RESEARCH ASSIGNMENT

Hamna Faisal

16L-4006

Q1- MongoDB is the software that manages durable storage and efficient retrieval & processing of data. Mongoose is a object data modeling (ODM) library that provides a rigorous modeling environment for your data. This means that Mongoose allows you to define objects with a strongly-typed schema that is mapped to a MongoDB document.

Mongoose provides an incredible amount of functionality around creating and working with schemas. Mongoose currently contains eight SchemaTypes that a property is saved as when it is persisted to MongoDB. Mongoose contains many different functions that allow you to validate, save, delete, and query your data using common MongoDB functions.

CREATE Operations

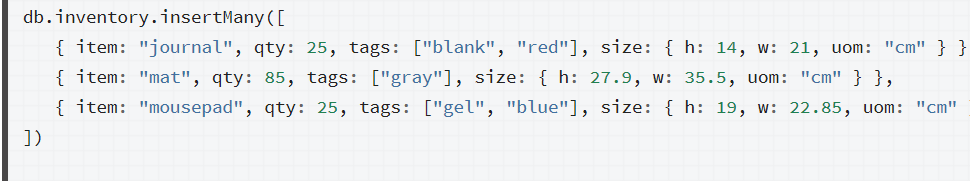
Create or insert operations add new [documents](https://docs.mongodb.com/manual/core/document/#bson-document-format) to a [collection](https://docs.mongodb.com/manual/core/databases-and-collections/#collections). If the collection does not currently exist, insert operations will create the collection.

MongoDB provides the following methods to insert documents into a collection:

[db.collection.insertOne()](https://docs.mongodb.com/manual/reference/method/db.collection.insertOne/#db.collection.insertOne)



[db.collection.insertMany()](https://docs.mongodb.com/manual/reference/method/db.collection.insertMany/#db.collection.insertMany)



READ Operations

Read operations retrieves [documents](https://docs.mongodb.com/manual/core/document/#bson-document-format) from a [collection](https://docs.mongodb.com/manual/core/databases-and-collections/#collections); i.e. queries a collection for documents. MongoDB provides the following methods to read documents from a collection:

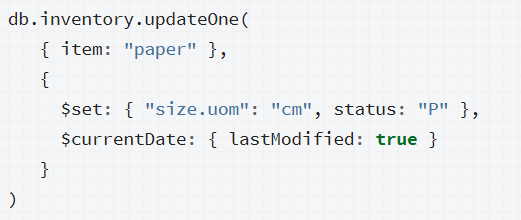
[db.collection.find()](https://docs.mongodb.com/manual/reference/method/db.collection.find/#db.collection.find)



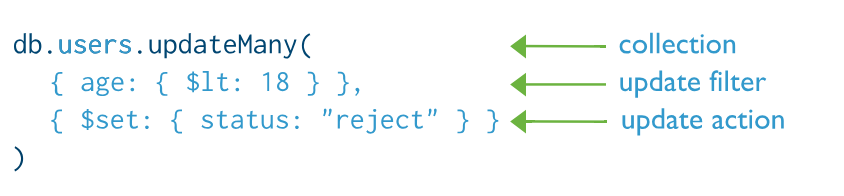
UPDATE Operations

Update operations modify existing [documents](https://docs.mongodb.com/manual/core/document/#bson-document-format) in a [collection](https://docs.mongodb.com/manual/core/databases-and-collections/#collections). MongoDB provides the following methods to update documents of a collection:

[db.collection.updateOne()](https://docs.mongodb.com/manual/reference/method/db.collection.updateOne/#db.collection.updateOne)



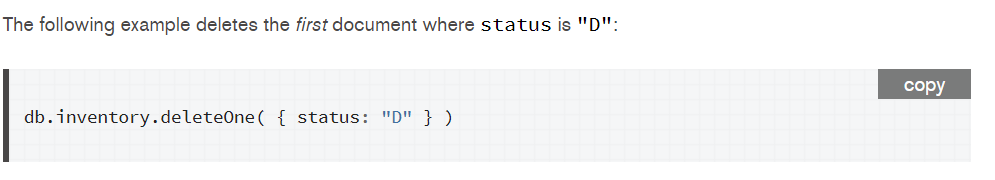
[db.collection.updateMany()](https://docs.mongodb.com/manual/reference/method/db.collection.updateMany/#db.collection.updateMany)



