Mem serwis

Przy użyciu React zbuduj aplikację wg poniższej specyfikacji.
Termin oddania: 13.06 do północy

Projekt należy wgrać na GitHub, a następnie wysłać prowadzącemu e-mail z linkiem.
Email: wiktor.jurczyszyn@wsb.wroclaw.pl

Zadaniem aplikacji jest wyświetlanie memów* i możliwość dawania upvote (“łapka w górę”) i downvote (“łapka w dół”).

*memy rozumiane jakoś “śmieszne obrazki”, oczywiście mogą też być nieśmieszne :) W okrojonej, uproszczonej wersji projektu można wyświetlić tylko tytuł mema, ale to nie będzie to samo…

1. Aplikacja ma zawierać route ‘/hot’ i ‘/regular’.
2. Memy z odpowiednią liczbą upvote (np. według wzoru *upvote - downvote > 5*) mają trafiać na route ‘/hot’, pozostałe znajdują się na ‘/regular’.
3. Dodaj proste menu nawigacji, które pozwoli przełączać się między sekcjami (route).
4. Baza memów ma być stała. Zalecana tablica w 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, obrazek oraz kontrolki do kliknięcia upvote, downvote.
6. Wygeneruj listę komponentów *Mem* i wyświetl je 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: jeżeli kliknę downvote i przestanie być spełniony warunek opisany w pkt.2, mem powinien zniknąć z listy wyświetlanej na HOT.
9. Przyciski do przechodzenia między routami powinny pokazywać czy jesteśmy aktualnie na “/hot” czy na “/regular“.
10. Lista memów powinna być przewijalna.
11**. Oznaczenie mema gwiazdką (wymagane nowe pole w bazie memów opisanej w pkt.4
12**. Dodaj dodatkowy route z formularzem do dodawania mema.

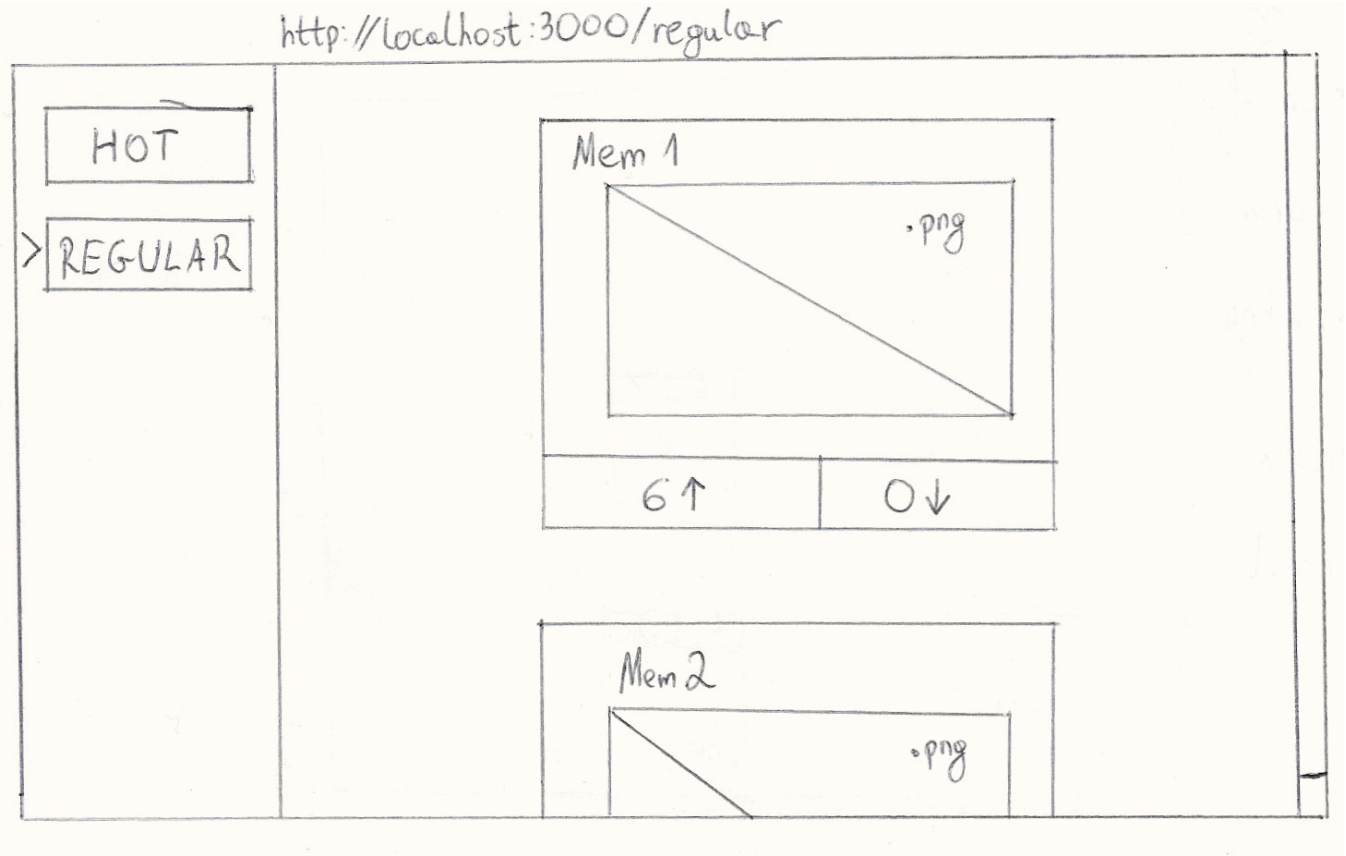
** zadania opcjonalne

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 kreatywna/kreatywny. Trzymaj się głównych założeń projektu, ale nie bój się własnej inwencji.
- Zalecam dodanie prettier'a w VSCode.
- Pracuj na GIT. Będzie łatwo śledzić zmiany i w razie problemu cofnąć się do poprzedniej wersji.
- Możesz skorzystać z wybranej biblioteki UI (np. https://material-ui.com/)
- **Zacznij prosto. Spróbuj zrobić samą listę memów, bez upvote/downvote na jednej stronie. Funkcjonalności dodawaj przyrostowo.**
- W wersji najprostszej 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.
  `this.state({regular: INITIAL_ARRAY, hot: []});`
- W przypadku użycia Redux pamiętaj o nie mutowaniu stanu!
- Zwróć uwagę jak działa funkcja JS (map i filter) oraz pamiętaj o `key={}`.

- Wodotryski w postaci szczegółowego stylowania czy bonusowych zadań zostaw na koniec.