SKIPLI

Coding Challenge #2

Introduction

Create an application that includes front-end, a back-end, and database (Firebase). Use the code editor of your choice.

Before you begin, make sure to sign up for free accounts on these services:
firebase (https://firebase.google.com) (for database)
twilio (https://www.twilio.com) or sms api of your choice (ex: sms.to) (for texting)

Please email engineering@skiplinow.com if you have any questions and provide as much details as possible.

Note: we don't condone cheating of any kind. You are welcome to look things up on Google but you can't have anyone else do this coding challenge for you. Our system detects if you copy and pastes code. It also runs your code against our database of solutions to check for similarities.

Front-end

To create the front-end, follow these initial steps

Step 1: Use the Create-React-App (https://github.com/facebook/create-react-app) to create a skeleton React project.

Step 2: Create a form that includes two input fields (phone number and access code). These fields don't have to be fancy, no styling is acceptable.

The idea is for the user to enter their phone number in the first input field and submit. The back-end will generate a random 6-digit access code that is then saved to the provided phone number already stored in the database. Once saved, the access code is sent to the phone number via text message.

Step 3: The customer receives the access code via text and enters it to the second input field. The front-end now needs to validate that access code by calling the back-end. If the code matches with the access code saved in the database, then return a success message to the front end and save the phone number to the front end local storage.

Step 4: The next screen shows a top navigation bar that has a search bar and profile icon to the right. The user types any string in the search bar to search for Github usernames that contain the search string. The results are shown on a table or grid display (your choice). Each table row/grid element must how id, login, avatar_url (profile pic), html_url, public_repos, followers.

Step 5: The Github api only makes the first 1000 results available. Your task is to add pagination which allows the user to select the page # and number of results per page. Please see the back end requirements for how to pass page number and number of results per page to Github API.

Step 6: Allow the user to like a github profile. When the user clicks on the like/heart icon on the profile, the front-end saves the id of the selected github profile to the user's list of liked profile in database (favorite_github_users). The like/heart icon on the selected github profile must be lighted up even after the page is refreshed.

Step 7: When the user clicks on his/her profile icon, display a modal/page that contains their phone number and the list of Github profile that they liked. Since the list of liked profiles only contain the id's of those profile, you will need to use findGithubUserProfile in the BE to get the profile details with that gthub user id

Back-end

You must create an Express backend. Follow this simple tutorial to start one within 5 minutes
(https://medium.com/@onejohi/building-a-simple-rest-api-with-nodejs-and-express-da6273ed7ca9)

You can create as many functions in the back-end as you want, but the back-end must have these functions:

(POST) CreateNewAccessCode
Parameters: phoneNumber
Return: a random 6-digit access code
Other requirement: save this access code to the phoneNumber in the database

(POST) ValidateAccessCode
Parameters: accessCode, phoneNumber
Return: { success: true }
Other requirement: set the access code to empty string once validation is complete

(GET) searchGithubUsers
https://api.github.com/search/users
Parameters: q (search term), page (page number), per_page (results per page)
Return: an array of users with login name that contains the search term

(GET) findGithubUserProfile
https://api.github.com/user/${id}
Parameters: github_user_id
Return: { login: “”, id: “”, avatar_url: “”, html_url: “”, public_repos, followers }

(POST) likeGithubUser
Paramaters: phone_number (phone number of the registered user), github_user_id (id of the github profile the user likes)
Return: 200 code

(GET) getUserProfile
Paramaters: phone_number (phone number of the registered user)
Return: { favorite_github_users: [user_object, user_object, user_object] }

Submission

Please send us a link to the Github repo where you publish your code.

You must include these items in the repo:

1. README file explaining how the project is structured and how to run it.
2. Screenshots of your application