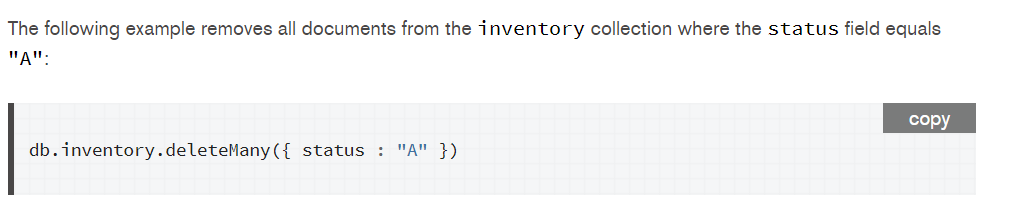
DELETE Operations

Delete operations remove documents from a collection. MongoDB provides the following methods to delete documents of a collection:

[db.collection.deleteOne()](https://docs.mongodb.com/manual/reference/method/db.collection.deleteOne/#db.collection.deleteOne)



[db.collection.deleteMany()](https://docs.mongodb.com/manual/reference/method/db.collection.deleteMany/#db.collection.deleteMany)



Q2- The POST method is used to request that the origin server accept the entity enclosed in the request as a new subordinate of the resource identified by the Request-URI in the Request-Line. It essentially means that POSTrequest-URI should be of a collection URI.

PUT method requests for the enclosed entity be stored under the supplied [Request-URI](https://restfulapi.net/resource-naming/). If the Request-URI refers to an already existing resource – an update operation will happen, otherwise create operation should happen if Request-URI is a valid resource URI (assuming client is allowed to determine resource identifier).

POST is NOT idempotent. So if you retry the request N times, you will end up having N resources with N different URIs created on server.

PUT method is [idempotent](https://restfulapi.net/idempotent-rest-apis/). So if you send retry a request multiple times, that should be equivalent to single request modification.

In the AJAX application we could use put request too but as you can use post to update a resource but not using the same URL as the resource you're updating. So, if the URL to use with PUT/PATCH is /api/cars/dealers/1 you'd have /api/cars/dealerupdate to send your POST requests with body as in your PATCH request.

Q3- The PUT method requests that the enclosed entity be stored under the supplied Request-URI. If the Request-URI refers to an already existing resource, the enclosed entity **should be considered as a modified version of the one residing on the origin server.**

The patch method requests that a set of changes described in the request entity be applied to the resource identified by the request-URI

Let's look at an example

{ "username": "skwee357", "email": "skwee357@domain.com" }

If we POST this document to /users, as you suggest, then you might get back an entity such as

## /users/1

{

"username": "skwee357",

"email": "skwee357@domain.com"

}

If we want to modify this entity later, choose between PUT and PATCH. A PUT might look like this:

PUT /users/1

{

"username": "skwee357",

"email": "skwee357@gmail.com" // new email address

}

You can accomplish the same using PATCH. That might look like this:

PATCH /users/1

{

"email": "skwee357@gmail.com" // new email address

}

When using PUT, it is assumed that we are sending the complete entity, and that complete entity replaces any existing entity at that URI. In the above example, the PUT and PATCH accomplish the same goal: they both change this user's email address. But PUT handles it by replacing the entire entity, while PATCH only updates the fields that were supplied, leaving the others alone.

Since PUT requests include the entire entity, if we issue the same request repeatedly, it should always have the same outcome (the data you sent is now the entire data of the entity). Therefore PUT is idempotent.

Q4- [React](https://reactjs.org/) is a JavaScript library for building user interfaces, released in 2013 by Facebook that is pretty much ideal for logicless applications or computation-less apps. On the other hand, [Angular](https://angular.io/) is a TypeScript-based, open-source front-end web application platform. It is an MVC framework created by Google. MVC is an architecture type for frameworks that develop user interfaces. It translates to **M**odel **V**iew **C**ontroller.

Throughout the years, AngularJs and ReactJs-the two prominent structure of JavaScript have been in discussion, there's no obvious answer at whatever point it comes to picking the best between two extremely well known and predictable structures. However, there are few points of interest of each and have some key contrasts too.

