**Name: Ahmad Mushtaq Ahmad**

**RollNo.: 16L-4245**

**Course: Web**

**Instructor: Shamsa Abid**

**RESEARCH ASSIGNMENT**

**Q.1** MongoDB vs Mongoose. Why are we using Mongoose package instead of MongoDB package? How would we perform CRUD operations using the MongoDB package? Give code examples for creating, retrieving, updating and deleting MongoDB documents. (8)

**Answer:** Using Mongoose, a user can define the schema for the documents in a particular collection. It provides a lot of convenience in the creation and management of data in MongoDB. On the downside, learning mongoose can take some time, and has some limitations in handling schemas that are quite complex.

However, if your collection schema is unpredictable, or you want a Mongo-shell like experience inside Node.js, then go ahead and use the MongoDB driver. It is the simplest to pick up. The downside here is that you will have to write larger amounts of code for validating the data, and the risk of errors is higher.

We are using mongoose because of our requirement and features, advantages we get from mongoose as given above.

For performing CRUD we need to follow these steps first.

1.Installation of MongoDb and NodeJs.

2.Install dependencies.

3.Create model.

4.Create Server.

5.Connect to server

6.Create middle-ware route.

## Read Operations: Read operations retrieves documents 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:

## Examples:

## db.inventory.find( {} )

The following example retrieves all documents from the inventory collection where status equals either "A" or "D":

## db.inventory.find( { status: { $in: [ "A", "D" ] } } )

n the following example, the compound query document selects all documents in the collection where the status equals "A" and either qty is less than ([$lt](https://docs.mongodb.com/manual/reference/operator/query/lt/#op._S_lt)) 30 or item starts with the character p:

## db.inventory.find( {

## status: "A",

## $or: [ { qty: { $lt: 30 } }, { item: /^p/ } ]

## } )

The following example queries for all documents where tags is an array that contains the string "red" as one of its elements:

## db.inventory.find( { tags: "red" } )

## { <array field>: { <operator1>: <value1>, ... } }

**Create Operations:** Create or insert operations add new documents to collection. If the collection does not currently exist, insert operations will create the collection.

Examples:

db.inventory.insertOne(

{ item: "canvas", qty: 100, tags: ["cotton"], size: { h: 28, w: 35.5, uom: "cm" } }

)

For many,

db.inventory.insertMany([

{ item: "journal", qty: 25, tags: ["blank", "red"], size: { h: 14, w: 21, uom: "cm" } },

{ item: "mat", qty: 85, tags: ["gray"], size: { h: 27.9, w: 35.5, uom: "cm" } },

{ item: "mousepad", qty: 25, tags: ["gel", "blue"], size: { h: 19, w: 22.85, uom: "cm" } }

])

## Update Operation:

## Examples:

## {

## <update operator>: { <field1>: <value1>, ... },

## <update operator>: { <field2>: <value2>, ... },

## ...

## }

## db.inventory.updateOne(

## { item: "paper" },

## {

## $set: { "size.uom": "cm", status: "P" },

## $currentDate: { lastModified: true }

## }

## )

## db.inventory.updateMany(

## { "qty": { $lt: 50 } },

## {

## $set: { "size.uom": "in", status: "P" },

## $currentDate: { lastModified: true }

## }

## )

## Delete Operation:

## Examples

## Delete only one document that matches the condition

## db.inventory.deleteOne( { status: "D" } )

## Delete all the documents in the inventory that matches the condition

## db.inventory.deleteMany({ status : "A" })

**Q.2** POST vs PUT. Why was POST used to update employee data in the Employees sample AJAX application? What happens if you replace POST with PUT? What difference does it make?

**Answer:** The HTTP POST (i.e. to create) method is used to send user-generated data to the web server. For example, a POST method is used when a user comments on a forum or if they upload a profile picture.

The PUT(i.e. to create/update) method completely replaces whatever currently exists at the target URL with something else. With this method, you can create a new resource or overwrite an existing one given you know the exact Request-URI

PUT method is idempotent because no matter how many times we send the same request, the results will always be the same. On the other hand, the POST method is not idempotent since if we send the same POST request multiple times, we will receive various results (i.e. a new subordinate will be created each time).

In the AJAX application we could use put request too (according to our respected instructor) 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.

**Q.3** PUT vs PATCH. Can PUT be used for partial updates e.g. in case of updating your name in an online university application form what method should be used? PUT or PATCH or anyone? Why?

Answer: When a there is a need to replace an existing resource entirely, they can use PUT. When they’re doing a partial update, they can use HTTP PATCH.

For instance, when updating a single field of the Resource, sending the complete Resource representation might be cumbersome and utilizes a lot of unnecessary bandwidth. In such cases, the semantics of PATCH make a lot more sense.

Another important aspect to consider here is idempotence; put is idempotent; PATCH can be, but isn’t required to.

Yes, but then you'll have to document that for your clients, and you'll have to find another buzzword for your API, because that's not RESTful. Based on RFC 7231, PUT should be used only for complete replacement of a representation, in an idempotent operation. PATCH should be used for partial updates, therefore, patch should be used in this case

**Q.4** Where does React lie in comparison to AngularJS? Compare with respect to advantages and disadvantages.

**Answer:**

* **Scalability:**  
  Angular is easy to scale thanks to its design as well as a powerful CLI. React is testable and therefore scalable compared to other frameworks like Vue.
* **Computed Properties**  
  As far as performance is concerned, plain getters in Angular are out of the scenario because they get called on each render. It is however possible to use BehaviorSubject from RsJS, as it serves the purpose.  
  React allows the use of @computed from MobX – achieving the same objective but with a nicer API. Dependency Injection.
* **Dependency Injection**  
  Dependency injection is a bone of contention as it is contrary to the prevalent React paradigm of functional programming and immutability. But some sort of dependency injection is almost unavoidable in data-binding environments, because it aids in decoupling (and thereby mocking and testing) where there is no separate data-layer architecture. Angular supports DI and one great advantage is the ability to have different lifecycles for different stores.

