.[REGEX]*
Software Services- Learn To Create

Node.js & MongoDB Test - SKIT - 3rd August 2024

Total points 54/60 ?

All questions must be completed and Each questions comprise a 1 mark.

Test duration: 1.5 hours

The respondent's email (**ankitkumarrai1702@gmail.com**) was recorded on submission of this form.

0 of 0 points

Enter Your email *

ankitkumarrai1702@gmail.com

Enter Your Name *

Ankit Kumar Rai

Node.js & MongoDB Test - SKIT - 3rd August 2024 54 of 60 points

✓ Which keyword is used to declare a variable that is block-scoped and cannot be reassigned? *1/1

- ○ var
- ○ let
- [✓ const] ✓
- ○ function

✓ What is the output of the following code? * 1/1

```
let a = 10;
function change() {
  let a = 20;
  console.log(a);
}
change();
console.log(a);
```

- [✓ 20 10] ✓
- ○ 10 20
- ○ 20 20
- ○ undefined 10

✓ What does Promise.race() do? * 1/1

- [✓ Returns a promise that resolves or rejects as soon as one of the promises in the iterable resolves or rejects] ✓
- Returns a promise that resolves when all of the promises in the iterable have resolved
- Returns a promise that resolves when the first promise in the iterable rejects
- Returns a promise that resolves or rejects as soon as all of the promises in the iterable resolve

✓ To find documents where the availability field is true, which query would you use? * 1/1

- [✓ { availability: true }] ✓
- { availability: { $eq: true } }
- { availability: { $in: [true] } }
- { availability: { $exists: true } }

✓ What does the $group stage in an aggregation pipeline do? * 1/1

- Filters documents
- Sorts documents
- [✓ Groups documents by a specified field] ✓
- Projects specific fields

✓ What is the result of the following code?* * 1/1

```
const obj = {
  name: 'John',
  greet() {
    console.log(this.name);
  }
};
const greetFunc = obj.greet;
greetFunc();
```

- ○ John
- [✓ undefined] ✓
- ○ ReferenceError
- ○ null

✗ Which method is used to handle asynchronous operations in JavaScript? * 0/1

- ○ async/await
- ○ callbacks
- ○ promises
- [✓ All of the above] ✗

Correct answer

- [✓ async/await]

✓ Which of the following correctly defines a constant variable? * 1/1

- [✓ const variable = 10;] ✓
- let variable = 10;
- var variable = 10;
- variable = 10;

✓ What does the $lookup stage do in an aggregation pipeline? * 1/1

- Filters documents
- [✓ Joins documents from different collections] ✓
- Groups documents by a specified field
- Projects specific fields

✓ What will the following code output? * 1/1

```
async function fetchData() {
  let data = await fetch('https://api.example.com');
  console.log(data);
}
fetchData();
```

- Response
- [✓ Promise { Response }] ✓
- undefined
- Error

✓ How do you ensure a piece of code runs only after all promises in an array are resolved? * 1/1

- [✓ Promise.all()] ✓
- Promise.any()
- Promise.race()
- Promise.resolve()

✓ How does const differ from let? * 1/1

- const allows reassignment of values, let does not
- [✓ const does not allow reassignment of values, let does] ✓
- const is block-scoped, let is function-scoped
- const creates global variables, let creates local variables

✓ Which operator is used to filter documents based on an array field? * 1/1

- [✓ $elemMatch] ✓
- $in
- $ne
- $gt

✗ What will the be result of the following code? * 0/1

```
let result = (function() {
  return new Promise((resolve, reject) => {
    setTimeout(() => resolve('Done!'), 1000);
  });
})();
result.then(console.log);
```

- [✓ Done!] ✗
- Promise { 'Done!' }
- undefined
- Error

Correct answer

- [✓ Error]

✓ Which operator is used to find documents where a field value is within an array of values? * 1/1

- [✓ $in] ✓
- $ne
- $lt
- $gt

✓ Which method is used to handle asynchronous errors in JavaScript promises? *1/1

- [✓ catch] ✓
- finally
- then
- all

✗ How can you limit the number of documents returned by a query? * 0/1

- [✓ By using the $limit stage] ✗
- By using the $count stage
- By specifying the limit in the query options
- By using the $skip stage

Correct answer

- [✓ By specifying the limit in the query options]

✓ How would you query for products that are either in stock or have a price less than $20? *1/1

- [✓ { $or: [{ stock: { $gt: 0 } }, { price: { $lt: 20 } }] }] ✓
- { $and: [{ stock: { $gt: 0 } }, { price: { $lt: 20 } }] }
- { $or: [{ stock: { $lt: 0 } }, { price: { $gt: 20 } }] }
- { $and: [{ stock: { $lt: 0 } }, { price: { $gt: 20 } }] }

