0

Farhan Qureshi

Other

Exported on Jan 31, 2024 4:44 PM

View this report on HackerRank

Score

0% • 0/175

scored in BNYM Fullstack (Java, Angular, REST API) - Senior TEST01 in 2 min 58 sec on 31 Jan 2024 16:38:09 GMT

Candidate Information

Full Name	Farhan Qureshi
Email	mohamedashfaq002@gmail.com
Test Name	BNYM Fullstack (Java, Angular, REST API) - Senior TEST01
Candidate Packet	View
Taken on	Jan 31, 2024, 04:38:09 PM
Time Taken	2 min 58 sec / 90 mins
Work Experience	> 5 years
Invited on	Jan 31, 2024, 04:37:32 PM
Invited by	DIS

Skill distribution

No.	Skill	Percentage
1	Angular Basic	0%
2	REST API Intermediate	0%
3	Java Basic	0%
4	SQL Basic	0%
5	Java Intermediate	0%

Tag distribution

Easy	0%	Angular	0%
Front-End Frameworks	0%	Component State Management	0%
Event Handlers	0%	TypeScript	0%
REST API	0%	Back-End Development	0%
Interviewer Guidelines	0%	OOPS	0%
Java	0%	Polymorphism	0%
SQL	0%	Database	0%
Exception Handling	0%	Core CS	0%
Programming	0%		

Questions

Status	No.	Question	Time Taken	Skill	Score
⊗	1	Angular: Kanban Board Project (Front-end Developer)	1m 11s	Angular Basic	0/50
⊖	2	REST API: Country Populations Approximate Solution	34s	REST ... Interme...	0/50
⊖	3	Car Fueling Coding	16s	Java Basic	0/50
⊖	4	Which command? Multiple Choice	35s	SQL Basic	0/5
⊖	5	Which of the following joins is like an '... Multiple Choice	4s	-	0/5
⊖	6	Which of the following SQL command... Multiple Choice	7s	-	0/5
⊖	7	Identify Runtime Exceptions Multiple Choice	6s	Java Interme...	0/5
⊖	8	Type Promotions Multiple Choice	3s	Java Interme...	0/5

