Mem serwis

Przy użyciu React zbuduj aplikację wg. poniższej specyfikacji.
Termin oddania: 20.06

Projekt należy wgrać na GitHub , a następnie wysłać prowadzącemu e-mail z linkiem.

rkulinski@kozminski.edu.pl

Zadaniem aplikacji jest wyświetlanie “memów*” i możliwość dawania upvote i downvote.

1. Aplikacja ma zawierać route ‘/hot’ i ‘/regular’
2. Memy z ilością (upvote - downvote > 5) mają trafiać na route ‘/hot’, pozostałe znajdują się na ‘regualr’.
3. Dodaj proste menu, które pozwoli przełączać się między sekcjami.
4. Baza memów ma być stała. Zalecana tablica postaci:

```
[
  {
     title: “Mem 1”,
     upvotes: 6,
     downvotes: 0,
     img: “path/to/image1.png”,
  },
  {
     title: “Mem 2”,
     upvotes: 1,
     downvotes: 2,
     img: “path/to/image2.png”,
  },
  ….
]
```

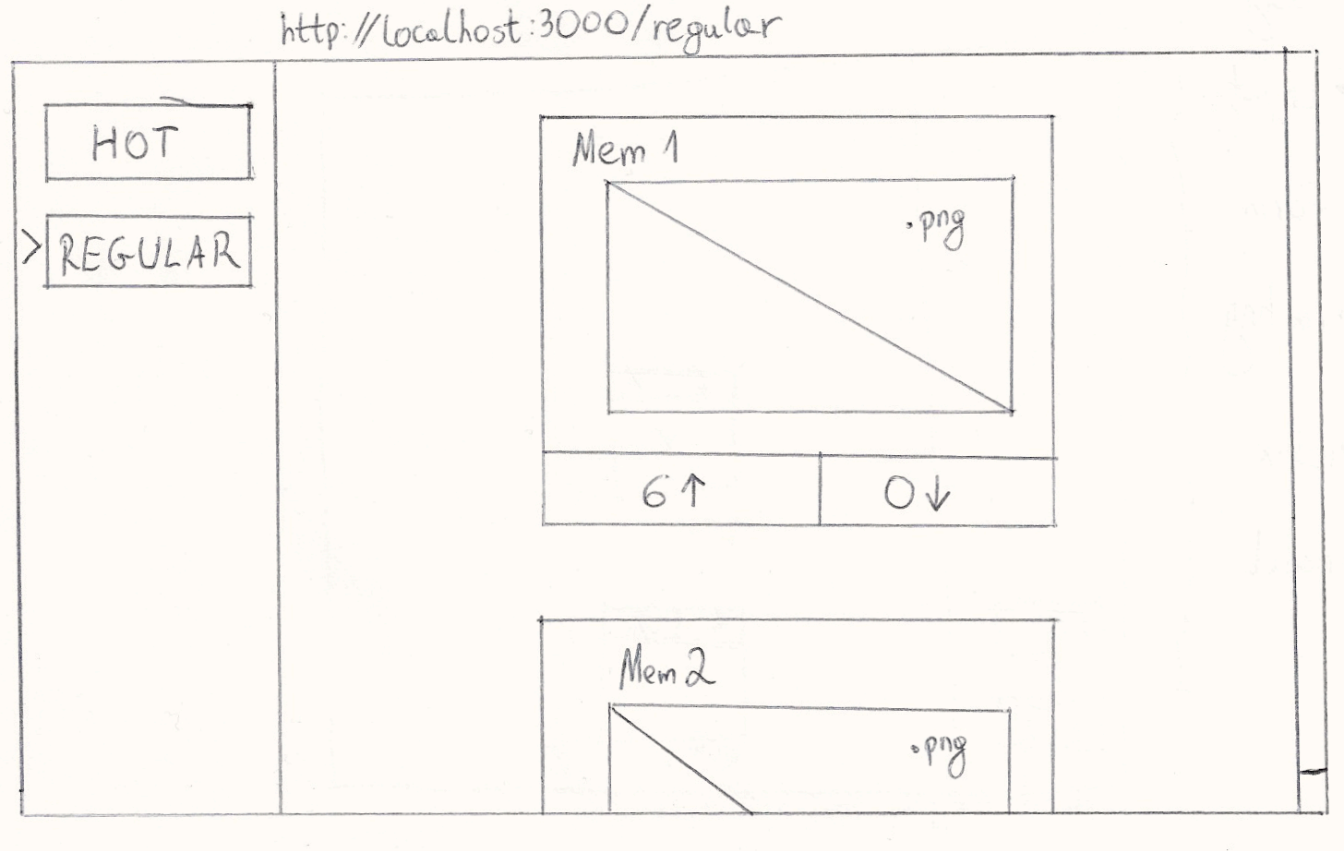
5. Utwórz komponent Mem, który wyświetli tytuł, liczbę upvotes/downvotes i ew. obrazek, oraz kontrolki do kliknięcia upvote, downvote.
6. Wygeneruj listę z komponentu Mem i wyświetl w ‘/hot’ i ‘/regular’
7. Na odpowiednich routach przefiltruj listę z bazą memów zgodnie z zasadami z pkt 2.
8. Filtrowanie powinno działać “live”. Przykład: jeśli kliknę downvote i (upvote - downvote) da 5, mem powinien znikniąć ze listy wyświetlanej na HOT.
9. Jeśli jestem na route ‘/hot’ przycisk “HOT” powinien się odróżniać od “REGULAR” i odwrotnie.
10. Lista memów powinna być przewijalna
11. Oznaczenie mema gwiazdką (wymagane nowe pole w bazie memów).
12**. Dodaj dodatkowy route z formularzem do dodawania mema.

* może to być downolny obrazek (podawany przez link, lub zapisany na dysku i podany jako ścieżka). Opcjonalnie zamiast obrazka można wyświetlić tylko tytuł mema.
** zadanie opcjonalne

Walory dodatkowe:

1. Opis aplikacji i instrukcja uruchomienia w README.md.
2. Kod jest czysty i jednolicie sformatowany.
3. Eleganckie ostylowanie aplikacji.

4. Zastosowanie Redux.



Przykładowy wygląd aplikacji

Tips:

- Bądź kreatywny. Trzymaj się głównych założeń projektu, ale nie bój się własnej inwencji.
- Zalecam dodanie prettier'a.
- Pracuj na GIT. Będzie łatwo śledzić zmiany i w razie problemu cofnąć się do poprzedniej wersji.
- Skorzystaj z biblioteki UI (np. https://material-ui.com/)
- Zacznij prosto. Spróbuj zrobić samą listę memów, bez upvote/downvote z na jednej stronie. Funkcjonalności dodawaj przyrostowo.
- W wersji najprostrzej umieść memy w stanie komponentu, który będzie wyświetlał komponent z Hot i Regular (prawdopodobnie App.js).
- Zamiast filtrowania memów w każdym z komponentów spróbuj rozdzielić memy na dwie tablice (hot, regular) i odpowiednio "przenosić elementy".
  np. setMemes({regular: INITIAL_ARRAY, hot: []});
- W przypadku użycia Redux pamiętaj o nie mutowaniu stanu!
- Zwróć uwagę jak działa funkcja JS (map i filter).
- Wodotryski w postaci stylowania, czy bonusowych zadań zostaw na koniec.