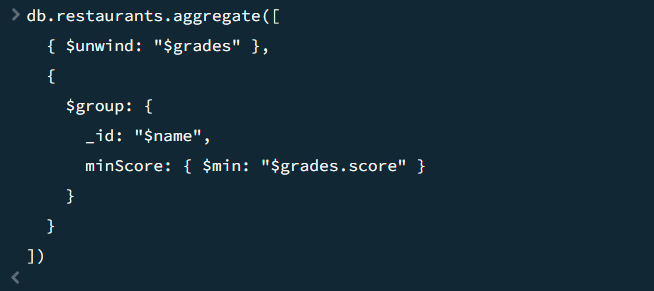


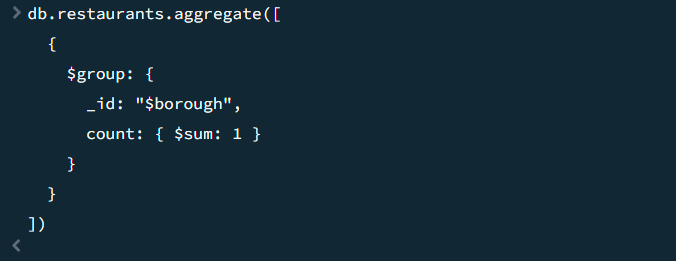
48. Write a MongoDB query to find the restaurants that have all grades with a score greater than 5.   

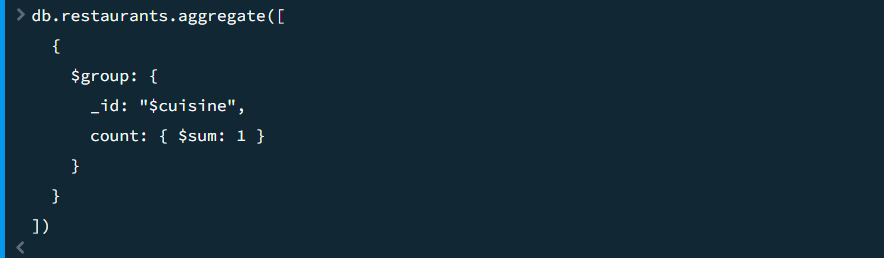

49. Write a MongoDB query to find the restaurants that have all grades with a score greater than 5 and are located in the borough of Manhattan.   


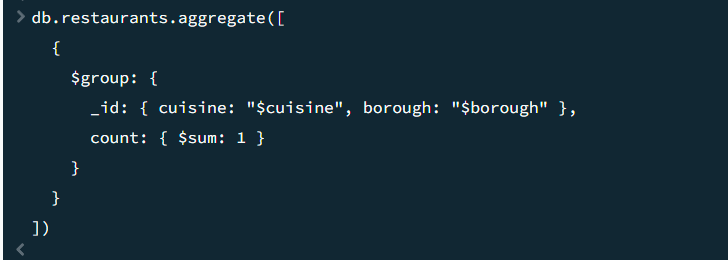
50. Write a MongoDB query to find the restaurants that have all grades with a score greater than 5 and are located in the borough of Manhattan or Brooklyn.

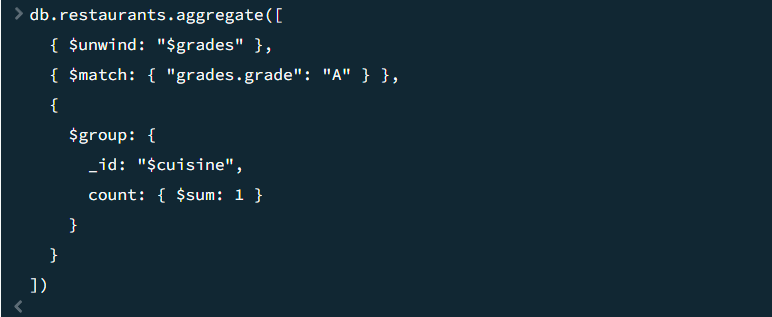
51. Write a MongoDB query to find the average score for each restaurant.   


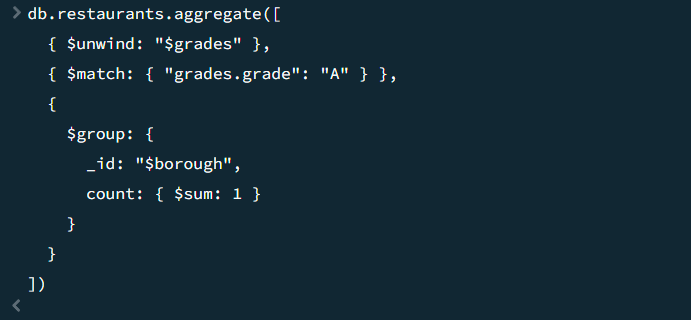
52. Write a MongoDB query to find the highest score for each restaurant.   
53. Write a MongoDB query to find the lowest score for each restaurant.   


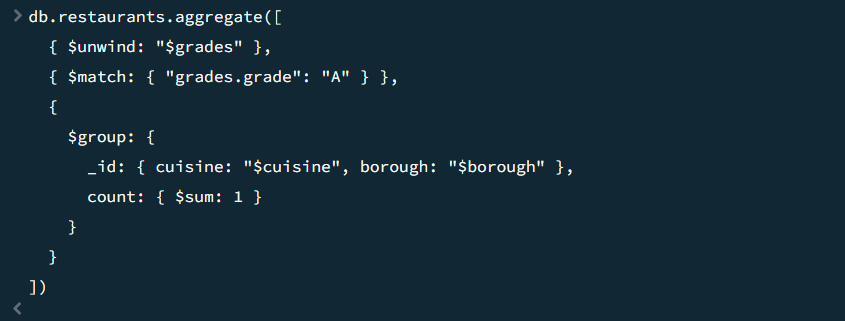
54. Write a MongoDB query to find the count of restaurants in each borough.   


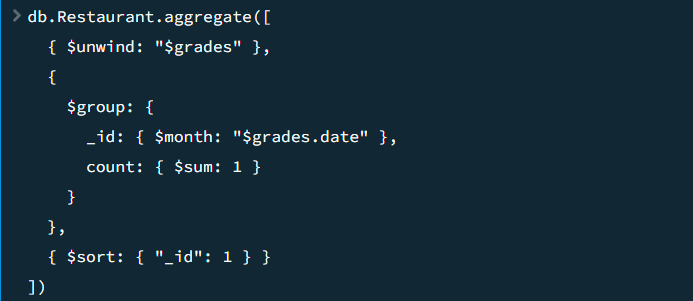
55. Write a MongoDB query to find the count of restaurants for each cuisine.   


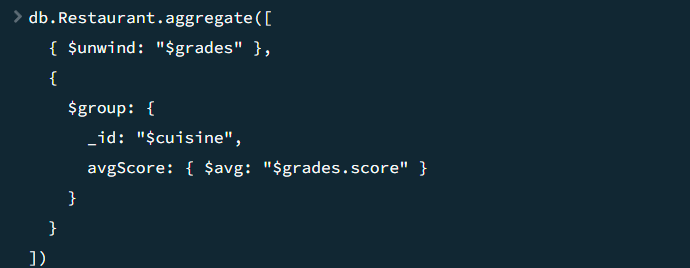
56. Write a MongoDB query to find the count of restaurants for each cuisine and borough.   


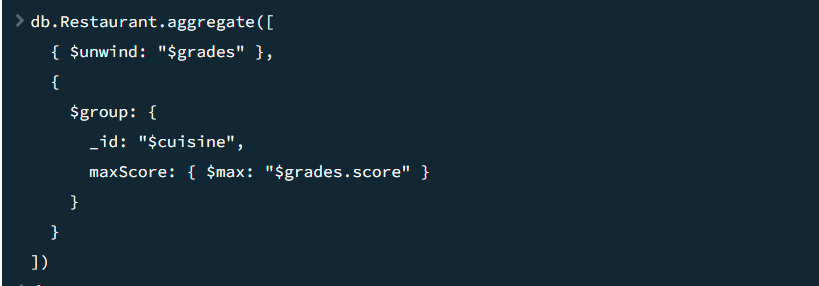
57. Write a MongoDB query to find the count of restaurants that received a grade of 'A' for each cuisine.   


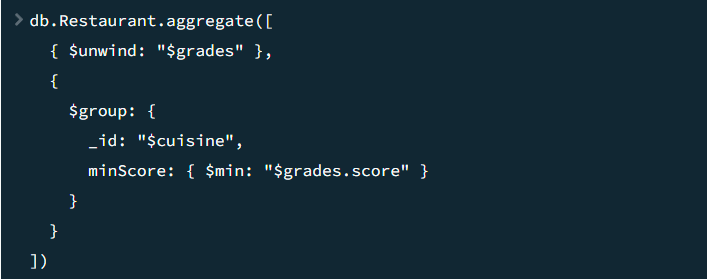
58. Write a MongoDB query to find the count of restaurants that received a grade of 'A' for each borough. 

