**Dear candidate,**

Thank you for taking part of the evaluation process to join the Experts Group here at Sela.

At Sela, our hiring process is usually very efficient and fast, and *usually* consists of the following steps:

* Phone interview with one of our wonderful Talent Acquisition specialists
* Getting to know you, and understand what are you looking for currently
* Off-site Candidate Exercise in a relevant field of expertise (you are here)
* Assess some of the relevant skills for the position (more on this below)
* Technical On-site interview with one of our consultants
* Validate the knowledge shown in the Off-site exercise, and see how well you know technology
* Technical & Personal On-site interview with our CTO
* Assessing architecture, and design skills
* Assessing your fit to consulting, training, or development work
* Personal On-site Interview with our Global VP (hiring manager)
* Getting to know you, assessing your fit to the family
* Discussing compensation options

Usually all On-site interviews are being scheduled within a few days, and mostly in the same day, as schedule permits.

**A Note before you start with the exercise:**

This exercise was compiled to help us assess your capabilities around relevant technological knowledge, design skills, as well as the ability to quickly get into new stuff and prototyping skills.

It would be great if you follow the instructions as-is and provide a fully-working and testable solution.

However, we also won’t mind to see you showing-off your abilities and throw in a few tricks of your own (whatever you feel like - i.e implementing this also in GraphQL, hosting it on Kubernetes, Cloud stuff, etc) - But - we do value your time - so it should be noted that it is NOT mandatory, and it will be considered as bonus points and won’t affect your score.

**Full-Stack Exercise:**

This exercise consists of two parts - front-end and back-end, if you are interviewing for a Full-Stack position, you should do both.

**Front-End Part:**

This exercise consists on implementing Single Page Application that works with a backend, based on one of the popular Frameworks\libraries of your choosing - Angular, ReactJS, VueJS.

The Web application should consists of four simple pages:

* Login Page
* Products List Page
* Product Detail Page
* Cart Page

**Login Page**

The login page should be fairly simple, and should provide singe option to the user - Login in by using a hard-coded username and password on the server-side.

Upon login, the server application will return an access-token that will be used to access the back-end endpoints.

After successful login, the user should be redirected to the Products List page.

**Products List Page**

The products list page is the main page of this simple application.

The page should consist a thumbnail products grid, with support for sorting of the list.

The grid should be infinitely-scrollable (without page links) - when the user scrolls the items are added smoothly to the grid for a great customer experience.

There should also be a search box for filtering the grid, with auto-completion mechanism that filter the grid according to the search terms when the user hits ‘enter’.

On clicking a product in the grid, the user should be redirected to the Product detail page.

**Product Detail Page**

The product detail page should display all the available information about the product selected.

The user can either go back to the previous page, or click on “Add to my cart”

**Cart Page**

The cart page shows the list of products that the user chose to add to the cart.

The user can remove items from his cart.

***\* We don’t expect you to be a Ninja UX Expert, but it is a common use-case, so you can get inspiration from similar web-sites. also don’t invest too much time on UI, you can use an existing packages for this (i.e Angular Material).***

**

**Backend Part:**

This exercise consists on implementing a simple backend service that will be consumed by a frontend, or tested using REST HTTP requests.

The backend should be implemented using a popular Frameworks\libraries of your choosing, among the following runtimes - **NodeJS, Python, .Net Core, Java**.

The underlying database should be **MongoDB** for simplicity.

The Backend service should expose the following Http REST routes and functionality:

* /api/products
  + Fetching products
  + Sorting, filtering, and searching using auto-complete
  + Getting a specific product
* /api/users
  + Log-in a user using JWT tokens
* /api/cart
  + Manage a users’ cart

Note:

* You can use any mock products data source - as long as there will be at least 1000 items.

**General guidelines for the exercise:**

* We expect to see great design, modular structuring, and beautiful and easily-understandable code.
  + The code should be self-explanatory - comment the code only if you think it's necessary, after all we do need to understand what we are reading
* We would like to see some unit tests, so you can choose one component and implement Unit Tests only for it.
* The working solution should be easily tested by using Docker (either you provide us with links for the hosted components, or provide a Docker image with a instructions on to run and configure the relevant components).
* The exercise should take a reasonable time (few hours). A complex design may take longer. You are encouraged to compromise in the proper places. But, be sure to note them in your design and implementation.

**Expected Deliverables:**

* Submission should be accompanied by a short document describing your solution, both in design and implementation aspects.
  + Feel free to tell us if you made work-assumptions, or had to workaround an issue.
  + Important: If you are not sure that you have not understand something exactly - that’s fine, do as you understand and mention it in the document. Don’t let anything hold you from finishing the exercise.
* The Code and the document should be archived and sent to your recruiter in a Zip format (sorry Rar and Tar).

**Thanks, and good luck!**

**The team at Sela**