ИТ Таланти, Сезон 14 JavaScript

Тест 3

1. Да се създаде SPA application за електронно гласуване “**vervaite-mi**”, който има следните страници:

    a. Вход
    b. Регистрация
    c. Главна страница
    d. Детайли за партия
    e. Резултати
    f. Страница за грешка

2. Страница вход: (10т.)

    На страницата се визуализират две инпут полета - потребителско име и парола, бутон за вход, линк към страницата за регистрация. Ако потребителят сбърка входните данни, да се покаже съобщение за грешка. След успешен вход, потребителят трябва да бъде прехвърлен на главната страница.

3. Станица регистрация: (10т.)

    На станицата се визуализират 3 инпут полета - потребителско име, парола и поле за повторение на паролата. Ако потребителят въведе две различни пароли, бутонът “регистрация” да бъде неактивен и да се изкара съобщение за грешка. Същото трябва да се случи и ако потребителско име не е въведено или при опит за регистрация на потребител с име, което вече е заето.

4. Главна страница:

    На страницата се показват всички политически партии, за които потребителят може да даде своя глас. Всяка една партия представлява картичка (виж дизайна), в която се показват снимка, име на партията, лозунг и два бутона - “гласувай” и “детайли”.(10т.)

    Бутонът “гласувай” трябва да бъде неактивен, ако потребителят вече е дал своя вот. (5т.)

    Бутонът “детайли” трябва да води на станица с подробна информация за съответната партия.

    На станицата има инпут поле, което търси по име на партията. Тази функционалност трябва да бъде имплементирана с дебаунс техника, за да не се създават множество заявки към сървъра. (5т.)

5. Детайли за партия: (15т.)

    На страницата се показва информация за съответната партия (виж дизайна). Снимка, лозунг, имена на председателя на партията, агитационен текст и два

бутона: “гласувай за нас” и “виж резултатите”. Бутонът “гласувай за нас” има сходна функционалност с този от главната страница, като ако е успешен вотът, трябва да се покаже съобщение за успех, а ако не е, трябва да му се покаже съответното съобщение за грешка. Бутонът “виж резултатите” води до страницата с резултати.

6. Страница с резултати: (15т.)
   На страницата се визуализира таблица с две колони. На първия ред от таблицата са съответните заглавия на колоните: “Име на партия” и “гласували в %”. На следващите редове са показани резултатите, сортирани в низходящ ред (партията с най-много гласове е най-горе).
7. Хедър секция: (10т.)
   В хедър секцията има (виж дизайна):
   линк към главната страница,
   съобщение за поздрав: “Здравейте, <потребителско име>! Готови ли сте да дадете своя глас? и
   логаут бутон.
   Хедър секцията е видима само, ако има логнат потребител.

Общи изисквания:

1. Дизайн:
2. Потребител, който не е влязъл в системата да няма достъп до страниците: главна, детайли за партия и резултати (5т.)
3. Ако потребител се опита да достъпи несъществуваща страница да му се покаже страницата за грешка (5т.)
4. Когато потребител влезне в системата, той да остане логнат, дори ако страницата се презареди или браузъра се затвори (5т.)
5. Потребител може да гласува само веднъж
6. За UI библиотека да се използва bootstrap.(5т.)

API документация:

Всички заявки се правят към адрес: https://itt-voting-api.herokuapp.com/

1. Регистрация на потребител:
   a. Метод: POST
   b. url: /users
   c. required headers: ‘Content-Type’ : ‘application/json’
   d. example req body:
      {

```
        "username": "Slavozar",
        "password": "bahur"
}
```

e. possible responses: 400 - когато има такъв потребител, 200 успех, 500 проблем със сървъра

2. Вход на потребител:
    a. Метод: POST
    b. url: /login
    c. required headers: ‘Content-Type’ : ‘application/json’
    d. example req body:

```
{
        "username": "Slavozar",
        "password": "bahur"
}
```

    e. possible responses: 400 - грешни входни данни, 200 успех
    f. response:

```
{
        "sessionId": "633a88217470ea6b429d2b21"
}
```

3. Логаут:
    a. Метод: POST
    b. url: /logout
    c. required headers: ‘Content-Type’ : ‘application/json’
    d. example req body:

```
{
        "id": "<sessionId>"
}
```

    e. possible responses: 400 - грешни входни данни, 200 успех
4. Резултати:
    a. Метод: GET
    b. url: /results
    c. required headers: ‘identity’: <sessionId>
    d. possible responses: 401, 200
    e. response:

```
[
```

```
  {
      "partyId": "6336a570635f11a4d96d0266",
      "voters": 1
  }
]
```


5. Гласувай:
    a. Метод: POST
    b. url: /vote/<partyId>
    c. required headers: ‘identity’: <sessionId>, ‘Content-Type’ : ‘application/json’
    d. possible responses: 401, 400, 406 user tries to vote twice, 200, 500
6. Детайлна информация за партия:
    a. Метод: GET
    b. url: /party/<partyId>
    c. required headers: ‘identity’: <sessionId>
    d. possible responses: 401, 200, 404 - no such party
    e. example response:


```
{
 "_id": "6336a570635f11a4d96d0266",
 "name": "Bahur",
 "slogan": "bahur slogan",
 "picture":
"https://rockymountevents.com/wp-content/uploads/2018/09/birthday_celebra
tion.jpg",
 "leader": "Slavozar Vargulev",
 "agitation": "Glasuvajte za nas!",
 "__v": 0
}
```


7. Търсене на партия:
    a. метод: GET
    b. url: /parties-search
    c. required headers: ‘identity’: <sessionId>, ‘partyName’: <search keyword>,
    d. possible responses: 401, 400 when the keyword is not provided, 200
    e. example response:


```
[
 {
     "name": "Bahur",
```

```
    "slogan": "bahur slogan",
    "picture":
"https://rockymountevents.com/wp-content/uploads/2018/09/birthday_celebra
tion.jpg",
    "id": "6336a570635f11a4d96d0266"
 }
]
```


8. Всички партии:
    a. метод: GET
    b. url: /parties
    c. required headers: ‘identity’: <sessionId>
    d. possible responses: 401, 200
    e. example response:


```
[
 {
    "name": "Bahur",
    "slogan": "bahur slogan",
    "picture":
"https://rockymountevents.com/wp-content/uploads/2018/09/birthday_celebra
tion.jpg",
    "id": "6336a570635f11a4d96d0266"
 }
]
```