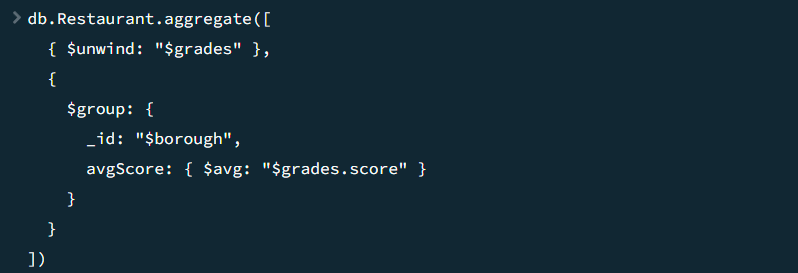
59. Write a MongoDB query to find the count of restaurants that received a grade of 'A' for each cuisine and borough.   


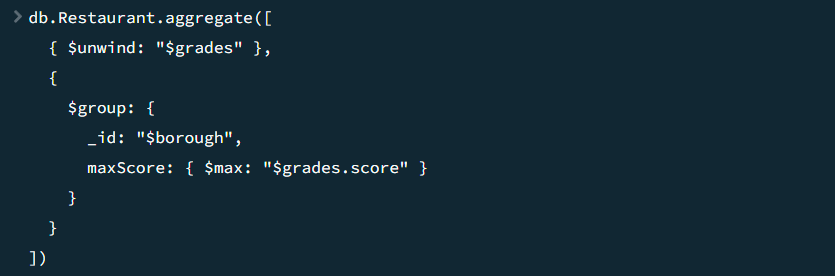
60. Write a MongoDB query to find the number of restaurants that have been graded in each month of the year.   


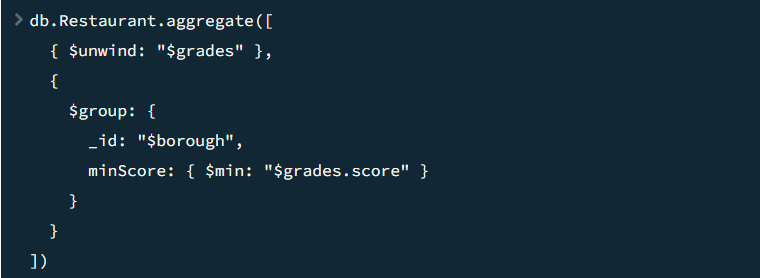
61. Write a MongoDB query to find the average score for each cuisine.   


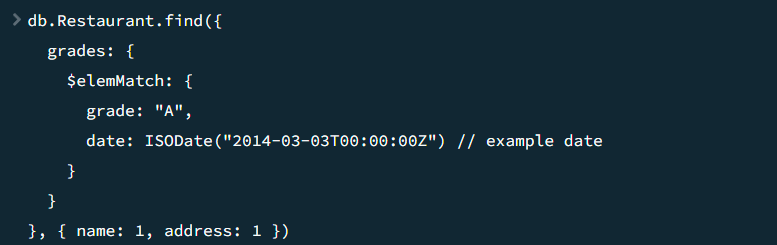
62. Write a MongoDB query to find the highest score for each cuisine. 

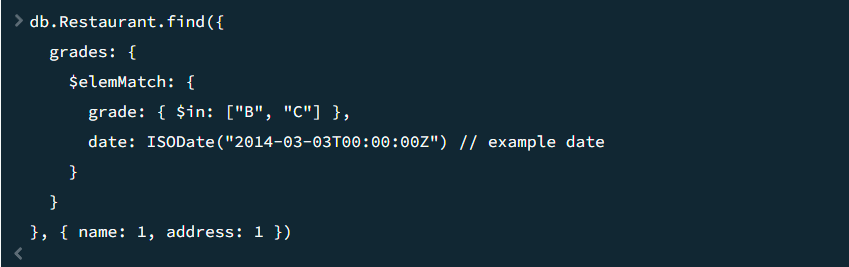
63. Write a MongoDB query to find the lowest score for each cuisine.   


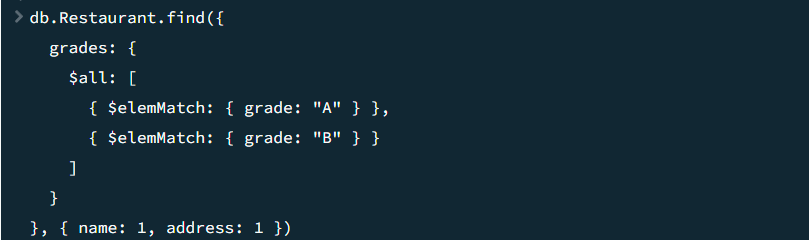
64. Write a MongoDB query to find the average score for each borough.   


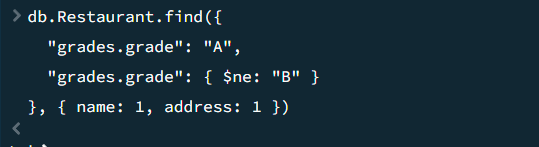
65. Write a MongoDB query to find the highest score for each borough. 

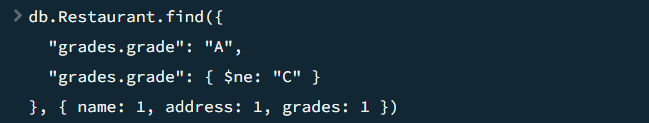
66. Write a MongoDB query to find the lowest score for each borough. 

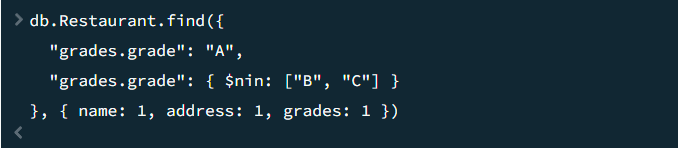
67. Write a MongoDB query to find the name and address of the restaurants that received a grade of 'A' on a specific date.   


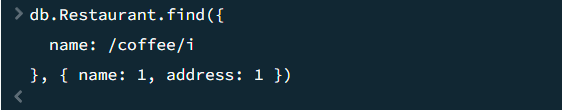
68. Write a MongoDB query to find the name and address of the restaurants that received a grade of 'B' or 'C' on a specific date. 

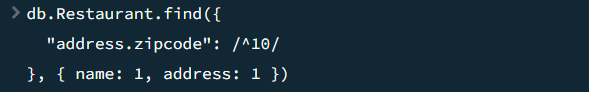
69. Write a MongoDB query to find the name and address of the restaurants that have at least one 'A' grade and one 'B' grade. 

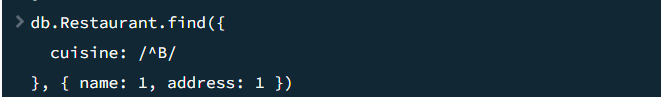
70. Write a MongoDB query to find the name and address of the restaurants that have at least one 'A' grade and no 'B' grades.   


71. Write a MongoDB query to find the name ,address and grades of the restaurants that have at least one 'A' grade and no 'C' grades. 

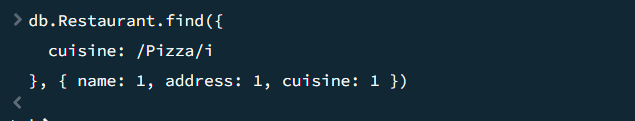
72. Write a MongoDB query to find the name, address, and grades of the restaurants that have at least one 'A' grade, no 'B' grades, and no 'C' grades. 

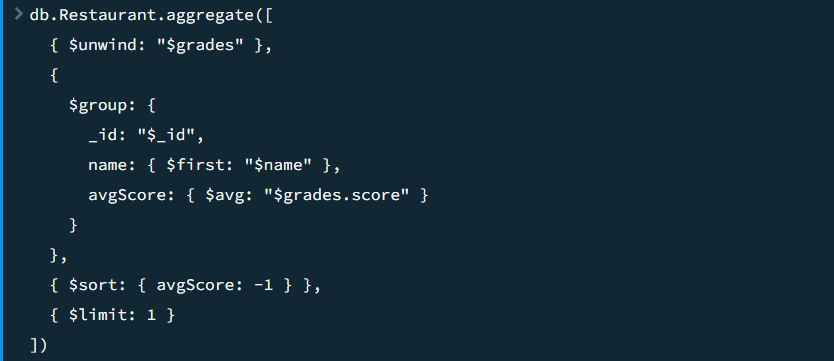
73. Write a MongoDB query to find the name and address of the restaurants that have the word 'coffee' in their name.   