Some of the common React paradigms deploy some kind of global app state that maps to disparate components, however it is conducive to the introduction of bugs when cleaning the global state on component unmount. On the other hand, a store that is created on component mount and is seamlessly available to the component’s children –is a more useful and often neglected concept. This is out of the box in Angular, but quite easily replicated with MobX as well.

* **Simplicity + Code length**  
  React is quite easy and simple to understand but it takes quite some time to set up a project in React.  
  Angular on the other hand, is not simple by any means. Its inherent complexity sometimes causes confusion and Angular specific 3rd party libraries and syntax.
* **Model Complexity**  
  Angular’s performance is sensitive in terms of scope because of copy-n-compare. You cannot use large models. However, this has benefits and disadvantages. The Pros are that it makes the code simpler and more testable but the cons are that you need to break down stuff that you normally use and rebuild it again (for example – for server requests). React however gives you the power of choice without the performance penalty. The outcome really depends on whether you’re a good coder or a bad coder.

**Q.5** What is Vue.js? Where does Vue lie in comparison to React and AngularJS? Give examples.

**Answer:**

In components React and Vue both excel at handling dumb components: small, stateless functions that receive an input and return elements as output. For example, you could have a grid component (consisting of a header component and several row components) with various properties (columns, header information, data rows, etc.) and be able to reuse the component with different data sets on another page.

In terms of size and performance there is a flip-side to all the functionality: the Angular framework is quite bloated. The gzipped file size is 143k, compared to 23K for Vue and 43k for React. Vue has great performance and the deepest memory allocation, but all these frameworks are really pretty close to each other when compared to particularly slow or fast frameworks (like Inferno).

React breaks with long-standing best practices. For decades, developers were trying to separate UI templates and inline Javascript logic, but with JSX, these are intermixed again. The Angular templates are enhanced HTML with special Angular language (Things like ngIf or ngFor). Vue features “single file component”. This seems like a trade-off with regard to the separation of concerns — templates, scripts and styles are in one file but in three different, ordered sections. This means you get syntax highlighting, CSS support and easier use of preprocessors like Jade or SCSS.

**Q.6** How is AngularIO different from AngularJS? Give examples.

**Answer:**  Angular is based on Typescript while AngularJS is based on JavaScript. Typescript is a superset of ES6 and its backward compatible with ES5. Angular has also benefits of ES6 like: lambda operators, iterators or reflection’s mechanism.   
AngularJS uses terms of scope and controller. To scope a variable you can add many variables that will be visible in View as well as in Controller. AngularJS has also a concept of root Scope. Variables in root Scope are available on all throughout application. Angular does not have a concept of scope or controllers. Instead of them it uses a hierarchy of components as its main architectural concept.

AngularJS has many directives and every developer can also specify custom new directive. Angular also has standard directives, but they are used in a bit different way. For example: ng-model in AngularJS means that you want to create two-way binding. If you want to create one-way binding, you should use ng-bind. Angular occurs only ngModel, but if you would write it only in: “[ ]”, you’ll get one-way binding. If you want to create two-way binding you must write it in: “[( )]”. We have to write it this way because of the fact that “[ ]” is used to property binding and “( )” is used to event binding. In Angular, some directives have changed their names like ng-repeat to ngFor.

Angular is newer version you can suspect that has some advantages over the old version. That’s right, Angular has many advantages. The first is modularity. Much core functionality was moved to different modules. That caused lighter and faster core, dynamic loading, asynchronous template compilation and added support for reactive programming. After beta version creators added really great thing: angular cli. With that package you can easily create scaffolding of your Angular project which will be all configured.

**Q.7** What is Linting? What is the use of JSLint? What is the use of ESLint? Give examples.

**Answer:** Linting is the process of running a program that will analyse code for potential errors. JSLint is a static code analysis tool used in software development for checking if JavaScript source code complies with coding rules. It is provided primarily as a web application through jslint.com, but there are also command-line adaptations. ESLint is an open source JavaScript linting utility originally created by Nicholas C. Zakas in June 2013. Code linting is a type of static analysis that is frequently used to find problematic patterns or code that doesn't adhere to certain style guidelines.

**Q.8** Give an example where you would prefer to use AngularJS over AJAX and vice versa.

**Answer:** AJAX is a way for JavaScript to request data from a server without refreshing the page or blocking the application. jQuery is a JavaScript library built to automate and simplify common web tasks like AJAX or animation. Angular is a hip JavaScript framework which is made for building large, single-page web applications.

$http is the service which you can inject in your angularjs controller and make the requests but if you want to use ajax features like async promises and others it will be time consuming for you to learn all that so in that case you can also simply put the ajax post method in the js of controller it will work fine.

**Example:** So in online form filling like signup if email or cell number already exist for that check we would prefer ajax over angular. But once we are logged in or lets say we our on a website which is a single page application like fb we scroll down and up and data get loads then in this scenario we would prefer angular over ajax

**References:**

https://gorrion.io/blog/angularjs-vs-angular

https://medium.com/unicorn-supplies/angular-vs-react-vs-vue-a-2017-comparison-c5c52d620176

https://www.rishabhsoft.com/blog/reactjs-vs-angularjs

https://stackoverflow.com/questions/19732423/why-isnt-http-put-allowed-to-do-partial-updates-in-a-rest-api

https://www.baeldung.com/http-put-patch-difference-spring