Building a system to manage the product list at tiki.

Type: Individual Test
Length: 4 days.
Total mark: 20
Submission: code as zip file and readme.doc that contains links of deployment if any

Tiki is a top online shopping company in Vietnam. They implement microservices architecture for their system. Product List management is one micro service within the whole Tiki system.

This product list contains millions of products with different categories and sub-categories.

Look at the tiki.vn pages and get a sense of how many attributes a product will need.

Students must use SpringBoot for backend and ReactJS for frontend

Students can make as many assumptions as possible if any information given here is not specific or in detail. Just go ahead with something that you believe is true, something that is obvious, and straightforward. Waiting for confirmation on minor questions may delay your progress.

Students need to do:

1. **(7 marks)** Build a simple backend to manage a list of products and their categories.
Students are free to select attributes that they think are most important.
Students are open to design how to structure categories to meet demands of different kinds of products, i.e. shoe vs smartphone.

2. **(6 marks)** Build a frontend to CRUD on products and categories.
Students are free to select their design to give users best experiences, i.e. friendly, easy to use, aesthetics. This CRUD includes also pagination, filter, search for easy selection of a product.
No authentication/authorization is required.

3. **(5 marks)** Write a quick reflection on a **readme.doc (max 3 pages)** to explain and justify why you design your system that way. For example, you should answer the following questions.

- What technology and architecture did you choose for?
- Why did you select that technology and architecture?

- Why did you define such attributes for your products and organize such categories?
- Whatever arguments to convince your clients, your boss that your system is elegantly designed.
- Etc.

This is very open so you are free to go your own way.

4. **(2 marks)** Deploy both frontend and backend to a cloud service (AWS, Heroku, Digital Ocean, Google Cloud) and provide the links in readme.doc. Some sample data on the database is expected. Don’t just deploy an empty system.