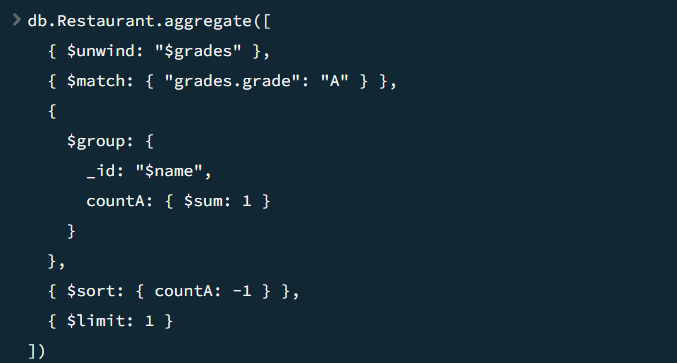
74. Write a MongoDB query to find the name and address of the restaurants that have a zipcode that starts with '10'.   


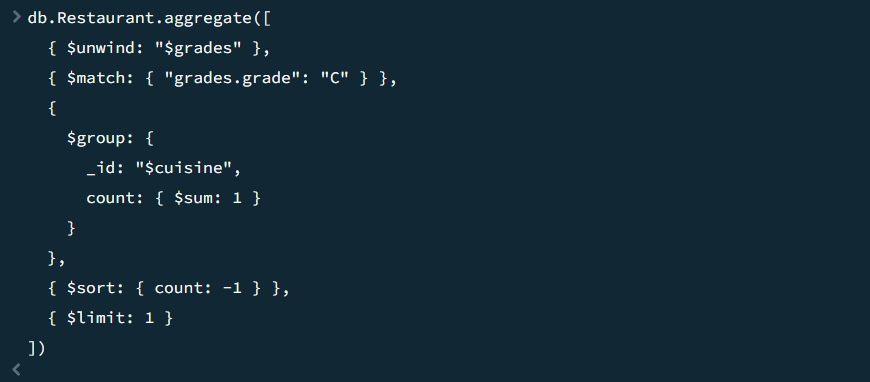
75. Write a MongoDB query to find the name and address of the restaurants that have a cuisine that starts with the letter 'B'.   


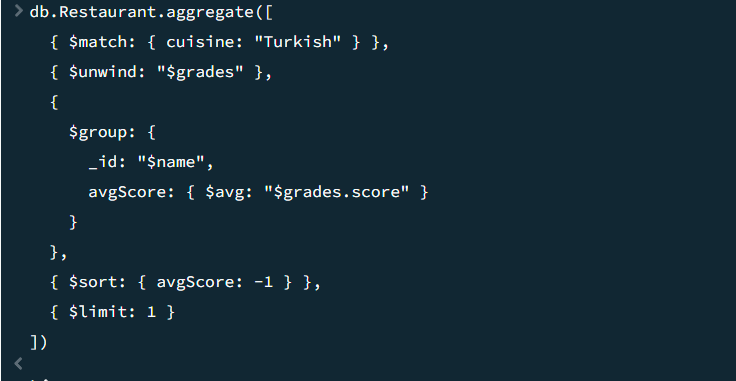
76. Write a MongoDB query to find the name, address, and cuisine of the restaurants that have a cuisine that ends with the letter 'y'.   


77. Write a MongoDB query to find the name, address, and cuisine of the restaurants that have a cuisine that contains the word 'Pizza'.   


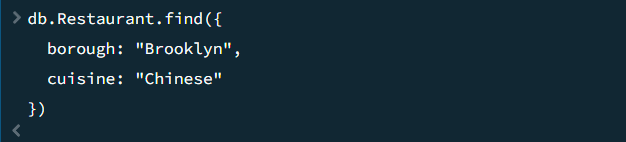
78. Write a MongoDB query to find the restaurants achieved highest average score.   


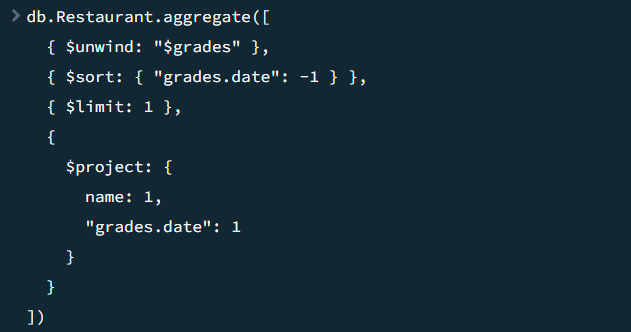
79. Write a MongoDB query to find all the restaurants with the highest number of "A" grades.   


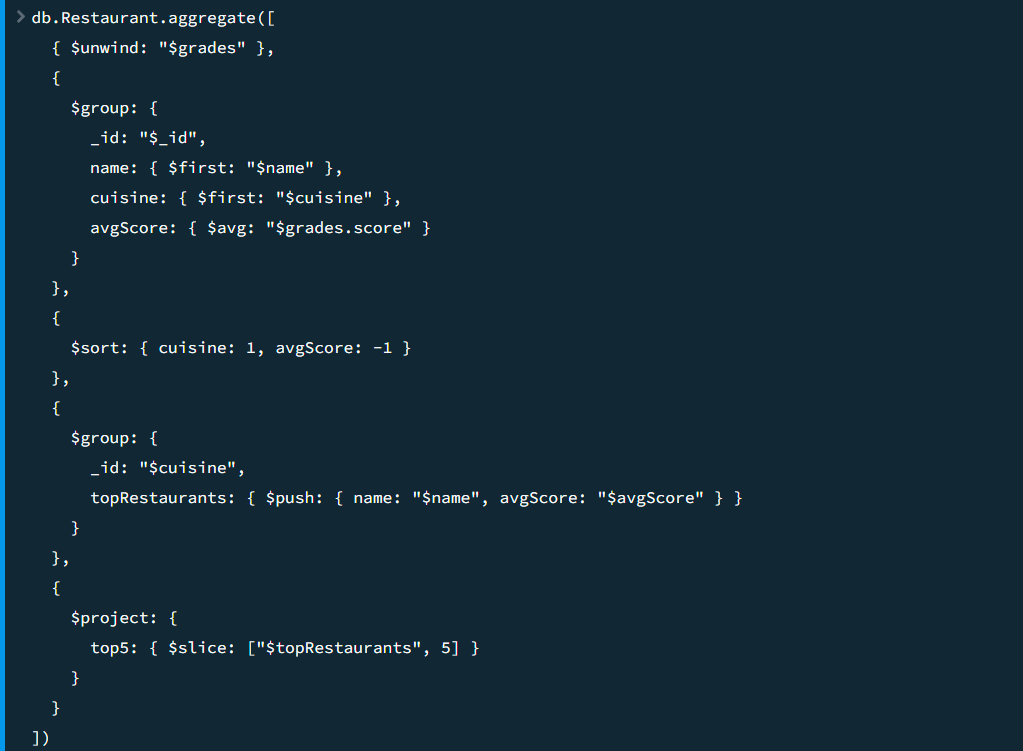
80. Write a MongoDB query to find the cuisine type that is most likely to receive a "C" grade.   


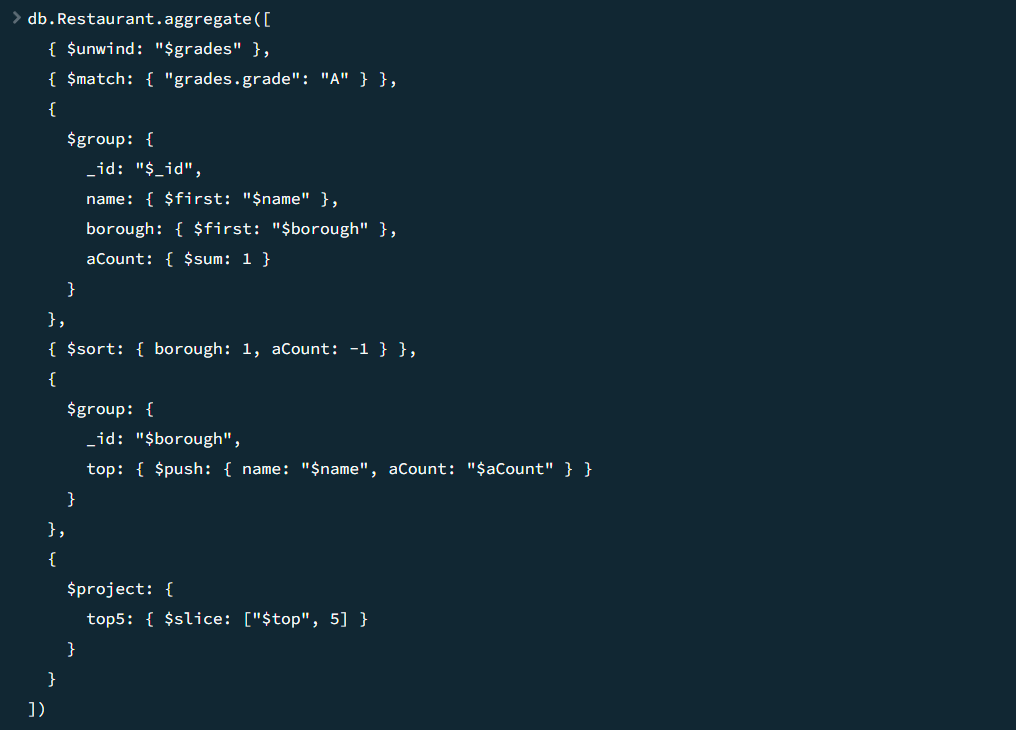
81. Write a MongoDB query to find the restaurant that has the highest average score for the cuisine "Turkish".   


