Reminder

Today, we will be building a website for Torquay ('Tor-kee') Towers, a hotel in the English countryside.

The site will have two interfaces (besides the built-in Admin interface):

1. one for visitors,
2. and one for hotel staff.

Visitors' Interface

- Display information about the hotel (location, images, description, what's in the surrounding area, etc.)
- View vacancies (dates when rooms are available)
- Book a stay at the hotel
- Fill in a form requesting more information
- Leave a review of the hotel

Staff Interface

- View bookings and vacancies
- Add, modify, and cancel bookings
- Read messages from guests requesting more information
- Delete reviews (We only want guests with a touch of class…)

Project – Part 2

Lets continue building our website for Torquay ('Tor-kee') Towers, a hotel in the English countyside. In Part 1 we worked on the visitors' interface; in this part we will work on the staff interface.

Staff Interface

- View bookings and vacancies
- Add, modify, and cancel bookings
- Read messages from guests requesting more information
- Delete reviews (We only want guests with a touch of class…)
- NB: every view function for this app/interface must require a valid staff member login! A regular user login is not enough (We don't want our visitors to be able to access our staff pages…)
- Be sure to check whether the logged-in user has the relevant permission. Read these pages in the Django documentation to find out more:
    - Permissions and Authorization
    - The permission_required decorator

Tasks

1. Create "staff" app skeleton.
    1. This will manage the Staff Interface, described above.
    2. Decide on a URL prefix for this app.
    3. Register the app and its urlconfig in the project.

2. Staff: create "List Bookings" page
    1. Make a user-friendly page which shows a calendar view of all bookings for the current month. Clicking on a specific booking should take you to a page that shows all the information about the booking. (Make this a dummy link for now).
    2. Add a dummy link to the staff version of the 'make booking' page.
    3. Bonus: use pagination to display only 10/30/50 bookings per page.

3. Staff: create "View Booking" page
    1. Create a page which will display a booking, given its ID (primary key).

    2. Add links to:
        1. Delete the booking (BONUS: add a confirmation dialog) – you will need to implement this URL path too.
        2. Add a new booking (dummy link for now)
        3. Go (back) to the bookings list page (Task #2 above)

4. Staff: Create "Make Booking" page
    1. This is a staff page – you are not making the booking for the currently logged-in user! Rather, as a staff member you will select a guest user, select

a room, and make the booking.

5. Staff: Create “View Messages” page.
    1. Display a list of messages received from visitors on the ‘Info Request’ page of the Visitors app.
    2. Display contact details. Style the email address as a hyperlink to enable easy communication.
    3. Display a list of previous visits by the user – if the message was submitted by a logged-in user. (You might need to make changes to the relevant view function in the visitors app to record this information.)

6. Staff: Allow modification of a booking
    1. All fields should be editable (Hint: if you already have a form class for the booking, this should be pretty easy to implement.)

7. Staff: Create “View Reviews” page
    1. Display a list of reviews of the hotel, as submitted by visitors
    2. Allow deletion