1. Angular: Kanban Board

⊗ Incorrect

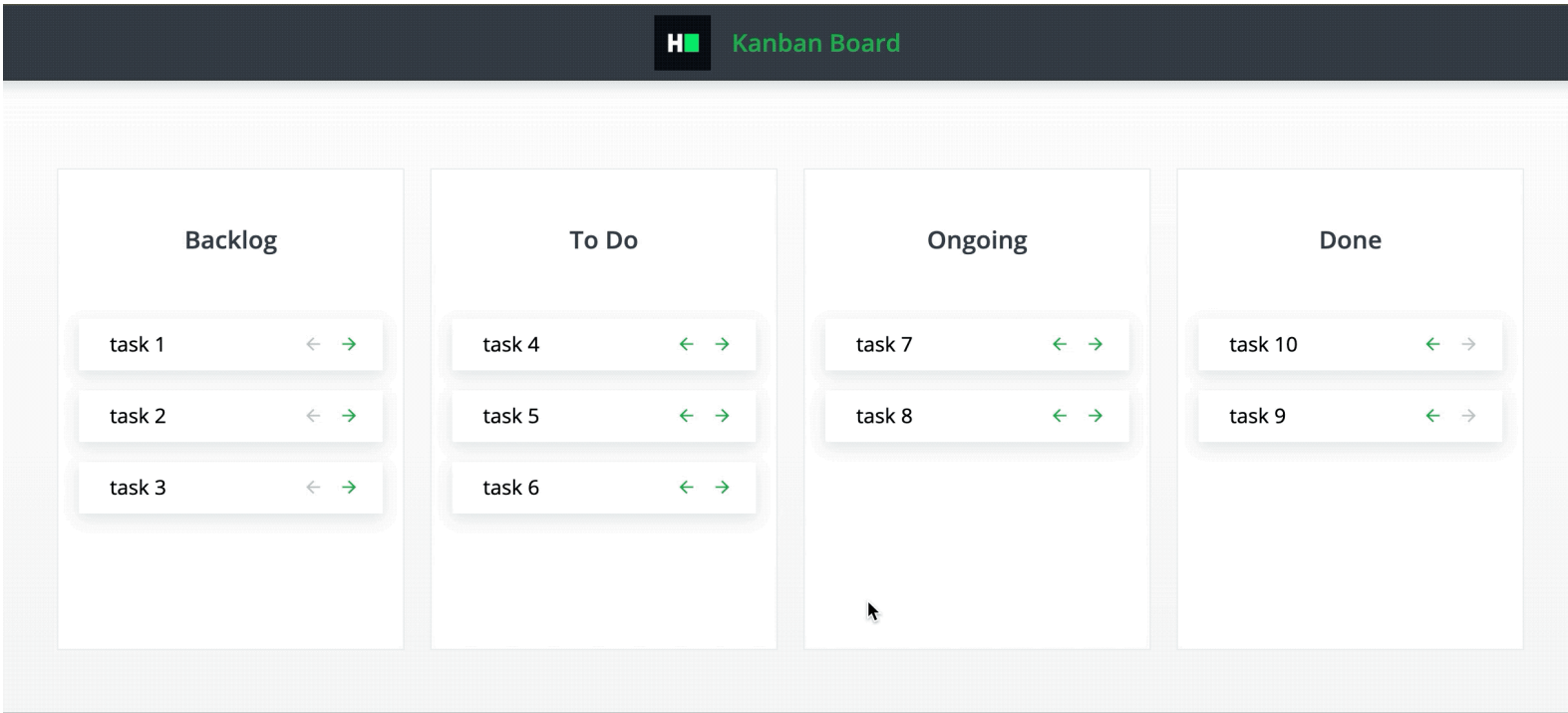
Easy | Angular | Front-End Frameworks | Component State Management | Event Handlers | TypeScript

DESCRIPTION

Kanban is a popular workflow used in task management, project management, issue tracking, and other similar purposes. The workflow is usually visualized using a Kanban Board.

Create a Kanban Board component with tasks, where each task consists of a name only, as shown below:

Hide animation



The component must have the following functionalities:

- The component board contains 4 stages of tasks in the sequence 'Backlog', 'To Do', 'Ongoing', and 'Done'.
- An array of tasks is passed as a prop to the component.
- In every individual stage, the tasks are rendered as a list <ul>, where each task is a single list item <li> that displays the name of the task.
- Each task list item has 2 icon buttons on the right:
    1. Back button: This moves the task to the previous stage in the sequence, if any. This button is disable the task is in the first stage.
    2. Forward button: This moves the task to the next stage in the sequence, if any. This button is disable the task is in the last stage.
- Each task has 2 properties:
    - name: The name of the task. This is the unique identification for every task. [STRING]
    - stage: The stage of the task. [NUMBER] (0 represents the 'Backlog' stage, 1 represents the 'To Do' st 2 represents the 'Ongoing' stage, and 3 represents the 'Done' stage)

The following data-test-id attributes are required in the component for the tests to pass:

- The <ul> for the 'Backlog' stage should have the data-test-id attribute 'stage-0'.
- The <ul> for the 'To Do' stage should have the data-test-id attribute 'stage-1'.
- The <ul> for the 'Ongoing' stage should have the data-test-id attribute 'stage-2'.
- The <ul> for the 'Done' stage should have the data-test-id attribute 'stage-3'.
- Every <li> task should follow these guidelines:
    1. The containing the name should have the data-test-id attribute 'TASK_NAME-name', where TASK_NAME is the name of the task joined by a hyphen symbol. For example, for the task named 'ta 0', it should be 'task-0-name'. For the task named 'abc', it should be 'abc-name'.