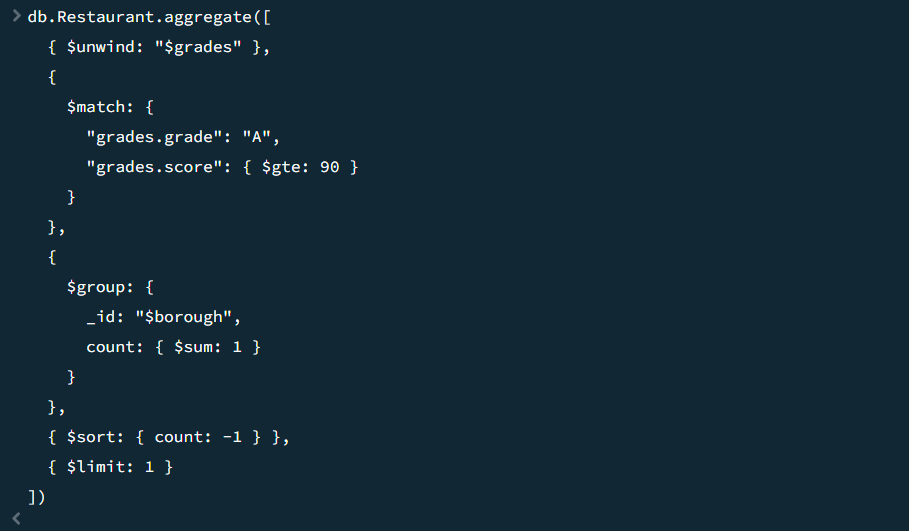
82. Write a MongoDB query to find the restaurants that achieved the highest total score.   


83. Write a MongoDB query to find all the Chinese restaurants in Brooklyn.   


84. Write a MongoDB query to find the restaurant with the most recent grade date.   


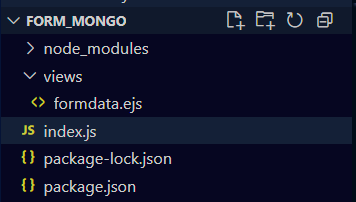
85. Write a MongoDB query to find the top 5 restaurants with the highest average score for each cuisine type, along with their average scores.   


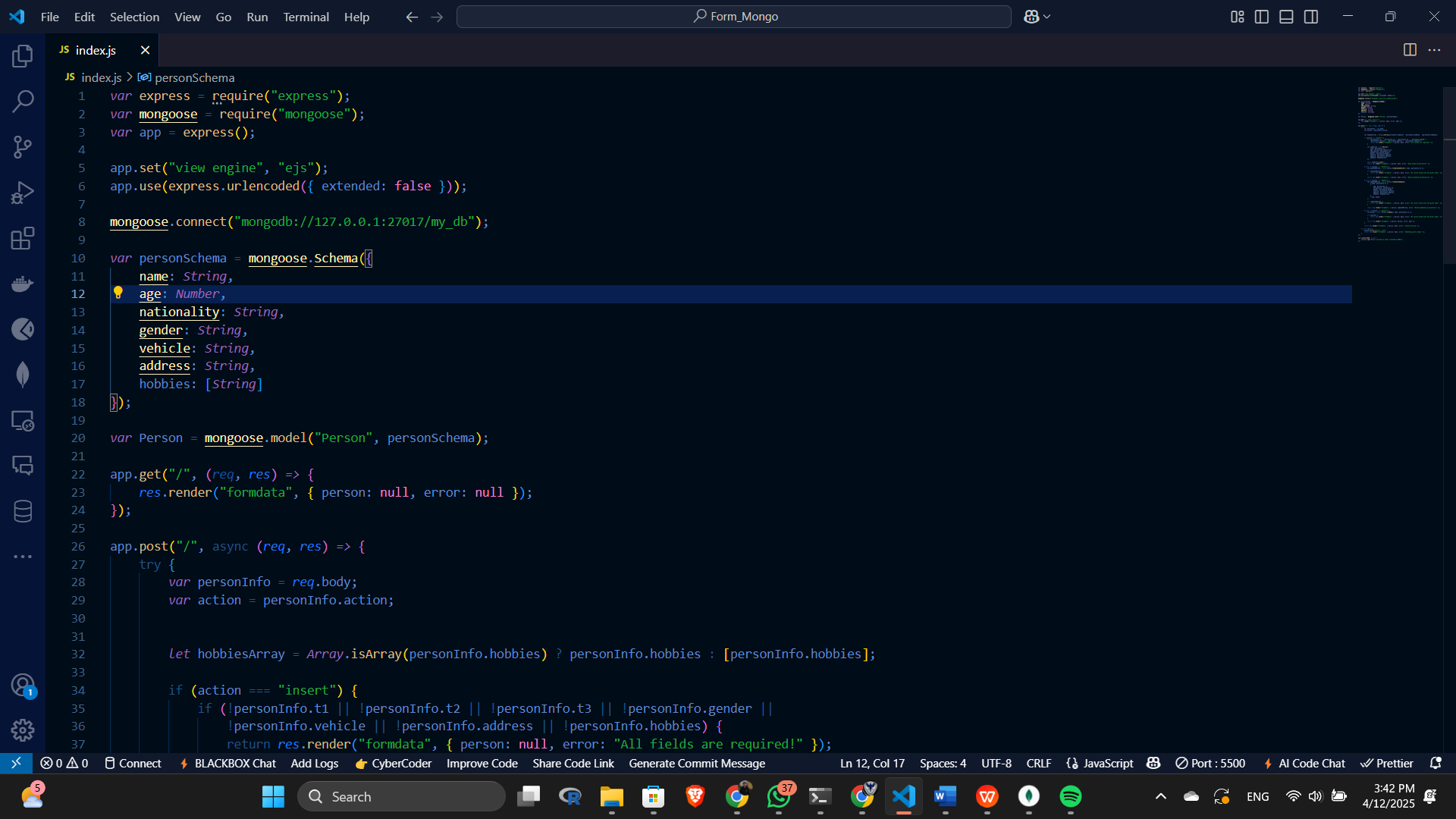
86. Write a MongoDB query to find the top 5 restaurants in each borough with the highest number of "A" grades.   


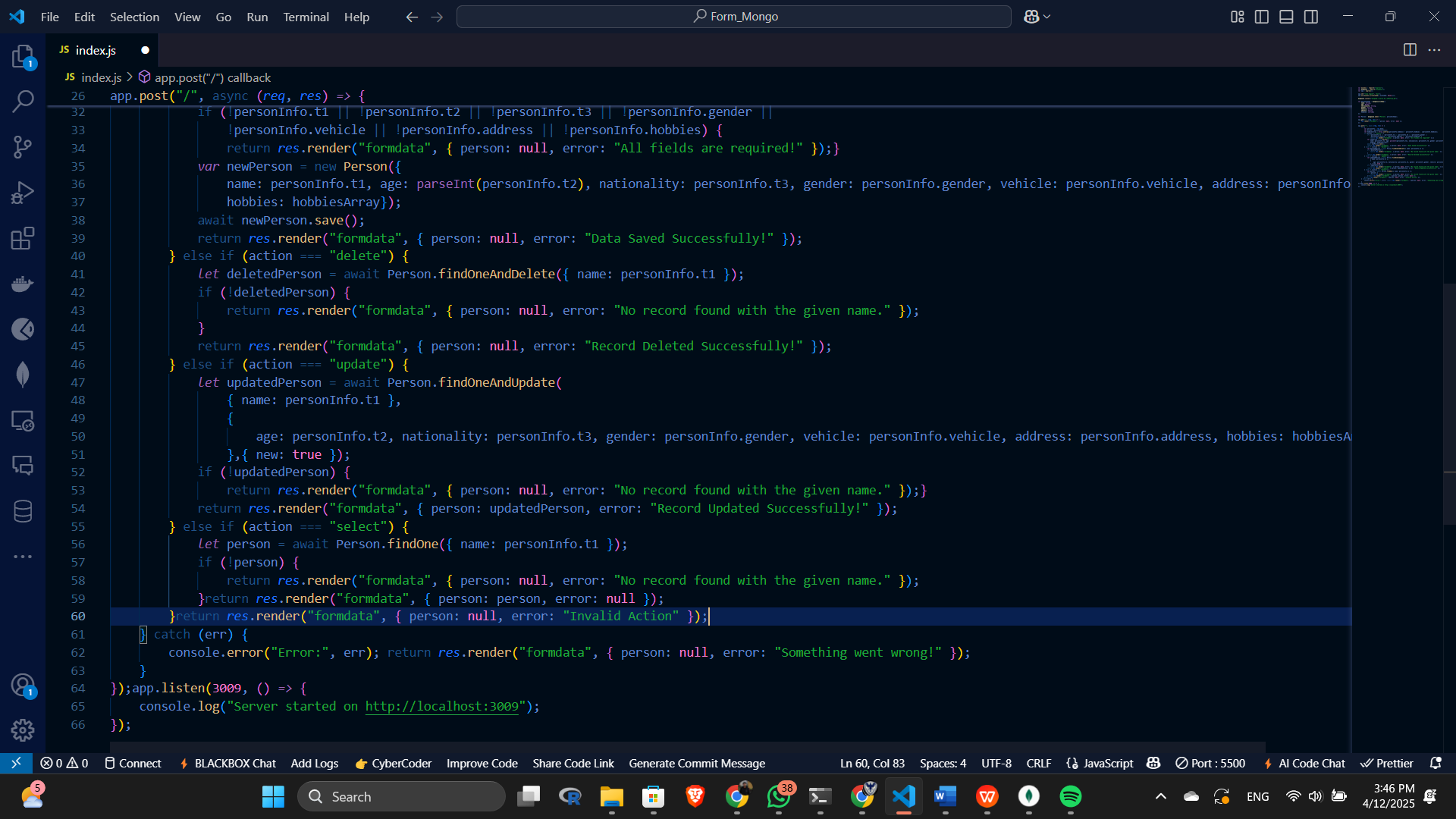
87. Write a MongoDB query to find the borough with the highest number of restaurants that have a grade of "A" and a score greater than or equal to 90.   


