Web Programming Assignment

**Q1.**

**MongoDB vs Mongoose:**

MongoDB is a NoSQL database system which stores data in the form of BSON documents. In terms of Node.js, mongodb is the native driver for interacting with a mongodb instance and Mongoose is an object document modeling (ODM) layer that sits on top of Node's MongoDB driver. If your coming from SQL, it's similar to an ORM for a relational database. While it's not required to use Mongoose with the Mongo, here are four reasons why using Mongoose with MongoDB is generally a good idea.

**Reasons to use Mongoose:**

1. Schemas

MongoDB is a denormalized NoSQL database. This makes it inherently schema-less as documents have varying sets of fields with different data types. While this provides your data model with flexibility as it evolves over time, it can be difficult to cope with coming from a SQL background. Mongoose defines a schema for your data models so your documents follow a specific structure with pre-defined data types.

2. Validation

Mongoose has built in validation for schema definitions. This saves you from writing a bunch of validation code that you have to otherwise write with the MongoDB driver.

3. Instance Methods

Mongoose provides optional pre and post save operations for data models. This makes it easy to define hooks and custom functionality on successful reads/writes etc. You can also define custom methods that act on a particular instance (or document). While you can achieve similar functionality with the native MongoDB driver, Mongoose makes it easier to define and organize such methods within your schema definition.

4. Returning results

Mongoose makes returning updated documents or query results easier. A prime example can be found with update queries. While the native driver returns an object with a success flag and the number of documents modified, Mongoose returns the updated object itself so you can easily work with the results.

**CRUD Operations:**

Create Operations:  
db.users.insertOne({

Name:”Farooq”,

Age:26

})

Read Operations:  
db.users.find({

{Name:”Farooq”},

{Age:26}

})

Update Operations:  
db.users.updateOne({

{Name:”Farooq”},

{$set: {Age:26}}

})

Delete Operations:  
db.users.DeleteMany({

Age:26

})

**Q2.**

**POST vs PUT:**

Put is used to create a resource, or overwrite it. While you specify the resources new URL, Post is used to modify and update a resource.

Better is to choose between PUT and POST based on idempotence of the action.

PUT implies putting a resource - completely replacing whatever is available at the given URL with a different thing. By definition, a PUT is idempotent. Do it as many times as you like, and the result is the same. x=5 is idempotent. You can PUT a resource whether it previously exists, or not (e.g, to Create, or to Update)!

POST updates a resource, adds a subsidiary resource, or causes a change. A POST is not idempotent, in the way that x++ is not idempotent.

Because of idempotence nature of PUT, POST was used in the employee AJAX app to update records. Thus we could Update an employee as many times as we wanted, had we used PUT with its idempotence nature, all attempts to update an employee would only give result one time.

**Q3.**

**PUT vs PATCH:**

The existing HTTP PUT method only allows a complete replacement of a document.Thus, a PUT request always contains a full resource. This is necessary because, a necessary quality of PUT requests is idempotence — the quality of producing the same result even if the same request is made multiple times.

While HTTP PATCH method, is used to modify an existing HTTP resource.A PATCH request on the other hand, is used to make changes to part of the resource at a location. That is, it PATCHES the resource — changing its properties. It is used to make minor updates to resources and it’s not required to be idempotent.

No, PUT should not be used to change only just my name in a University application form. PATCH should be the technical choice here.

**Q4.**

**ReactJS vs AngularJS:**

The primary difference between AngularJS and ReactJS lies in state management. Angular has data-binding bundled in by default, whereas React is generally augmented by Redux to give unidirectional data flow and work with immutable data. Both are opposing approaches and there is no consensus on which is better: mutable/data binding or immutable/unidirectional.

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 Behavior Subject from RsJS, as it serves the purpose.

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.

Conclusion: Which Is Better React or Angular?

Both React and Angular work on completely diverse approaches to front-end application development for startup, small and medium enterprises. The technologies are both powerful and flexible and while neither is better or worse, it depends on your business application goals and system constraints that make the final choice.

Overall, both frameworks provide a robust set of tools for quality, scalable, reactive web-based applications. For those who prefer to code in classic JS, React may find more favors, but for those looking at a more mature and sophisticated solution, AngularJS might be your best bet.

**Q5.**

**VueJS:**

Vue.js is a progressive framework typically used for building user interfaces. It provides developers with the capability to build with modern technologies. Vue.js has gained a lot of popularity due to its gentle learning curve, and because of its scalability. It can be used in single page applications as well as small components in large websites.

**VueJS vs ReactJS vs AngularJS:**

For a real engineer, there is no substantial difference which framework to choose, because it just takes some time to get used to the new one. In our company, we grow expertise in mostly ReactJS and Angular 2/4/5, but Vue.js is also on board. Every framework has its own pros and cons, meaning that there should be just a right choice for every single case during the product development.

Pros:

Smaller in size: VueJS compared to other two frameworks is really small and lightweight. It just takes 18kb space with a zip file. In spite of such small size, it is capable to do better than many bulky frameworks.

Easy for developers: Offering developers capability to build large-scale templates it saves a lot of time and makes the development easier.

Easy for integrating components: whether you want to develop a single page app from the scratch or want to integrate components into an existing app, VueJS makes integration extremely easier.

Very well written and detailed documentation: Vue.js offers a very comprehensive and detailed documentation to help beginners writing an app easily.

Flexibility: VueJS is very flexible for building all sorts of apps ranging from simple browser apps to highly sophisticated and feature-rich apps.

Two-way binding: Vue.js allows two-way data binding to boost responsiveness of apps and capability to work with a variety of components.