2. The back button should have the data-test-id attribute 'TASK_NAME-back', where TASK_NAME is the name of the task joined by a hyphen symbol. For example, for the task named 'task 0', it should be 'task-0-back'. For the task named 'abc', it should be 'abc-back'.
3. The forward button should have the data-test-id attribute 'TASK_NAME-forward', where TASK_NAME the name of the task joined by a hyphen symbol. For example, for the task named 'task 0', it should 'task-0-forward'. For the task named 'abc', it should be 'abc-forward'.

Please note that components have the above data-test-id attributes for test cases and certain classes and ids for rendering purposes. It is advised not to change them.

CANDIDATE SUBMISSION

TESTCASE	TEST FILE	STATUS	SCORE
KanbanBoard For a task in stage 4, can be moved backward till stage 0 and check for buttons are enabled/disabled correctly	unit.xml	Failed	0 / 10.00
KanbanBoard For each task in stage 2 and stage 3, backward and forward buttons are correctly enabled/disabled	unit.xml	Failed	0 / 10.00
KanbanBoard For each task in stage 0 and stage 1, backward and forward buttons are correctly enabled/disabled	unit.xml	Failed	0 / 10.00
KanbanBoard For a task in stage 0, can be moved forward till stage 4 and buttons are correctly enabled and disabled in stage 4	unit.xml	Failed	0 / 10.00
KanbanBoard after many forward and back operations, tasks are in correct state	unit.xml	Failed	0 / 10.00

Review logs: output log

View candidate code

2. REST API: Country Populations

Not Attempted

REST API | Easy | Back-End Development | Interviewer Guidelines

DESCRIPTION

Use the *HTTP GET* method to retrieve information from a database of countries. Query *https://jsonmock.hackerrank.com/api/countries/search?name=s* where *s* is the value of *s*, to find all of the countries that have the substring in their names (not case sensitive). The query response is

paginated and can be further accessed by appending to the query string &*page=num* where *num* is the page number.

The query response from the website is a JSON response with the following fields:
page: current page number
per_page: maximum number of records to display per page
total: total records matching the query
total_pages: the number of pages required to display the matching records
data: a json array with the following fields

- *altSpellings*: Alternative spellings of the country's name
- *area*: The country's area (in km^2)
- *capital*: The name of the country's capital
- *currencies*: The country's official currency
- *latlng*: The capital's latitude and longitude
- *name*: The country's name
- *population*: The country's estimated population
- *timezones*: The timezones that the country spans across
- *topLevelDomain*: The country's top-level domain extension
- *callingCodes*: The international phone calling codes of the country
- *region*: The country's region within a continent
- *subRegion* : The country's sub-region within a continent
- *nativeName*: The country's native or ancient name
- *languages*: A list of languages spoken in the country

Given a substring *s*, and the minimum population *p*, find the number of country names that contain *s* and that have populations greater than *p*.

Note: the search is NOT case sensitive

Function Description
Complete the function *getCountries* in the editor below.

getCountries has the following parameter(s):
string s: the substring to search for
int p: the lower population limit

Returns:
int: the number of countries that meet the criteria

Constraints

- $1 \leq p \leq 10^7$

Note: Please review the header in the code stub to see available libraries for API requests in the selected language. Required libraries can be imported in order to solve the question. Check our full list of supported libraries at https://www.hackerrank.com/environment.

▼ Input Format For Custom Testing

A single string describing the substring to search for, *s*.
An integer describing the lower population limit, *p*.