**Practical 8:** Database Connectivity:

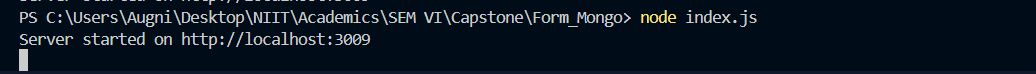
**Code**



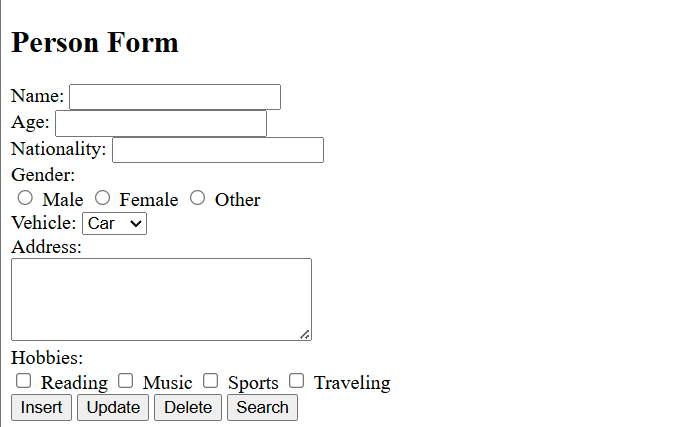




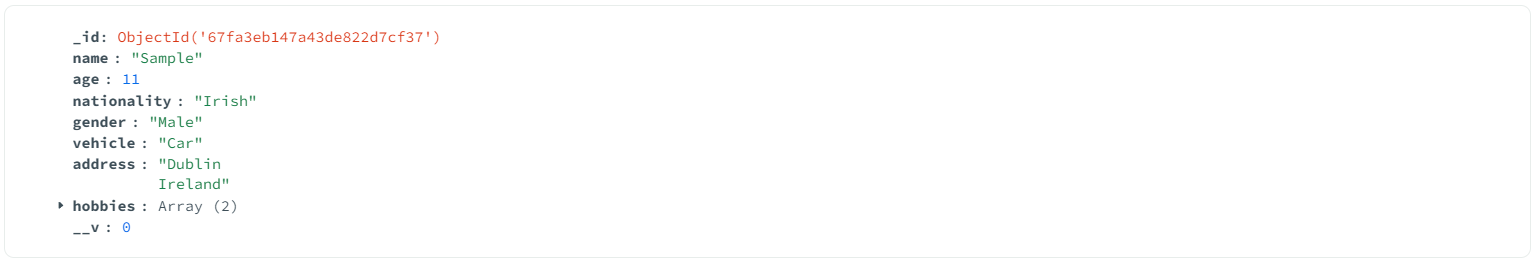
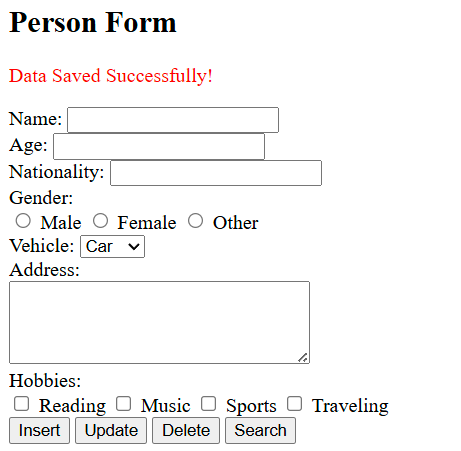
**Run the App**



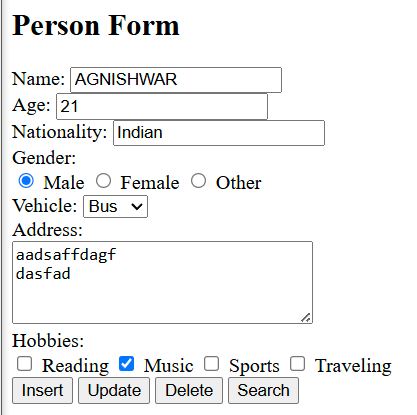
**CRUD Operations and Output**



Insert Operation



Search Operation



Update Operation



Delete Operation



**Practical 9:** Session and Cookies Demo :

Session management

Make use of cookies for storing client data

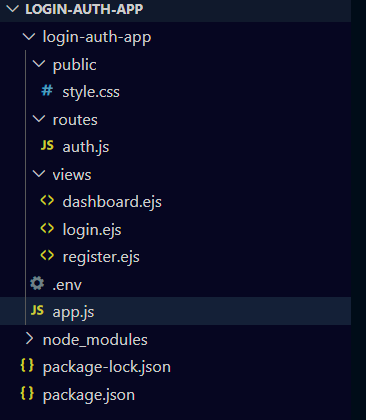
**Practical 10:** Login Exercise :

