Non-functional Testing

Non-Functional Testing During Development

Arthur Clarkson

4th September 2024

Error handling was outside of the functional specification, and is brought down to the developer to implement and test - meaning that it had to be non-functionally tested. I tried multiple UI components to show the exception returned from the server, such as an banners at the top of the holiday entry form, popups, and toast notifications.

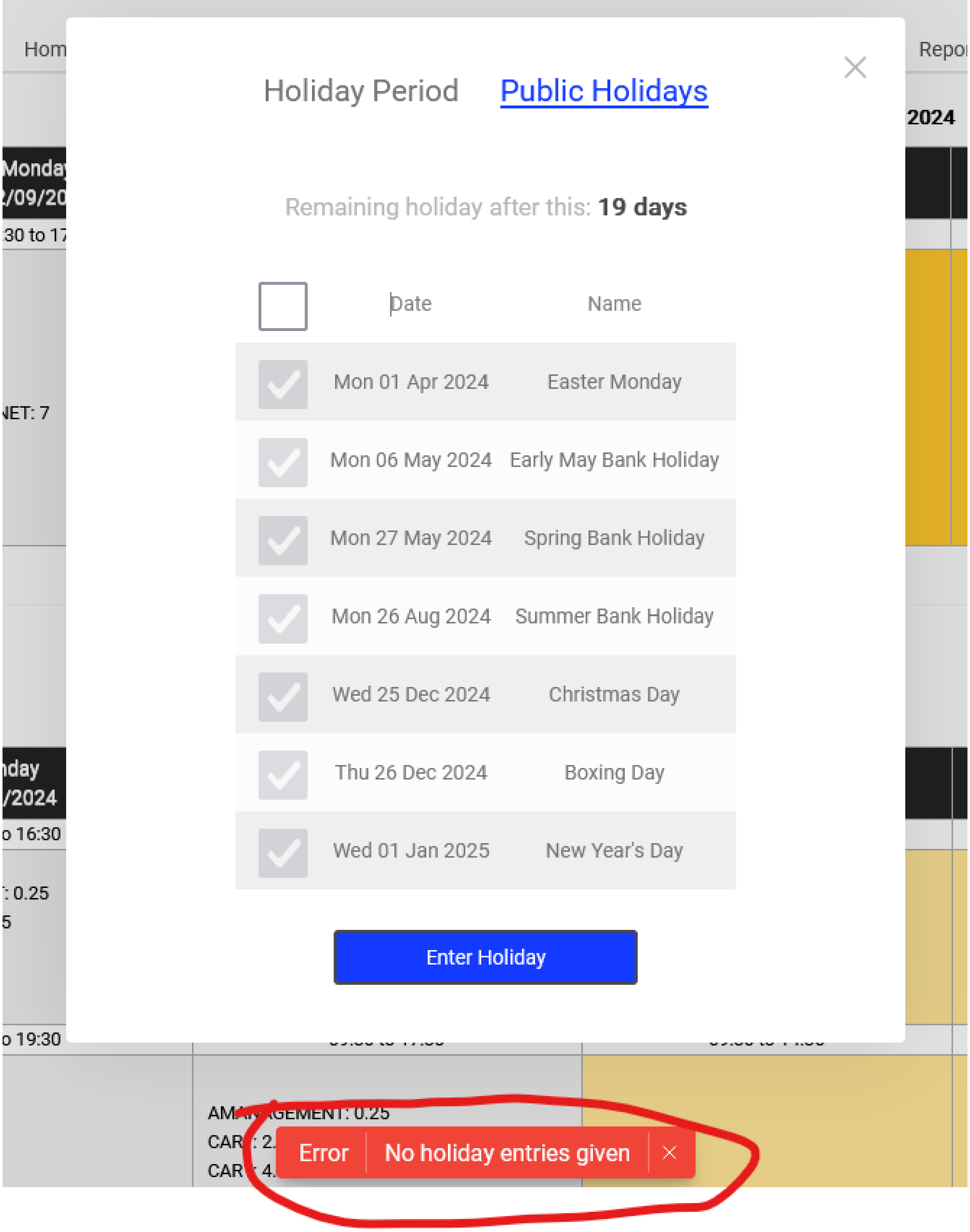
Toast notifications were the best pick, due to them being more visible than a banner, and not being as obtrusive as popups. The API returns an exception if the user gives no days to add as holiday, this is caused by this bit of code:

```
public void CreateHolidayEntriesForUser(List<HolidayEntryItem> holidayEntries, int userId)
{
        // Start of CreateHolidayEntriesForUser...

        bool noHolidayEntires = holidayEntries.Count == 0;
        if (noHolidayEntires)
        {
                throw new Exception("No holiday entries given");
        }

        // Rest of CreateHolidayEntriesForUser...
}
```

To test that the toast notifications are working, I pressed the "Enter Holiday" without any holidays selected:



Remaining holiday after this: **19 days**

□	Date	Name
☑	Mon 01 Apr 2024	Easter Monday
☑	Mon 06 May 2024	Early May Bank Holiday
☑	Mon 27 May 2024	Spring Bank Holiday
☑	Mon 26 Aug 2024	Summer Bank Holiday
☑	Wed 25 Dec 2024	Christmas Day
☑	Thu 26 Dec 2024	Boxing Day
☑	Wed 01 Jan 2025	New Year's Day

Error | No holiday entries given ×

With the toast notification being displayed with the exception message thrown in the API's business method, this non-functional test is considered a success.