✓ Which operator is used to check for the presence of a specific field in documents? * 1/1

- [✓ $exists] ✓
- $type
- $regex
- $size

✓ How can you sort documents in descending order based on a field using aggregation? * 1/1

- $sort: { field: 1 }
- [✓ $sort: { field: -1 }] ✓
- $order: { field: 1 }
- $order: { field: -1 }

✓ Which operator is used to project fields in a document? * 1/1

- $group
- $match
- [✓ $project] ✓
- $sort

✓ What will be the output of the following code? * 1/1

```
function outer() {
  let outerVar = 'outer';
  function inner() {
    console.log(outerVar);
  }
  return inner;
}
const innerFunc = outer();
innerFunc();
```

- [✓ outer] ✓
- undefined
- ReferenceError
- inner

✓ Which operator is used to find documents where a field value matches a pattern? * 1/1

- $match
- [✓ $regex] ✓
- $text
- $search

✓ Which stage is used to calculate aggregate values such as sums or averages? *1/1

- [✓ $group] ✓
- $project
- $match
- $lookup

✓ Which method is used to wait for the result of a promise in an async function? *1/1

- [✓ await] ✓
- then
- catch
- finally

✓ How do you include the result of an aggregation operation in the output document? *1/1

- [✓ $addFields] ✓
- $project
- $merge
- $out

✓ To find all documents where the category field is either "Electronics" or "Books", which query would you use? * 1/1

- [✓ { category: { $in: ["Electronics", "Books"] } }] ✓
- { category: { $or: ["Electronics", "Books"] } }
- { category: { $eq: ["Electronics", "Books"] } }
- { category: { $ne: ["Electronics", "Books"] } }

✓ How do you sort documents in ascending order by a field? * 1/1

- [✓ { field: 1 }] ✓
- { field: -1 }
- { field: "asc" }
- { field: "desc" }

✗ How do you create a promise that resolves immediately with a value? * 0/1

- new Promise(resolve => resolve(value))
- Promise.resolve(value)
- new Promise((resolve, reject) => { resolve(value); })
- [✓ All of the above] ✗

Correct answer

- [✓ new Promise((resolve, reject) => { resolve(value); })]

✓ If you need to aggregate sales data by month and calculate the total sales per month, which stage is appropriate? * 1/1

- ○ $project
- [✓ ● $group] ✓
- ○ $lookup
- ○ $sort

✓ What does the async keyword indicate in a function? * 1/1

- [✓ ● The function will return a promise] ✓
- ○ The function will execute synchronously
- ○ The function will not use promises
- ○ The function will handle errors

✓ What does the await keyword do in an async function? * 1/1

- [✓ ● Blocks code execution until the promise is resolved] ✓
- ○ Allows code execution to continue immediately
- ○ Makes the function synchronous
- ○ Converts a promise into a callback

✓ How do you pass additional parameters to a callback function? * 1/1

- ○ By using closure
- [✓ By using the bind method] ✓
- ○ By using async/await
- ○ By using promises

✓ Which operator is used to project fields in a document? * 1/1

- ○ $group
- ○ $match
- [✓ $project] ✓
- ○ $sort

✓ What does the $unwind stage do in an aggregation pipeline? * 1/1

- [✓ Unwinds arrays into separate documents] ✓
- ○ Groups documents
- ○ Projects specific fields
- ○ Joins collections

✓ Which operator is used to match documents where a field value is not equal to a specific value? *1/1

- [✓ $ne] ✓
- $lt
- $gt
- $eq

✓ What is a closure in JavaScript? * 1/1

- [✓ A function that has access to its own scope, the outer function's scope, and the global scope] ✓
- A function that has access only to the global scope
- A function that can only be used within a block
- A function that creates a new object

✗ How can you retrieve only specific fields from documents in a query? * 0/1

- [✓ By using the $project stage] ✗
- By using the $select stage
- By specifying fields in the projection part of the query
- [✓ By using the $match stage] ✗

Correct answer

- [✓ By using the $select stage]

✓ What is the purpose of the finally block in a promise? * 1/1

- [✓ To handle both resolved and rejected states] ✓
- To handle only resolved states
- To handle only rejected states
- To initialize variables

✓ How can you find documents where the price field is less than the average price of all products? * 1/1

- [✓ { price: { $lt: { $avg: "$price" } } }] ✓
- { price: { $lt: { $avg: "#price" } } }
- { price: { $lt: { $sum: "$price" } } }
- - D) { price: { $lt: { $max: "$price" } } }