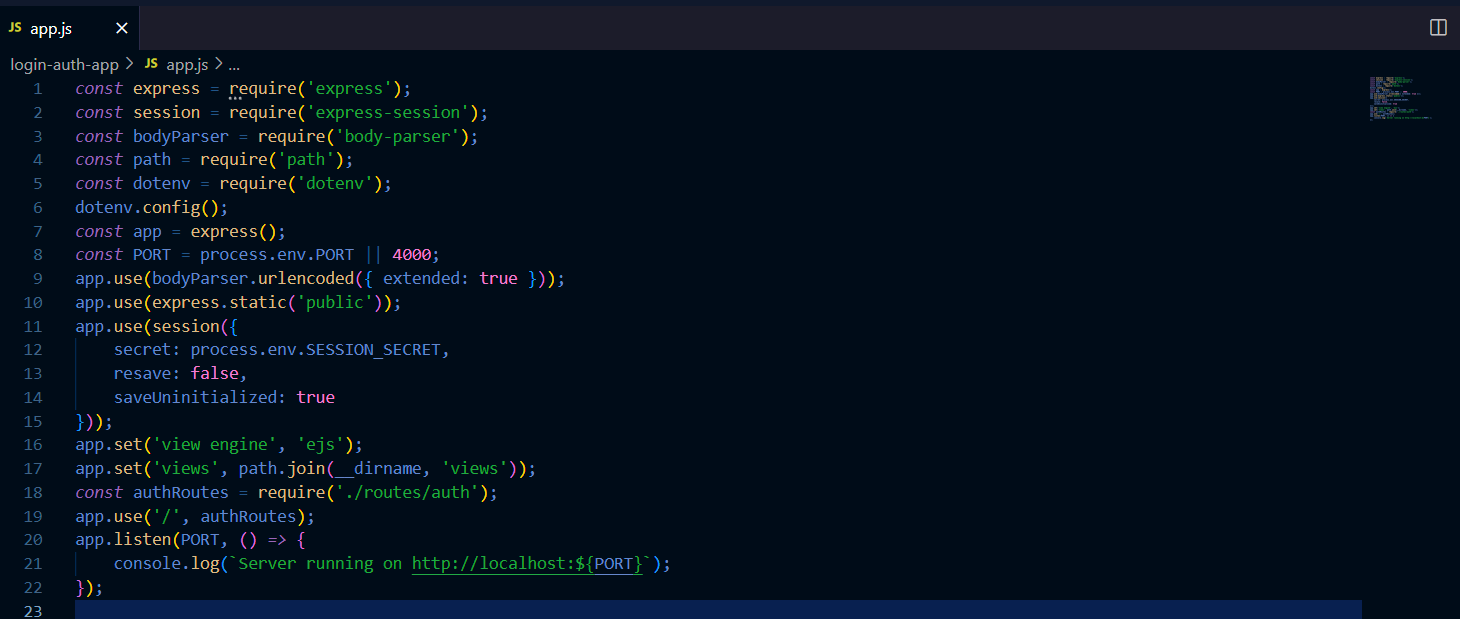
**Install dependencies**

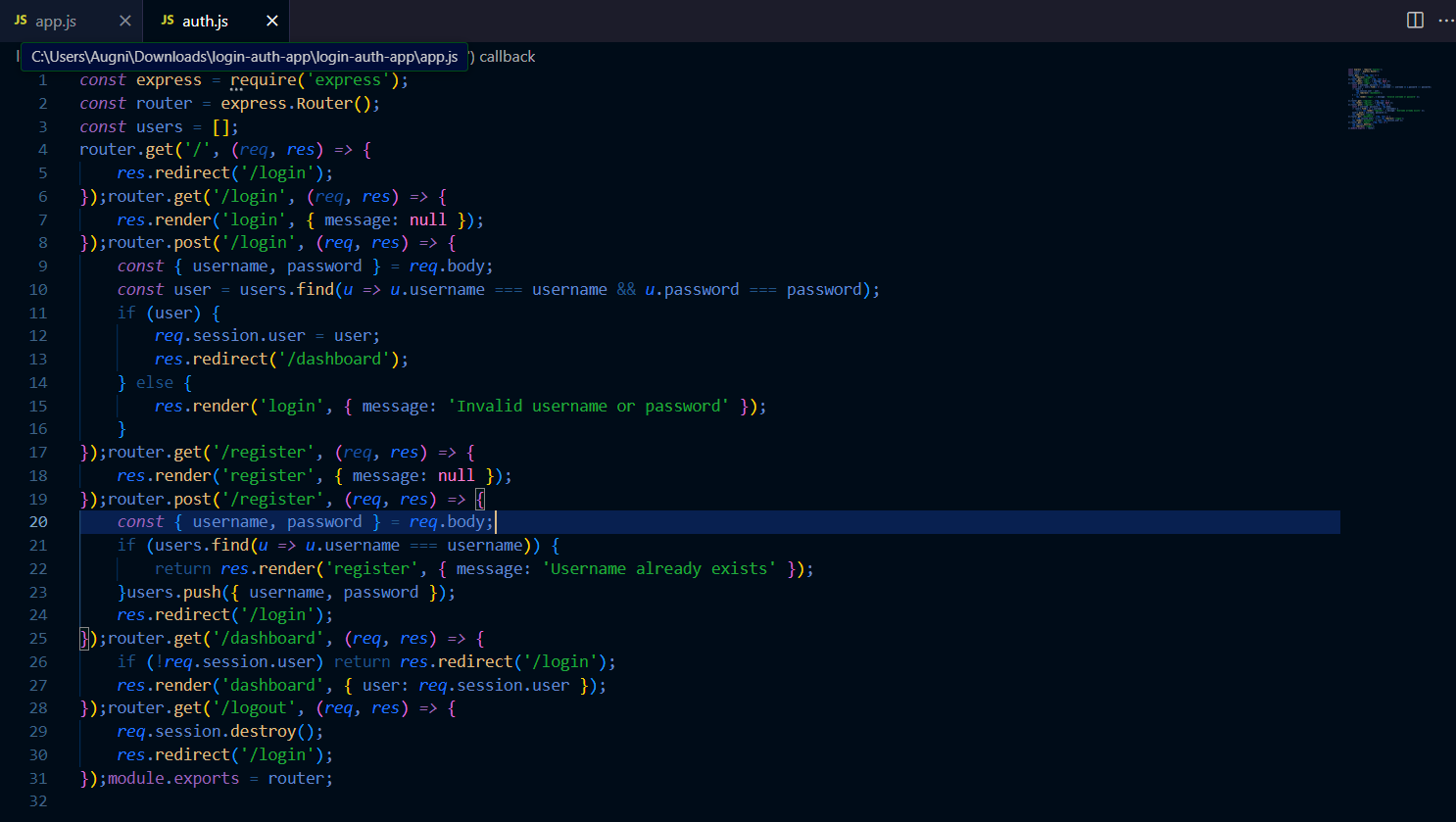




**Code**



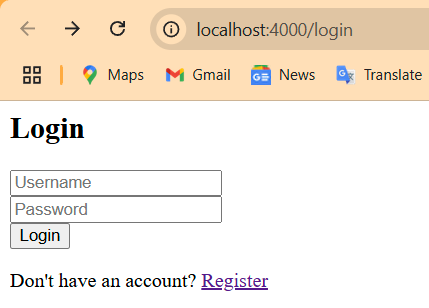


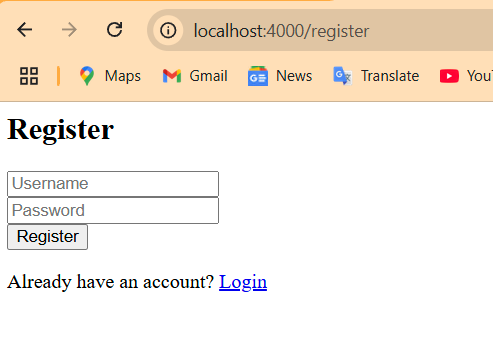