▼ Sample Case 0

Sample Input

```
STDIN       Function
-----       -----
un      →   s = 'un'
100090  →   p = 100090
```

Sample Output

```
8
```

Explanation

Given *s = 'un'*, the query is *https://jsonmock.hackerrank.com/api/countries/search?name=un* and the response is:


```
[
  {
    "name": "United States Minor Outlying Islands",
    "topLevelDomain": [
      ".us"
    ],
    "alpha2Code": "UM",
    "alpha3Code": "UMI",
    "callingCodes": [
      ""
    ],
    "capital": "",
    "altSpellings": [
      "UM"
    ],
    "relevance": "0",
    "region": "Americas",
    "subregion": "Northern America",
    "translations": {
      "de": "Kleinere Inselbesitzungen der Vereinigten Staaten",
      "es": "Islas Ultramarinas Menores de Estados Unidos",
      "fr": "\u00celes mineures \u00e9loign\u00e9es des \u00c9tats-Unis",
      "ja": "\u5408\u8846\u56fd\u9818\u6709\u5c0f\u96e2\u5cf6",
      "it": "Isole minori esterne degli Stati Uniti d'America"
    },
```

```
    "population": 300,
    "latlng": [

    ],
    "demonym": "American",
    "area": null,
    "gini": null,
    "timezones": [
      "UTC-11:00",
      "UTC-10:00",
      "UTC+12:00"
    ],
    "borders": [

    ],
    "nativeName": "United States Minor Outlying Islands",
    "numericCode": "581",
    "currencies": [
      "USD"
    ],
    "languages": [
      "en"
    ]
  }....
```


The response has *8* countries with populations greater than *p = 100090*.

INTERVIEWER GUIDELINES

▼ Solution

```
import requests

def getCountries(s, p):
    # Write your code here
    page = 0
    pages = 1
    count = 0
    while page < pages:
        page += 1
        result = requests.get(f'https://jsonmock.hackerrank.com/api/countries/search?nam
        obj = result.json()
        if page == 1:
            pages = obj.get('total_pages')
        for country in obj.get('data'):
            if country['population'] > p:
                count += 1
    return count
```

CANDIDATE ANSWER

The candidate has not attempted this question

3. Car Fueling

Not Attempted

OOPS Java Easy Polymorphism

DESCRIPTION

Implement the following two classes:

1. class Car will have the following methods.
* public void topSpeed() prints "Top Speed of the vehicle is 100 kmph" with a newline.
* public void fuelType() prints "Car fuel type is Petrol" with a new line.

2. class SUV extends the class car and implements the following method.
* public void fuelType() and prints "SUV fuel type is Diesel" with a new line, thus overriding the parent class's method.

Once submitted, the hidden *Solution* class will read input and call the methods to test. The Solution class can be viewed by expanding the section at the bottom of the code edit window.

▼ Sample Case 0

Sample Input

```
topSpeed
fuelType
```

Sample Output

```
Top Speed of the vehicle is 100 kmph
Top Speed of the vehicle is 100 kmph
SUV fuel type is Diesel
Car fuel type is Petrol
```

Explanation

The method *topSpeed* was called on a Car object and an SUV object. Next the method *fuelType* was called on an SUV object and a Car object.

CANDIDATE ANSWER

4. Which command? Not Attempted

SQL Easy Database

DESCRIPTION

There are multiple records in a table and some are duplicates. Which command will fetch only one copy of the duplicate records?

CANDIDATE ANSWER

The candidate has not attempted this question

5. Which of the following joins is like an 'inner join'? Not Attempted

SQL Easy Database

DESCRIPTION

Which of the following joins is like an 'inner join'?

CANDIDATE ANSWER

The candidate has not attempted this question

6. Which of the following SQL commands can be used to add data to a database table? Not Attempted

SQL Easy Database

DESCRIPTION

Which of the following SQL commands can be used to add data to a database table?

CANDIDATE ANSWER

The candidate has not attempted this question

7. Identify Runtime Exceptions

Not Attempted

Easy Java Exception Handling Core CS Programming

DESCRIPTION

Which of the following is **not** an example of RuntimeException ?

CANDIDATE ANSWER

The candidate has not attempted this question

8. Type Promotions

Not Attempted

Easy Java

DESCRIPTION

Given the following type declarations:

```
int a;
double b;
float c;
long d;
```

The expression *(a + d) * (c + b)* will be promoted to which of the following?

CANDIDATE ANSWER

The candidate has not attempted this question