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

1. **(Reduceret response tid).** Større virksomheder som har integreret deres systemer med JavaScript, som en backend platform har påstået at deres request pr. sekund er fordoblet og at deres response tid er reduceret med 35%. Virksomheder som PayPal valgte at teste deres mobile platform på en Black Friday og de havde så en 200 millioner online bruger og intet problem opstod. Som vi kan se at ved at bruge JavaScript Frameworks så som Node JS og Express så kan man håndtage rigtige mange request og give meget hurtige reposens tilbage.
2. **(Nemt modificere og opretholde)**. Skrive applikation i Node JS skrives i små moduler. Dette betyder at applikationen bliver udviklet i små dele i stedet for en stor del, på denne måde er det nemmere at ændre i koden, jo flere krav der fremkommer til applikationen.
3. **(Produktivitet).** Node JS programmer er udviklet i JavaScript. Dette betyder at der ikke er nogen barrier mellem front-end og back-end udvikler. Dette betyder dermed at det er nemmere at integrere front-end og back-end hvilket kan resulterer i et godt produkt. Node JS har også en god måde at håndtage pakker på (NPM) (beskriv mere).
4. **Asynchournous:** Mere end 1 proces eller ting køre samtidig. NodeJS er asynkron. Java Script er synkron det vil sige at 1 proces bliver kørt af gangen, tænk på det som om det 1 linje kode af gange der bliver kørt.
5. **Non-Blocking:** Lave andre ting uden at stoppe programmet. Dette er muligt i Node JS fordi det er asynchournous.
6. **Modules:** Det er stykke kode, som ikke kan påvirke noget andet kode ved et uheld. Altså det er et stykke kode som ikke kan påvirke noget andet kode i din applikation, ved et uheld eller af tilfældighed.
7. **Libuv:** Er en c++ bibliotek, og det bruges i NodeJS. Det Libuv gør er at det håndtage event der kommer fra vores operativ system. Libuv connecter til operativ systemet ved at lave requests, det kunne være at downloade noget fra internettet, eller åbne en fil. Libuv inden i har så en Queue, som indeholder events, også har vi også en Event Loop, denne loop vil kører disse events igennem og ved hvert loop vil Libuv tjekke om der er noget i Queue’en Libuv vil så tag det og det vil så kører en callback, det vil sige når vores event er complete jamen så vil callback koden kører i v8, og husk v8 er synkron men hele processen vi har beskrevet er asynkron, og denne proces foregår inden i node.
8. <https://gist.github.com/caike/5588918>
9. <https://www.toptal.com/nodejs/why-the-hell-would-i-use-node-js>

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

**Node.js + Express:**

**Pros:**

1. Fordi Node JS er Asynchronous event driven IO så er den god til at håndtage samtidige requests.
2. Bruger Java Script, som er nemt at lære.
3. Det er den samme stykke kode både for front-end og back-end.
4. NPM måden Node JS håndtage pakker på.
5. Node JS er god streame stor filer.

**Cons:**

1. Callbacks kan være svære at håndtage ved stor applikationer da man ender med at have nested Callbacks.

http://voidcanvas.com/describing-node-js/