Cons:

A limited developer community: Without being as popular as the React or Angular Vue.js is backed by a closed and smaller community of developers.

Language constraint: From the creator to the majority of the developers, most of them belong to non-English speaking community and that is one big constraint for developers to access community help.

The overburden of options: Being too flexible it comes with too many options that at times it felt like a burden, especially in large projects involving too many developers.

**Example:**

new Vue({

el: '#editor',

data: {

input: '# hello'

},

computed: {

compiledMarkdown: function () {

return marked(this.input, { sanitize: true })

}

},

methods: {

update: \_.debounce(function (e) {

this.input = e.target.value

}, 300)

}

})

**Q6.**

**AngularIO vs AngularJS:**

Angular.io is angular (without the js at the end) its also called angularjs 2 or angularjs 3,4 which adds to the confusion.

It uses typescript, or ES5/ES6 and is newer.

AngularJs

The original angularjs is:

1. More popular
2. Has more libraries and frameworks already established for it
3. Uses javascript

I recommend using AngularJS due to its easier learning curve and diverse collection of libraries and community.

|  |  |
| --- | --- |
| Angular JS | Angular |
| Bindings/interpolation  Your favorite hero is: {{vm.favoriteHero}} | Bindings/interpolation  Your favorite hero is: {{favoriteHero}} |
| Filters  <td>{{movie.title | [uppercase](https://angular.io/api/common/UpperCasePipe)}}</td> | Pipes  <td>{{movie.title | [uppercase](https://angular.io/api/common/UpperCasePipe)}}</td> |
| Local variables  <tr ng-repeat="movie in vm.movies">  <td>{{movie.title}}</td>  </tr> | Input variables  <tr \*ngFor="let movie of movies">  <td>{{movie.title}}</td>  </tr> |
| ng-click  <buttonng-click="vm.toggleImage()">  <buttonng-click="vm.toggleImage($event)"> | Bind to the click event  <button (click)="toggleImage()">  <button (click)="toggleImage($event)"> |

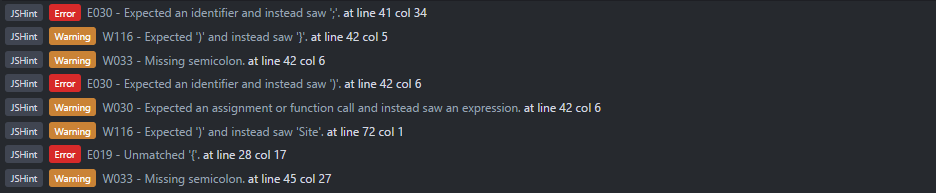
**Q7.**

**Linting:**

Linting is the process of running a program that will analyse code for potential Programmatic as well as Stylistic errors. The term is now applied generically to tools that flag suspicious usage in software written in any computer language.

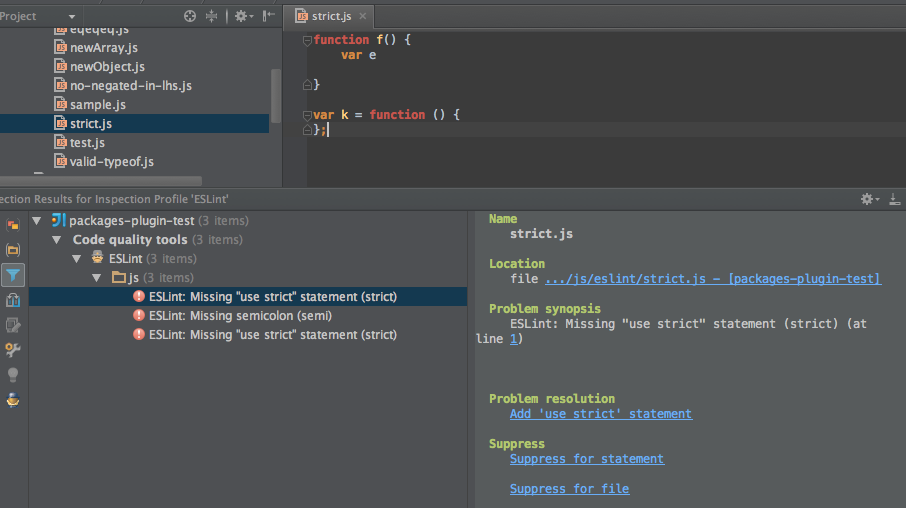
**JSLint:**

JSLint is a static code analysis tool used in software development for checking if JavaScript source code complies with coding rules.



**ESLint:**

A pluggable and configurable linter tool is used for identifying and reporting on patterns in JavaScript.



**Q8.**

**AngularJS vs AJAX:**

AJAX as a way to communicate (send requests and get responses) with the server without refreshing. i.e. Asynchronously.

AngularJS is a full fledged, front end MVC framework which does a lot more. It extends the above ($http module) with a lot of neat features such as 2-way data binding, templating, filters and directives etc.

Most of the time, people fail to comprehend the real value of these technologies during application development. AngularJS is best suited for the web application development as it works on the HTML code and JSON data which helps in developing for interactive and robust applications but using the same for a simple website development results in slow loading and quite erratic websites.

While jQuery AJAX is a fast and feature-rich language which has a a commendable JavaScript library and a great tool for creating feature-rich websites. It has in-built features such as HTML document traversal, event handling, manipulation, animation and Ajax support and others which make it easier and simpler to develop hardcore websites. Therefore before utilizing any of these highly intuitive and robust languages, it is necessary to frame a sound approach dedicated either to develop an advanced web application or website development.