UDACITY

Back to Mobile Web Specialist

Restaurant Reviews: Stage 3

REVIEW

CODE REVIEW 3

HISTORY

client/js/restaurant_review.js 2

```
1 (function registerServiceWorker() {
```

AWESOME

For Future Reference ⚡

A popular technology nowadays to boost web application performance is to implement server-side re

For a start, I think this short article explains it well:

- The Benefits of Server Side Rendering Over Client Side Rendering by Alex Grigoryan

The dev team at Walmart has developed Electrode, a solid framework for server-side rendering:

- Introducing Electrode, an open source release from @WalmartLabs by Alex Grigoryan

If you use React, check out this article on how to implement server-side rendering:

- Quick Start Tutorial: Universal React, with Server Side Rendering by Firas Durri

```
2   if (navigator.serviceWorker) {
3     /*
4     navigator.serviceWorker.register('serviceWorker.min.js')
5       .then(() => {
6         // console.log('Service worker registered successfully.');
7       })
```

```
      .catch(err => {
        console.error('Error registering service worker:', err);
      });
    */
    navigator.serviceWorker.addEventListener('message', event => {
      if (event.data && event.data.id === 'synchronize-data') {
        DBHelper.synchronizeData()
          .then(refresh => {
            if (refresh) {
              initalizeRestaurantReviewPage();
            }
          })
          .catch(err => {
            console.error('Error synchronizing data:', err);
          });
        }
      });
  }
})();

window.addEventListener('load', event => {
  initalizeRestaurantReviewPage();
});

function initalizeRestaurantReviewPage() {
  fetchDataFromURL((error, restaurant, review) => {
    if (error) {
      console.error(error);
      return alert(error);
    } else if (!restaurant) {
      error = 'initalizeRestaurantReviewPage: invalid restaurant object!';
      console.error(error);
      return alert(error);
    } else {
        let isNewReview = false;
        const restaurantTitle = document.getElementById('restaurant-title');
        if (review) {
          restaurantTitle.innerText = `Edit review for '${restaurant.name}'`;
        } else {
          isNewReview = true;
          review = DBHelper.getNewReview(restaurant);
          restaurantTitle.innerText = `Add a new review for '${restaurant.name}'`;
        }
        const inputName = document.querySelector('.review-fields-container #name')
        inputName.value = review.name;
        const inputRating = document.querySelector('.review-fields-container #rati
        inputRating.value = review.rating;
        const inputComments = document.querySelector('.review-fields-container #co
        inputComments.value = review.comments;
        const reviewForm = document.querySelector('#review-form');
        reviewForm.onsubmit = event => {
          submitReview(event, restaurant, review);
        }
        setTimeout(() => {
          if (isNewReview)
            inputName.focus();
          else
            inputComments.focus();
        }, 0);
    }
  });
}

fetchDataFromURL = callback => {
  const restaurantId = UrlHelper.getParameterByName('restaurant_id');
```

```
  if (!restaurantId) {
    error = 'No restaurant id in URL';
    callback(error, null);
  } else {
    DBHelper.fetchRestaurantById(restaurantId, (error, restaurant) => {
      if (error)
        return callback(error);
      if (!restaurant)
        return callback(`Restaurant with id '${restaurantId}' cound not be found!`
      const reviewId = UrlHelper.getParameterByName('review_id');
      if (!reviewId)
        return callback(null, restaurant);
      DBHelper.fetchReviewById(reviewId, (error, review) => {
        if (error)
          return callback(error, restaurant);
        if (!review)
          return callback(`Review with id '${reviewId}' cound not be found!`, rest
        return callback(null, restaurant, review);
      });
    });
  }
};

submitReview = (event, restaurant, review) => {
  event.preventDefault();
  event.stopPropagation();
  const inputName = document.querySelector('.review-fields-container #name');
  const inputRating = document.querySelector('.review-fields-container #rating');
  const inputComments = document.querySelector('.review-fields-container #comments
  review.name = inputName.value;
  review.rating = parseInt(inputRating.value);
  review.comments = inputComments.value;
  review.updatedAt = new Date();
  DBHelper.saveReview(review).then(() => {
    setTimeout(() => {
      const restaurantUrl = DBHelper.urlForRestaurant(restaurant);
      UrlHelper.goToUrl(restaurantUrl);
    }, 500);
  })
  .catch(err => {
    alert(err);
  });
};
```

AWESOME

As you have now completed this Nanodegree, you may (or may not) want to take the Google Mobile W
a few extra knowledge that can help you advance your skills and be more prepared for the examinatic

- Web Workers API - an API to run scripts in the background in a separate thread from the web a
- Prefetching, preloading, prebrowsing - Techniques to improve performance over asset loading

Note: I have taken the test a while ago, so I may not remember all the tech that is involved in the test.
exam, don't panic! There are usually enough time for us to do a bit of extra research to complete the t