```
~/Projects/SovTech
→ npm start -- career
```

Backend Engineer Test (Chuck/Swapi/C#)

Focus	Backend
Last Edited	@Oct 15, 2020 9:15 AM
Ready	☑
Tech Stack	C# Dotnet

At SovTech our development teams are multidisciplinary, and we 💙 C# and dotnet. It saves us time, works incredibly well, and opens up many many doors. We've put this test together to afford Fullstack engineers at any level out there, the freedom to prove their skills 😎

Estimated completion time (skill dependant): **~ 6 - 8 hours**

SovTech.com | Custom Software Development Sorted | USA

SovTech.com is a leading custom software development company. New York and San Francisco. Mobile and Web Apps, Software Development Teams and Blockchain.

https://www.sovtech.com/

⚙️ What is the technical challenge?

We're looking for professionals who are very familiar with the stack we're using.

As this is a Backend engineer test, you are tasked with implementing an OpenAPI (formerly Swagger) compliant web service that abstracts away two downstream APIs; the Chuck Norris API and the Star Wars API.

Your API should do the following:

1. You should develop an OpenAPI compliant API
2. Your API should have 3 root paths:
    1. **/chuck**
    2. **/swapi**
    3. **/search**
3. It should have an endpoint at /**chuck/categories** which returns the result of https://api.chucknorris.io/jokes/categories (all the joke categories)
4. It should have an endpoint at **/swapi/people** which returns the result of https://swapi.dev/api/people/ (all the Star Wars people)
5. Finally, the **/search** endpoint should call both the https://api.chucknorris.io/jokes/search?query={query} and the https://swapi.dev/api/people/?search={query} when queried in order to simultaneously search both the Chuck Norris and Star Wars API. The response should also contain metadata which indicates which API the result belongs to.
6. The **/search** endpoint should accept query data as a query string

👮 What are the requirements?

For this application, you should, at the least:

- An OpenAPI compliant API, you can use any packages or libraries

- Create a C# .NET Core based server (any framework/boilerplate can be used)
- Push your solution to a public Git repository along with detailed instructions on how to bootsrap your service within the README
- Ensure that a link to your submission are emailed to us

Optionally, services such as CodeSandbox, Heroku, AWS, DO etc. may be used to demo your build.

✅ How will you be scored?

This test we feel allows you to express your resourcefulness with many Fullstack elements. It is for this reason we have left it pretty open.

Aspects to note:

- A lot of emphasis will be on your familiarity with modern C# and Dotnet development, tooling and techniques
- Most best-practices and standards should be portrayed in your file organization,
  methods, naming conventions etc.
- Your ability to execute the required task

🚀 Are there any other considerations?

Again, use this test as an opportunity to express yourself. Here are some of the things we use and like - only to keep in mind, of course 😄

- We like Git
- We believe in a DevOps culture
- We believe in quality code (think tests)
- We always consider responsiveness
- We always go above and beyond
- We like all the new GraphQL things (think Prisma, Hasura etc.)

- We value security, strongly
- We like Serverless
- We like containerisation
- We like clean code

🔒 Closing

Thank you for taking the time to complete this test. We look forward to your submission. If you have any feedback, please feel free to share it with us. As we are continuously improving our hiring process.