## Why would you consider a Scripting Language as JavaScript as your backend platform

To simplify the process of going from frontend to backend, and use the same language for both parts. Node.js is also faster than Java so that is another thing to consider when choosing what language to use for your backend.

## Explain Pros & Cons in using Node.js + Express to implement your backend compared to a strategy using for example Java/JAX-RS/Tomcat

### Pros:

* Using the same language for both frontend and backend. This is great as it makes the same programmer capable of writing both the frontend and the backend without extensive knowledge in two different languages.
* Database queries. Related to the first pro, as for writing database queries you don’t have to change language into SQL. This is due to NOSQL databases and that newer databases like CouchDB are written in JavaScript.
* JSON which stands for JavaScript object notation is easy to use in Node.js as no parsers are needed to change the format of the code. This is needed in Java when you get a query back from a database or a rest API.
* Node.js is faster than Java as it doesn’t set up separate threads so there is no overhead to slow anything down.

### Cons:

* New. When using Java you have more than 20 years of testing that the Java Virtual Machine works as it’s supposed to. You don’t have that certainty when working with Node.js
* The IDEs for Node.js is not as great as they are for Java. Both Netbeans and Ecplipse are great tools with debuggers, decompilers and servers. Writing code for Node.js is done in a text editor by many programs and even though Webstorm is a good IDE, it’s just not as well developed as Netbeans or Eclipse.
* Debugging is a lot easier to do in a language like Java as it’s been used by big companies that demand they can fail proof their systems, and a good debugger is needed for that.
* Java libraries are simply better as they have been worked on for longer. There are great libraries for JavaScript but the depth and quality of the Java code base is superior.
* Deadlocks. If one request is slow it slows down everything else. This is because JavaScript doesn’t use threads. You hardly ever notice this after the new V8 engine for JavaScript

## Compare, using relevant examples, the express strategy toward templating with the one you used with Java on second semester

## Explain using relevant examples the Express concept middleware

An express application is often a series of middleware calls. Middleware functions are functions that have access to the request and response object (req, res), and also has a reference to the next middleware function (next). Middleware can execute any code. Make changes to the request and response objects and call the next middleware function in the stack.

The last middleware in an application does not have to call the next() to pass the next middleware function, but if any other middleware functions don’t call the next() the request will be left hanging.

You can also use middleware as error handling. When creating an error handling function you should pass it 4 arguments in stead of 3. The arguments are err, req, res, next, so three of them are the same and then we have an error object as well.

A lot of the middleware you need is already created such as body-parser or cookie-parser. You can get these kinds of middleware by writing in the console: “npm install body-parser”.

## Explain using relevant examples how to implement sessions and the legal implications of doing this

Sessions are used to store information about user behavior. This can be how many clicks a user does on a website or storing the username of a user to display other places.

When using cookies/sessions it’s important to inform the user of what is being stored and that the user gives his or her consent in storing the data.