✓ What is the scope of a variable declared with let? * 1/1

- Global
- Function
- [✓ Block] ✓
- Object

✓ Which statement correctly defines a closure? * 1/1

- ○ A function that is passed as an argument
- [✓ ● A function that returns another function] ✓
- ○ A function that is called immediately
- ○ A function that defines a variable in the global scope

✓ How can you find documents where the date_added field is within the last 30 days? * 1/1

- [✓ ● { date_added: { $gte: new Date(Date.now() - 30 * 24 * 60 * 60 * 1000) } }] ✓
- ○ { date_added: { $lte: new Date(Date.now() - 30 * 24 * 60 * 60 * 1000) } }
- ○ { date_added: { $gt: new Date(Date.now() - 30 * 24 * 60 * 60 * 1000) } }
- ○ { date_added: { $lt: new Date(Date.now() - 30 * 24 * 60 * 60 * 1000) } }

✓ How do you handle errors in an async function? * 1/1

- [✓ ● Using try/catch] ✓
- ○ Using if/else
- ○ Using catch method of a promise
- ○ Using finally

✓ Which JavaScript construct allows a function to be executed only once? * 1/1

- ○ Closure
- [✓ ● IIFE (Immediately Invoked Function Expression)] ✓
- ○ Callback
- ○ Promise

✗ How can you retrieve documents where the description field contains the word "premium"? * 0/1

- ○ { description: { $regex: "premium" } }
- ○ { description: { $text: "premium" } }
- ○ { description: { $match: "premium" } }
- [✓ ● { description: { $search: "premium" } }] ✗

Correct answer

- ● { description: { $regex: "premium" } }

✓ What will be the output of the following code? * 1/1

```
javascript
function test() {
  var x = 1;
  if (true) {
    var x = 2;
    console.log(x);
  }
  console.log(x);
}
test();
```

- ○ 1 2
- ○ 2 1
- [✓ 2 2] ✓
- ○ ReferenceError

✓ What does the $count stage do in an aggregation pipeline? * 1/1

- [✓ Counts the number of documents in the pipeline] ✓
- ○ Counts the number of fields in a document
- ○ Groups documents by a specified field
- ○ Projects specific fields

✓ What operator can be used to check if a field exists in a document? * 1/1

- [✓ $exists] ✓
- $ne
- $in
- $type

✓ What is the purpose of the callback function in JavaScript? * 1/1

- [✓ To handle asynchronous operations] ✓
- To define a new variable
- To create an object
- To sort an array

✓ What is the output of the following code? * 1/1

```
function outer() {
  let x = 10;
  function inner() {
    x++;
    console.log(x);
  }
  return inner;
}
const innerFunc = outer();
innerFunc();
innerFunc();
```

- [✓ 11 12] ✓
- ○ 10 11
- ○ 11 11
- ○ 10 10

✓ How do you find documents where a field value is greater than a specific value? * 1/1

- [✓ { field: { $gt: value } }] ✓
- ○ { field: { $lt: value } }
- ○ { field: { $eq: value } }
- ○ { field: { $ne: value } }

✓ How can you execute multiple asynchronous operations in parallel? * 1/1

- [✓ Using Promise.all()] ✓
- Using Promise.allSettled()
- Using Promise.race()
- Using Promise.any()

✓ Which operator is used to find documents with a field value that is an array of a specific size? * 1/1

- [✓ $size] ✓
- $length
- $count
- $array

✓ What is the main difference between var and let? * 1/1

- var is block-scoped, while let is function-scoped
- [✓ var is function-scoped, while let is block-scoped] ✓
- var creates a new variable, while let updates an existing one
- var is immutable, while let is mutable

✓ To find documents where the rating field is between 4 and 5, which query would you use? * 1/1

- [✓ { rating: { $gte: 4, $lte: 5 } }] ✓
- { rating: { $lt: 4, $gt: 5 } }
- { rating: { $eq: 4, $ne: 5 } }
- { rating: { $in: [4, 5] } }

✓ Given a dataset with fields product_id, product_name, price, and stock, which query will find products with a price greater than $50? * 1/1

- { price: { $lt: 50 } }
- [✓ { price: { $gt: 50 } }] ✓
- { price: { $eq: 50 } }
- { price: { $ne: 50 } }

✓ Which MongoDB aggregation stage is used to filter documents? * 1/1

- $group
- [✓ $match] ✓
- $project
- $sort

✓ What will be the output of the following code? * 1/1

```
    async function foo() {
  return 1;
}
foo().then(console.log);
```

[✓ 1] ✓

Promise { 1 }

undefined

Error

✓ What will be the output of the following code? * 1/1

```
  let x = 1;
function test() {
  let x = 2;
  console.log(x);
}
test();
console.log(x);
```

[✓ 2 1] ✓

1 2

2 2

ReferenceError