1. **Data Binding**

Angular allows two-way data binding while React allows one-way data binding. Two-way data binding means that any changes you make to the model affect the view, and vice versa. One-way data binding means any changes you make to the model affect the view, but not the other way around. Two way data binding is the primary key difference, it disperses after every set of data changes.

### ****DOM Usage****

DOM is the Data Object Model of a web app. **Angular** uses the browser's DOM, while **React** uses a virtual DOM which help developers manage an extensive database. By using a virtual DOM, you can change any element very quickly and without needing to render the whole DOM. It drastically changes the performance from good to excellent.

### ****Learning Curve****

This will differ from individual to individual based on skill and experience. On average, TypeScript is considered harder to learn than JSX, in turn increasing the learning curve with Angular as compared to React.

1. **Overall performance**

React has a prompt rendering feature that gives it a slight edge over the Angular JavaScript. It consists of various approaches to lessen the amount of DOM operation and thereby speeds up the updating process, making it more efficient.

According to above-mentioned factors both Angular and React works absolutely on diverse approaches. Both the technologies are superior; it entirely depends on system requirements that will help make the final decision. In case if you are building dynamic applications that uses virtual DOM, single page app(display all changes made to the content without reloading the current page.), and native app, React is the ideal choice and if you are building cross-platform mobile apps(addresses limiting factors such as navigation via touch, different screen sizes, and mobile hardware.), enterprise software(Angular uses an MVC architecture, it is great for creating enterprise level apps.)and progressive web apps or hybrid mobile apps, Angular is the best of all. In a nutshell, both the frameworks provide a vigorous set of tools for scalable, quality and reactive web-based applications.

Q5- Vue is a **progressive framework** for building user interfaces. Unlike other monolithic frameworks, Vue is designed from the ground up to be incrementally adoptable. The core library is focused on the view layer only, and is easy to pick up and integrate with other libraries or existing projects. On the other hand, Vue is also perfectly capable of powering sophisticated Single-Page Applications when used in combination with [**modern tooling**](https://vuejs.org/v2/guide/single-file-components.html) and [**supporting libraries**](https://github.com/vuejs/awesome-vue#components--libraries).

1. Flexibility

A great deal of flexibility is another advantage of Vue.js. It allows the user to write his template in HTML file, JavaScript file, and pure JavaScript file using virtual nodes. This flexibility also makes it easy to understand for the developers of React.js, Angular.js, and any other new JavaScript framework. Vue.js has proved a lot beneficial in the development of those simple applications that run directly from browsers.

1. Size

The success of JavaScript framework depends on its size. The smaller the size is, the more it will be used. One of the greatest advantages of Vue.js is its small size. The size of this framework is 18–21KB and it takes no time for the user to download and use it. This does not mean that it has low speed because of small size. Instead, it beats all the bulky frameworks like React.js, Angular.js, and Ember.js.

1. Learning curve

One of the reasons for the popularity of this framework is that it is quite easy to understand. The user can easily add Vue.js to his web project because of its simple structure. Both the small as well as large scales templates can be developed through this framework which saves a lot of time. In case of any problem, the user can easily trace the blocks with errors. All this is because of its simple structure.

1. [Directives vs Components](https://vuejs.org/v2/guide/comparison.html#Directives-vs-Components)

Vue has a clearer separation between directives and components. Directives are meant to encapsulate DOM manipulations only, while components are self-contained units that have their own view and data logic. In AngularJS, directives do everything and components are just a specific kind of directive.

Q6- AngularJS is an open-source, JavaScript-based, front-end web application framework for dynamic web app development. It utilizes HTML as a template language. By extending HTML attributes with directives and binding data to HTML with expressions, AngularJS creates an environment that is readable, extraordinarily expressive and quick to develop.

AngularIO is the blanket term used to refer to Angular 2, Angular 4 and all other versions that come after AngularJS. Both Angular 2 and 4 are open-source, TypeScript-based front-end web application platforms.

Angular 4 is the latest version of Angular. Although Angular 2 was a complete rewrite of AngularJS, there are no major differences between Angular 2 and Angular 4. Angular 4 is only an improvement and is backward compatible with Angular 2.

