DCUBE Coding Exercise: URL Shortener

Through a simple user interface, I want to be able to submit any URL so that it can be shortened and used/shared with the capability of URL management for each user.

As a user, I would like to manage (create/delete) my own set of shortened urls e.g.

- Create a new shortened url
    a. Sample request: www.google.com
    b. Sample response: short.en/abc
- Delete an existing shortened url

Requirements

Backend

- Create a backend API server (of any preferred tech stack, preferably NodeJS, Go) to be able to handle the conversion of a given URL to a shortened version of the URL
- The backend API servers should also handle user account creation and be capable of storing and retrieving user saved URLs.
- Do not use any authentication framework but implement a **username and password** authentication using the database of your choice. Also do apply security best practices when storing and transmitting passwords.
- You may consider using cookies or JWT to manage user sessions.
- Store the converted URL and the user accounts in a database (of your choice, preferably relational DB, NoSQL)

Frontend

- Create a frontend application (of any preferred tech stack. Preferably ReactJS, VueJS, AngularJS, Javascript/Typescript) that allows users to submit a request to convert a URL, and prints the converted URL to the screen
- The frontend should allow users to login and have a panel to only display URLs saved by the current user for URL management (Add/View/Delete only).

Assumptions

1. Different users shortened URLs **must** be different if the request is the same
   e.g.
   a. User 1
      i. Sample request: www.google.com
      ii. Sample response: short.en/abc
   b. User 2
      i. Sample request: www.google.com
      ii. Sample response: short.en/def

Bonus Requirements

- Give the frontend application a unique look! Show us you know how to style a webpage and/or make it mobile responsive.
- Deploy your application(s) to the cloud or any publicly accessible hosting service! E.g. Heroku.
- Persistence of user created urls across system reboots.
  E.g.
  - MySQL / Postgres
  - ElasticSearch / MongoDB
- Writing 1 or 2 unit/functional tests to demonstrate the understanding on how to write automated tests.
- Include a feature to further generate a QR code which can be downloaded in PNG or SVG.

What's Next

- Upload your code in a private git repository (Be mindful of the deadline given in the email, we will observe your commit history)
- Do show the progress of your work with atomic git commits.
- You will be invited to demonstrate the applications(s) you have built and participate in a pair programming exercise to add more features to the application. (Attempt the assignment without assistance from others - it'll show during your pair programming performance. Don't disadvantage yourself!)

Submission

- Please send us a link to your repository when you complete your assessment via email
    - paul_weng@tech.gov.sg, gerald_png@moe.gov.sg, darrel_cai@hive.gov.sg

Deadline

- Please complete the assessment within **5 calendar days** upon receiving this email. Failure to do so will be deemed as withdrawal of the TAP application.

Additional Notes

- For any questions relating to this assessment, you can also reach out to us by email
    - paul_weng@tech.gov.sg, gerald_png@moe.gov.sg, darrel_cai@hive.gov.sg

Important!

We will assess your submission holistically (i.e. not just in terms of functionality), including factors such as:

- Readability and code cleanliness
- Good coding practices such as error handling
- Code structure/design, e.g. modularity, testability