Why would you consider a Scripting Language as JavaScript as your Backend Platform

- JS is faster on the server, than anyone has expected. Faster than Java or other options;
- quick requests for data thanks to Node.js - perfect for more dynamic pages
- It's much easier to stick with JavaScript for both sides of the client/server divide than it is to write something once in Java and again in JavaScript, which you would likely need to do if you decided to move business logic you wrote in Java for the server to the browser
- Node.js and JavaScript make it much easier to migrate code
- Queries for some of the newer databases, like CouchDB, are written in JavaScript. Mixing Node.js and CouchDB requires no gear-shifting, let alone any need to remember syntax differences
- Many Web services and databases return data in JSON, a natural part of JavaScript. JSON is part of the foundation of JavaScript. You don't need libraries. It's all there and ready to go
- People love to praise the speed of Node.js. The data comes in and the answers come out like lightning. Node.js doesn't mess around with setting up separate threads with all of the locking headaches. There's no overhead to slow down anything. You write simple code and Node.js takes the right step as quickly as possible
- Once the JavaScript code is in the browser cache, the only thing that moves along the wires is the new data. There's not a ton of HTML markup, and there are no repeated trips to download the entire page. Only the data has changed. If you're willing to put in the time to create a slick browser-side Web app, there's a big payoff. Node.js is optimized to deliver the data and only the data through Web services. If your app is complex and data-rich, it's a good foundation for efficient delivery.
- *Cons*:
    - Node.js is far behind Java considering debugging and maintaining large enterprise applications
    - JS is lacking a solid and reliable tools such as Eclipse or Netbeans in Java
    - JS doesn’t have as good collection of libraries as Java, which can be used for serious work
    - If one Node.js request runs too slowly, everything slows down. There's only one thread in Node.js, and it will get to your event when it's good and ready