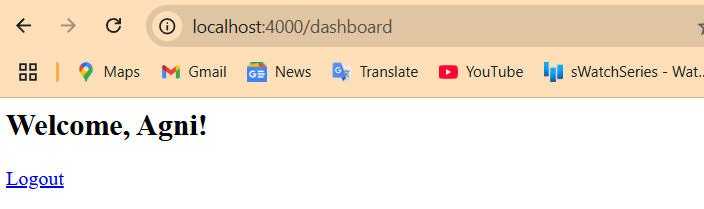
Run the app



**Output**

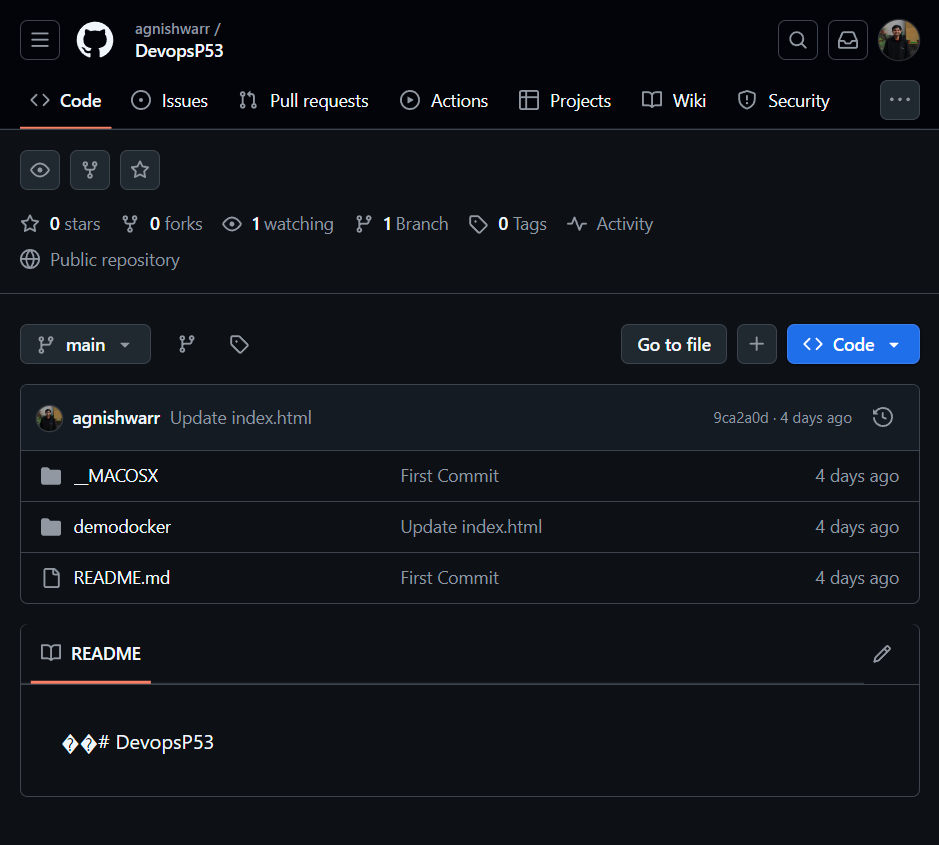




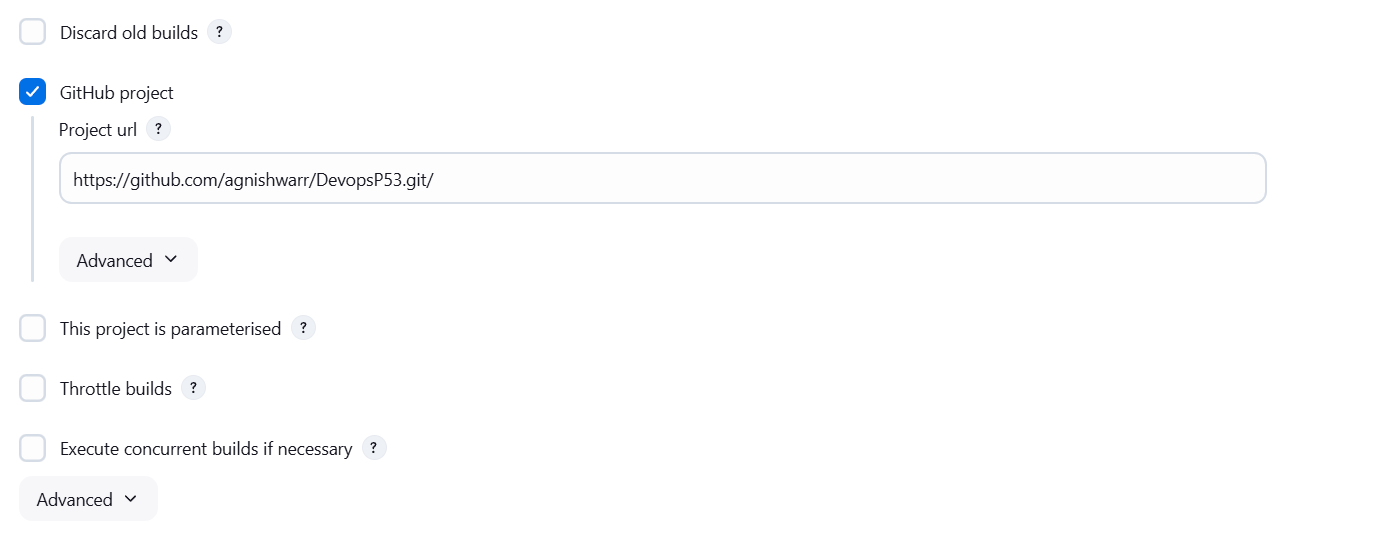


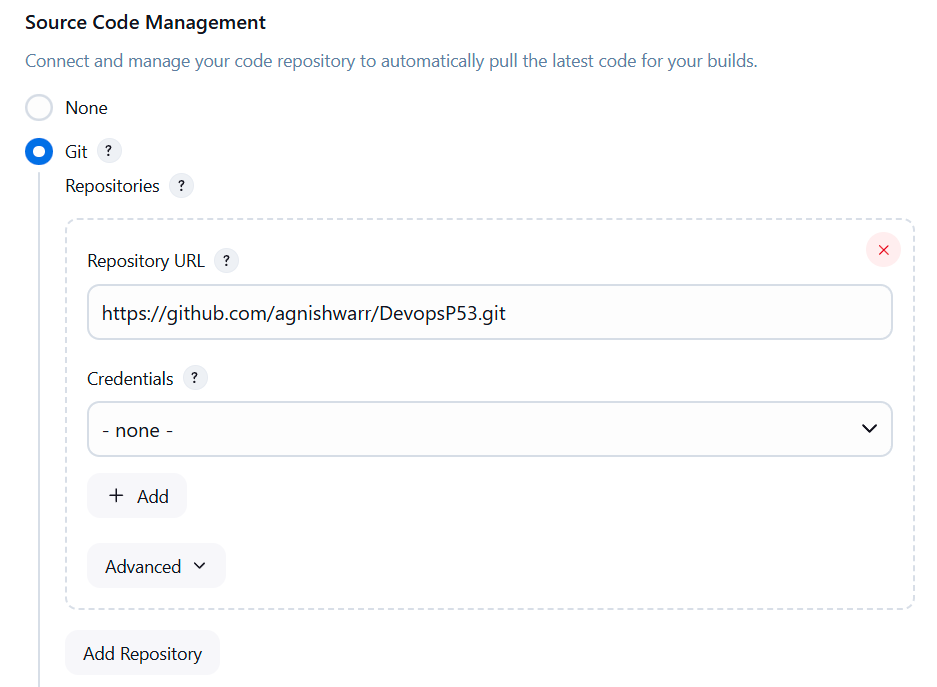
**Practical 10:** Setting up DevOPS for Project :

