Peer39 exercise

part1

You need to design and implement a service that brings the text of a URL for a list of given Web Pages.

Input: URLs

Output: URL, URL Text

General guidelines

- Implementation must be documented and readable
- Text is read in one pass
- Use java8 implementation guidelines
- You should use gradle for build and deploy
- Bonus: Retrieve only the text from html. Tags should be cleaned but the text between them should remain. Also, text should not include scripts.

Delivery and testing guidelines

- Test your code against various example URLs including the URLs below:
  - http://www.msn.com/en-nz/travel/tripideas/70-of-the-planets-most-breathtaking-sights/ss-AAIUpDp
  - https://www.radiosport.co.nz/sport-news/rugby/accident-or-one-last-dig-eddie-jones-reveals-hansens-next-job/
  - https://www.glamour.de/frisuren/frisurenberatung/haarschnitte
  - https://www.bbc.com
  - https://www3.forbes.com/business/2020-upcoming-hottest-new-vehicles/13/?nowelcome
  - https://www.tvblog.it/post/1681999/valerio-fabrizio-salvatori-gli-inseparabili-chi-sono-pechino-express-2020
  - http://edition.cnn.com/

- Zip the project file and email to pninit.dvir@peer39.com

Part 2

You need to design and implement a system that categorizes web pages based on a keyword category.

A Keyword can contain either single word (one word only) or phrases up to 6 words.

A Keyword Category can contain 1 to 1000 words/phrases.

If a Keyword from a category is found on the page, then the page should be categorized with that category. Match is case insensitive.

You should use the output text generated by the previous part.

For example:

Category “Star Wars” – has keywords “Star wars”, “starwars”, “star war”

Web page: https://www.starwars.com/news/everything-we-know-about-the-mandalorian containsthe text:

Set about five years after the fall of the Empire, before the rise of the First Order, *The Mandalorian* is an exploration of a new era in *Star Wars* storytelling onscreen.

……

As you can see, this web page should be categorized as “Star Wars” because it contains the phrase “STAR WARS”

1. Write models for Category and CategoryKeyword
    a. No database is required – all models should be kept within memory

2. Initialize the models with the following categories and keywords (suggesting to implement Runner.initializeModel)
    a. Category name: Star Wars. Keywords: star war, starwars, starwar, starwars, r2d2, may the force be with you
    b. Category name: Basketball. Keywords: basketball, nba, ncaa, lebron james, john stokton, anthony davis

3. implement a flow that classify URL for matching categories
    a. Mention implementation complexity assuming the text length is N, number of categories is M, max keyword length is K.

4. Implement Runner class
    a. The main method get two parameters: list of categories and list of URLs and print all matching categories. The given list of categories is a sub-set of the predefined categories.
5. **BONUS:** Present the solution above (1-4) using K8s, keep in mind things like (monitoring,scaling deployment etc). Please provide just the design doc.

General guidelines

- Implementation must be documented and readable
- Use java8 implementation guidelines
- You can add more library to build.gradle as needed – not required
- Bonus: write API that get URL/s and return if categories exist
- Bonus: write few implementations with different complexity

Delivery and testing guidelines

- Test your code against various example URLs with the two categories you created
    - http://www.starwars.com
    - https://www.imdb.com/find?q=star+wars&ref_=nv_sr_sm
    - https://edition.cnn.com/sport

- Zip the project file or upload to drive and email to pninit.dvir@peer39.com