Differences between AngularJS and AngularIO:

1. Three different types:

Angular JS is an open source JS-based, frontend web application framework for dynamic web app development whereas, AngularIO (angular-2 and angular-4) are open-source, Typescript based frontend web application platforms.

1. Architecture:

The architecture of angularJS is based on MVC structure i.e. Models, Views and Controllers. This in angularIO, was first replaced by controllers and directives, which were later on improved.

1. Language:

Angular JS is written in JavaScript. Angular uses Microsoft’s TypeScript language, which is a superset of ECMAScript 6 (ES6). This has the combined advantages of the TypeScript features, like type declarations, and the benefits of ES6, like iterators and lambdas. Angular 4 is compatible with the most recent versions of TypeScript that have powerful type checking and object-oriented features.

1. Speed:

By providing features like 2-way binding, AngularJS reduced the development effort and time. However, by creating more processing on the client side, page load was taking considerable time. Angular2 provides a better structure to more easily create and maintain big applications and a better change detection mechanism. Angular 4 is the fastest version yet.

1. Routing:

AngularIO has simpler routing as compared to Angular JS. For example, AngularJS uses $routeprovider.when() to configure routing while Angular uses @RouteConfig{(…)}.

Q7- **Linting** is the process of checking the source code for Programmatic as well as Stylistic errors. This is most helpful in identifying some common and uncommon mistakes that are made during coding.

**JSLint** is a [static code analysis](https://en.wikipedia.org/wiki/Static_code_analysis) tool used in [software development](https://en.wikipedia.org/wiki/Software_development) for checking if [JavaScript](https://en.wikipedia.org/wiki/JavaScript) [source code](https://en.wikipedia.org/wiki/Source_code) complies with [coding rules](https://en.wikipedia.org/wiki/Programming_style). It is provided primarily as a [web application](https://en.wikipedia.org/wiki/Web_application) through jslint.com, but there are also command-line adaptations. It was created in 2002 by [Douglas Crockford](https://en.wikipedia.org/wiki/Douglas_Crockford).

**Advantages**

* Comes configured and ready to go (if you agree with the rules it enforces)

**Disadvantages**

* JSLint doesn’t have a configuration file, which can be problematic if you need to change the settings
* Limited number of configuration options, many rules cannot be disabled
* Can’t add custom rules
* Undocumented features
* Difficult to know which rule is causing which error

ESLint is the most recent out of the four. It was designed to be easily extensible, comes with a large number of custom rules, and it’s easy to install more in the form of plugins. It gives concise output, but includes the rule name by default so you always know which rules are causing the error messages.

Uses

ESLint is a tool for identifying and reporting on patterns found in JavaScript code, with the goal of making code more consistent and avoiding bugs. ESLint uses [Espree](https://github.com/eslint/espree) for JavaScript parsing, ESLint uses an AST to evaluate patterns in code and ESLint is completely pluggable, every single rule is a plugin and you can add more at runtime.

### Advantages

* Flexible: any rule can be toggled, and many rules have extra settings that can be tweaked
* Very extensible and has many plugins available
* Easy to understand output
* Includes many rules not available in other linters, making ESLint more useful for detecting problems
* Best ES6 support, and also the only tool to support [JSX](https://jsx.github.io/)
* Supports custom reporters

### Disadvantages

* Some configuration required
* Slow, but not a hindrance

## My Recommendations

My choice of these two is ESLint as it is extensible, flexible and easy to interpret and understand

Q8- The main purpose of Ajax is to improve the speed, performance and usability of a web application. A great example of Ajax is the movie rating feature on [Netflix](https://signup.netflix.com/MediaCenter). The user rates a movie and their personal rating for that movie will be saved to their database without waiting for the page to refresh or reload. These movie ratings are being saved to their database without posting the entire page back to the server. Facebook, Gmail, and Pinterest are examples of sites that use a lot of AJAX.

The “Asynchronous” part refers to the fact that when the JavaScript makes the AJAX call to the webserver, it continues to work until the response – it doesn’t “block” and stop while the data is being processed server-side.

Angularjs is best used for single page applications which provide the client with a fluid experience because all the code is retrieved up-front or dynamically loaded as necessary to create an experience that feels like a desktop application than a traditional, multi-page website