Task Force application challenges

We have prepared the following three challenges for you. We want you to complete as many challenges as you can, but we are looking for at least one completed challenge. Although we have outlined the requirements for each challenge, we'd love to see your creativity, so feel free to add your own touch. Good luck!

1. **Star Wars app (Mobile)**

   Develop an android mobile app that lists star wars characters using this REST API (https://awesome-star-wars-api.herokuapp.com/). The app should:

   - Display a list of characters in the star wars movies
   - Show a character's details when selected from the list. Such details include the character's name, image, birth year, eye color, etc.
   - From a character's details, the user shall be able to see the character's affiliations, masters, and apprentices.
   - Allow the user to save locally characters fetched from API.
   - Allow the user to delete a character that was saved locally.
   - Allow the user to nickname a locally saved character.

   Extra:
   - Implement Localization based on users' phone settings. Eg: If the user's phone language is French, the app's texts should also be in french.

   NOTE:
   - Preferably, use android architecture components with MVVM design pattern or BLOC (Flutter) (https://developer.android.com/jetpack#architecture-components)

- Use native languages (Java{Android},Kotlin{Android}, Swift{iOS}) or the Flutter Framework
- Remember to unit test your implementation

2. Restaurant listing app (Web)

Develop a web application for listing restaurants.
The app should:

- Allow the user to search for restaurants by location (e.g: Kigali, Rwanda).
- Display found restaurants(name, location) from a search.
- Show a restaurant's full details upon selection of the restaurant.
- Show a restaurant's daily menu upon selection of the restaurant.
- Show a restaurant's reviews upon selection of the restaurant.

NOTE:

- Implement the web application per the given mockup using Bootstrap 4. Link to mockups and assets here.
- Feel free to use your preferred frameworks/library(Vue, React, Angular, jQuery, JavaScript, etc...)

Extra:

- Integrate the web application with the Zomato API (https://developers.zomato.com/documentation) to list actual data from the API. With this API, you can search for restaurants by location, query restaurant details such as menus and reviews.

3. Employee management REST API (Back-End)

Create a Restful API and database for an employee management system.

- **Employees management:**

    - A manager should be able to create an employee bypassing the below details to the API (**employee name, national id, phone number, email, date of birth, status**(active, inactive) **and position**(manager, developer, designer, etc...). After creating the employee, the system should send a communication email to the uploaded employees informing, they joined the company with the company name.
      >ROUTE: /employees (REQUEST: POST)
    - A manager should be able to edit, suspend, activate, and delete employee records.
      >Delete Employee ROUTE: /employees/{employee id} (REQUEST: DELETE)
      >Edit Employee ROUTE: /employees/{employee id} (REQUEST: PUT)
      >Activate Employee ROUTE: /employees/{employee id}/activate (REQUEST: PUT)
      >Suspend Employee ROUTE: /employees/{employee id}/suspend (REQUEST: PUT)
- **Search Feature:**
    - A manager should be able to search for an employee based on his position, name, email or phone number.
      >Search ROUTE: /employees/search (REQUEST: POST)
- **Document your API eg: Swagger, Postman**

Extra (optional):

- **Authentication:** A manager should be able to signup and confirm his email after registration.
  The Manager should signup by providing his/her employee name, national id number, phone number, email, date of birth, status and position; the position property should be generated automatically as Manager upon signup.

  The manager should also be able to log in and receive a token, preferably JWT(https://jwt.io) or OAUTH2() for authentication after successful login;

Reset the password by providing an email address to which a password reset link will be sent, afterward, the user should be able to log in with the new password.

- A manager should be able to upload an excel sheet containing a list of his/her employees (employee name, national id number, phone number, email, date of birth, status and position). After uploading the employee list, the system should send a communication email to all the uploaded employees informing them they just joined a company with the company name.
- **System Logs:**
    - The system should record all the manager's activities (login, logout, password reset, profile update etc...) in order to comply with external audit requirements.

Requirements

1. The manager's password should be encrypted.
2. Apply validation (phone number must be a Rwandan number, and email should be validated).
3. The national id should be 16 numbers.
4. Not allowing the registration of an employee who is below 18 years of age.
5. National Id, Email and phone number should be unique.
6. The system should throw an exception if any error occurs.

Submission:

- Create a git repo for each challenge (if you work on more than one), with a README file documenting how to set up, run, and test your app.
- Make sure to explain the features that were achieved as well as those that were not in your README file, screenshots of the mobile challenge are welcomed as well.

- If you have any inquiry, send it via email to ***team@awesomity.rw***, with the subject “**2020 Awesomity Task Force - #Number**”, where #Number represents the challenge number (1, 2, or 3)