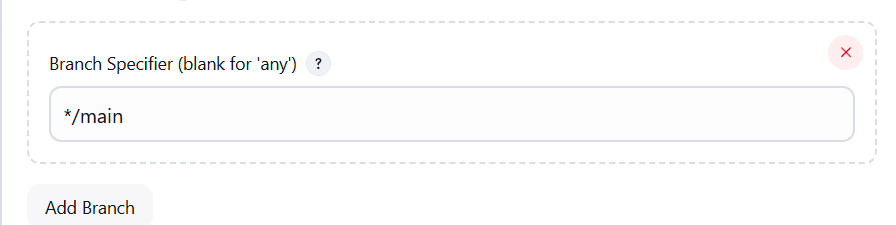
**GitHub Setup**

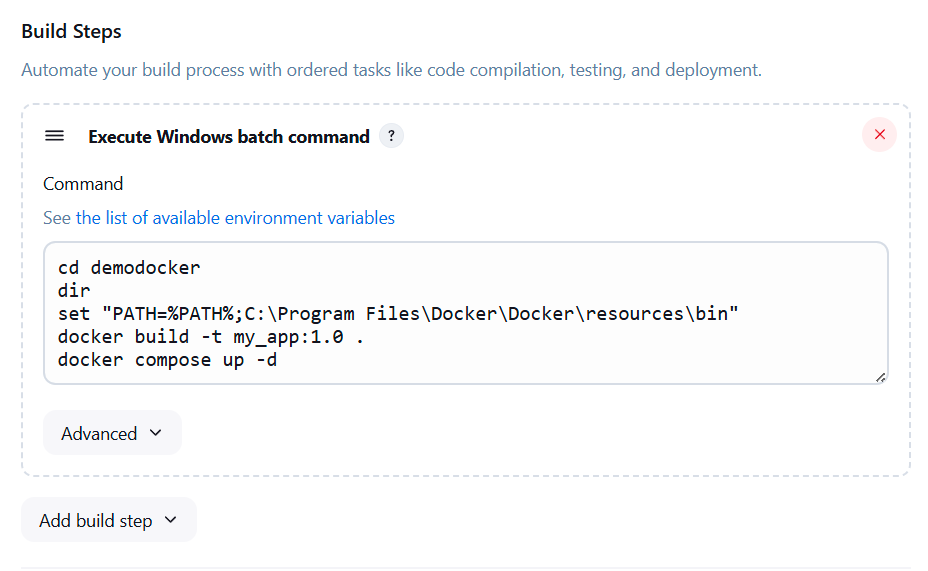


**Jenkins Setup**

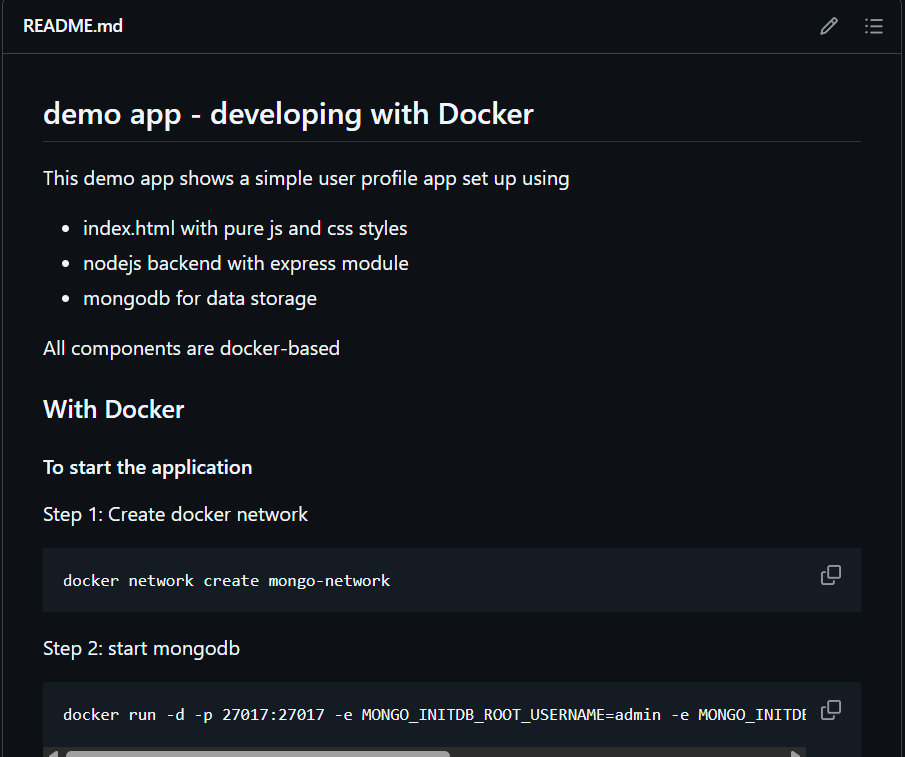


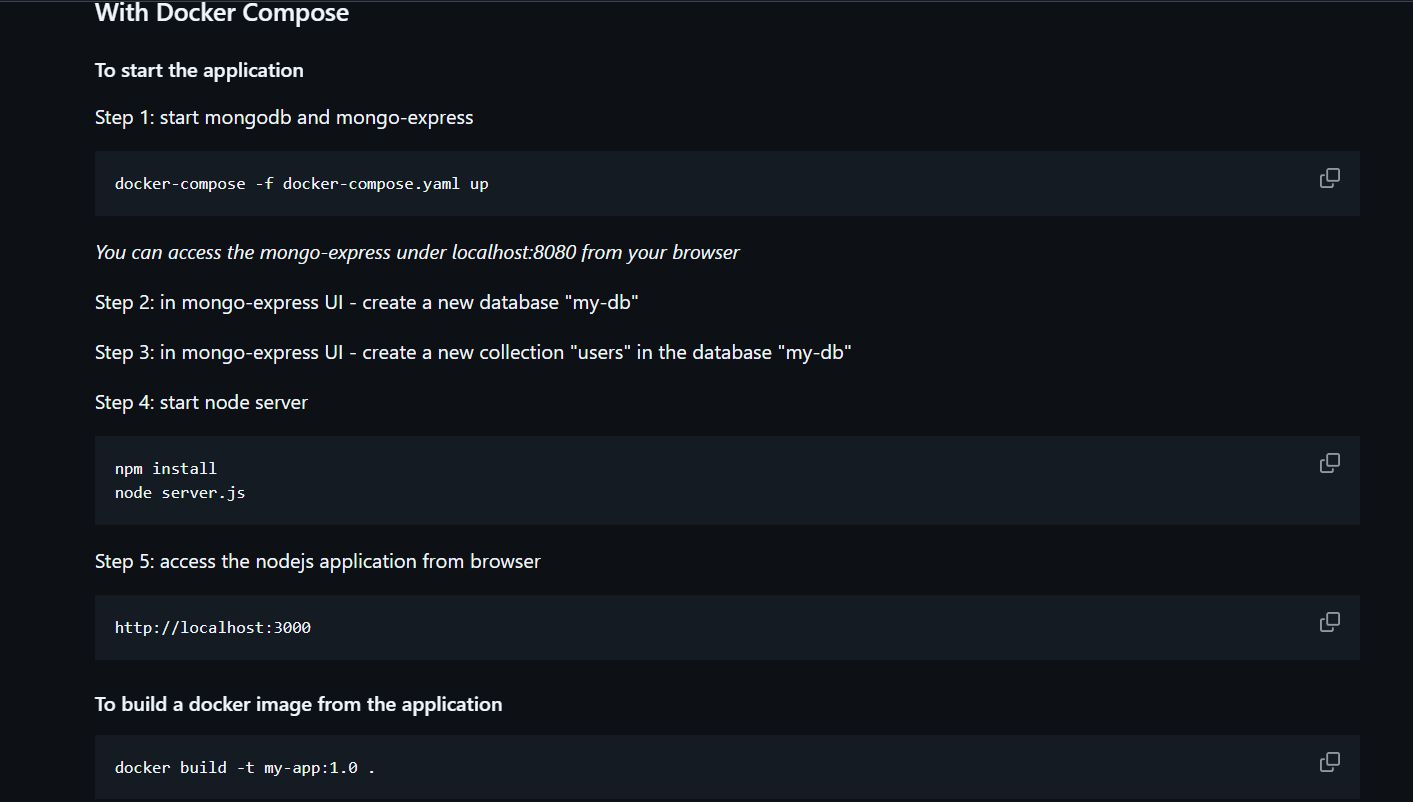




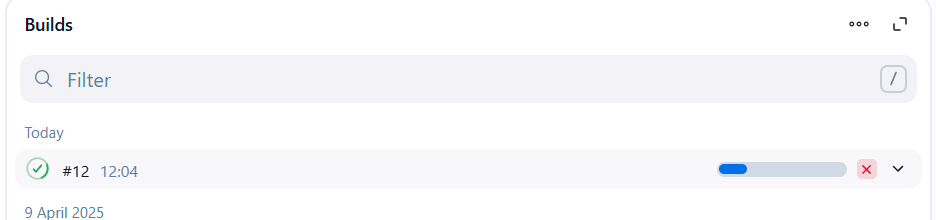


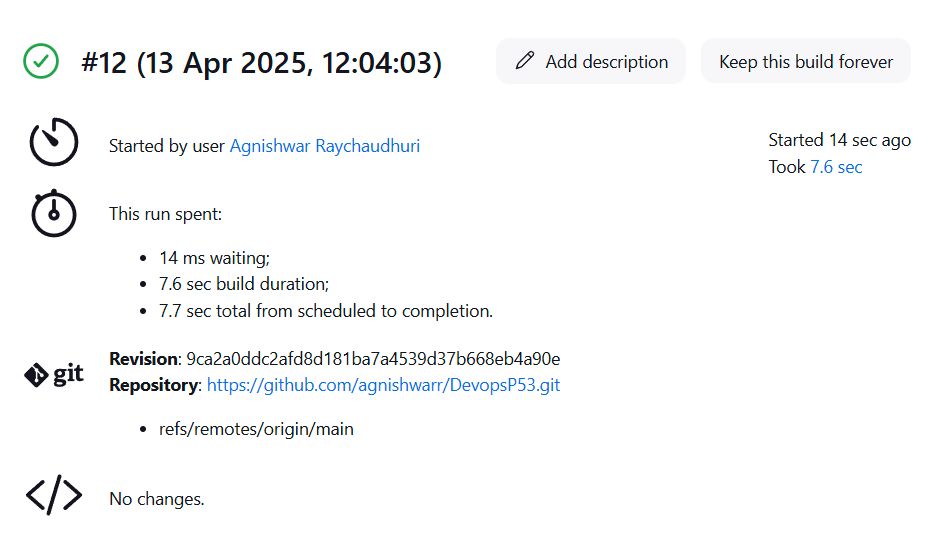
**Docker and Dockerfile**



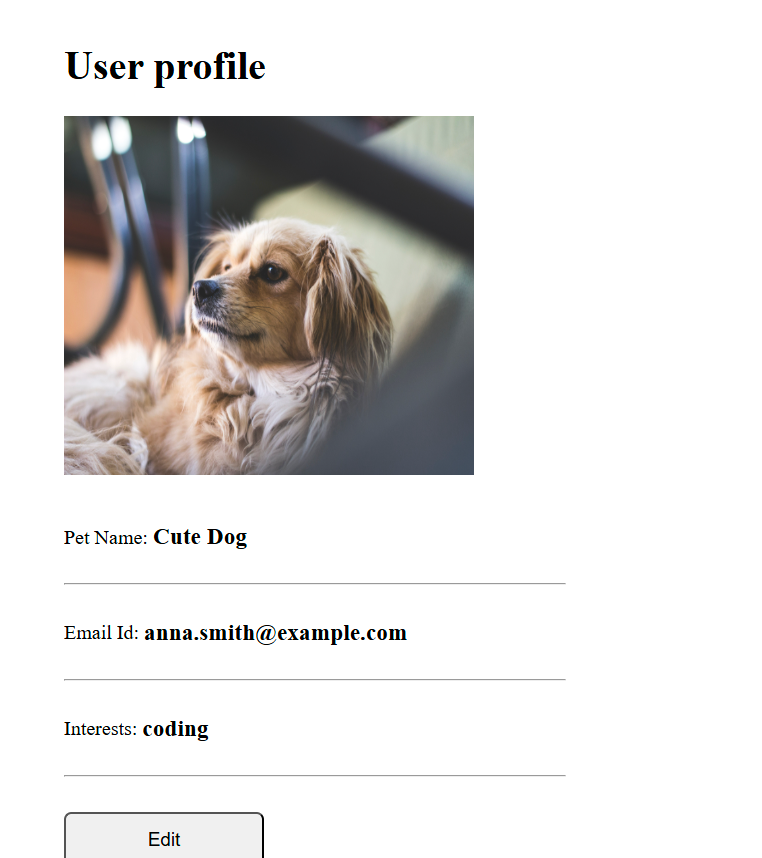


Let us now build the project using Jenkins





**Output**



**Practical 12:** APIs :

Host Student API and check it.