Week 3 - Project

Scenario

That page you made last week was *great!* Initech were really impressed, and now they want to see how some “real” data might look. We can get rid of the mock check-in API integration now, as they want us to be storing our own check-ins.

For now, we need to be able to submit a check-in, and display all submitted check-ins on your product page.

We’re going to be using PHP to do this - so don’t forget to rename your file from index.html to index.php! Feel free to put all of this in the same file for now, need to complicate things.

Tasks

1. Create a “check-in” class (☕)

Simply create class named `CheckIn` which has properties for the following:

- Name
- Rating
- Review
- Timestamp

2. Display a form in a popup when clicking on the “Check In” button (☕☕)

When clicking the button, a Bootstrap modal should be displayed. It should have a field for each of the properties on the `Review` class, *except* the timestamp. The form should submit to the current page, and the submitted check-in data shouldn’t appear in the URL.

3. Hydrate the check-in class from the form data (☕)

Create a new instance of your `CheckIn` class, and assign the values to it from the submitted data. The timestamp will need to be store the current date/time upon submission.

4. Save all check-in data to a single file (☕☕☕)

If the file “check-ins.txt” doesn’t exist, your code will need to create it, and should contain a serialised array of `CheckIn` instances.

If the file *does* exist, load all data from it, append your newly hydrated instance to the array, and then overwrite the file.

Hint:

Load all the contents of the file to a string, and then unserialise it. Update your array as needed, and then serialise and write it all back.

5. Display all check-in data on the page (☕☕)

Using the array of `CheckIn` instances from the previous task, loop through and display them all on the product page.

Hint:

Use PHP’s templating syntax:

```php
<?php foreach ($checkIns as $checkIn): ?>
  <!-- ... -->
<?php endforeach; ?>
```

6. Display a “success” message (☕)

Upon a successful submission, display a Bootstrap “success” alert notifying the user that their review was successfully saved.

7. Stretch Goal: Add validation to the check-in form (☕☕)

Add some constraints to the check-in values. Be reasonable in your values, but consider:

- Maximum length for the user’s name
- Making user’s name a required field, and not saving if it’s missing

- Min/max values for ratings
- Maximum length for review
- Prevent any XSS attempts
- Perhaps allow users to use some basic HTML tags, such as `strong` and `em`?

If validation fails, show a Bootstrap “danger” alert.

Useful Reading

- Bootstrap Modals - Bootstrap documentation on modal popups. Pay special attention to the “Live Demo” section
- Bootstrap Alerts - Bootstrap documentation on alert boxes
- time - PHP documentation on getting the current timestamp
- fileputcontents - PHP documentation on writing content to a file from a string
- filegetcontents - PHP documentation on reading the contents of a file to a string
- serialize - PHP documentation on serialising data to a string
- unserialize - PHP documentation on unserialising data from a string
- filter_var - PHP documentation on input filtering
- strip_tags - PHP documentation on removing HTML tags from a string