Routing

My apologies for the oversight. In Frappe, the `docstatus` field is used to determine the state of a document. A `docstatus` of `0` means the document is in Draft, and a `docstatus` of `1` means it is Submitted. Here are the corrected templates and context builders for the two portal pages:

For Submitted Orders (`submitted_orders.html`):

```
{% extends "templates/web.html" %}

{% block title %}Submitted Orders{% endblock %}

{% block page_content %}
<div class="container mt-5">
  <div class="row">
    <div class="col-12">
      <h2 class="mb-4">Submitted Orders</h2>
      <table class="table">
        <thead>
          <tr>
            <th scope="col">Order ID</th>
            <th scope="col">Order Date</th>
          </tr>
        </thead>
        <tbody>
          {% for order in orders %}
          <tr>
            <td>{{ order.name }}</td>
            <td>{{ order.order_date }}</td>
          </tr>
          {% endfor %}
        </tbody>
      </table>
      {% if not orders %}
      <div class="alert alert-warning" role="alert">
        No submitted orders found.
      </div>
      {% endif %}
    </div>
  </div>
</div>
{% endblock %}
```

Python Context Builder (`submitted_orders.py`):

```
import frappe

def get_context(context):
    context.show_sidebar = True
    context.orders = frappe.get_all(
        'SHQ Order',
        filters={'docstatus': 1},
        fields=['name', 'order_date']
    )
```

For Draft Orders (`draft_orders.html`), you would just need to change the title block's content to "Draft Orders" and use the same template structure. The Python context file would be almost identical, with the `docstatus` filter changed to `0`.

Please ensure that these HTML files are stored in the `www` directory of your custom Frappe application, and the Python context files should be named accordingly and placed in the same directory.

The `website_route_rules` in your `hooks.py` would look like this:

```
website_route_rules = [
    {"from_route": "/submitted-orders", "to_route": "submitted_orders"},
    {"from_route": "/draft-orders", "to_route": "draft_orders"},
]
```

Remember to reload or restart your bench for the changes to take effect. This way, you will have two different portal pages for orders in the Submitted and Draft states without any links to view details, as per your latest requirements.

Yes, if you want the URL to be different from the filename of the template you're rendering, you would need to use both `website_route_rules` and `standard_portal_menu_items`. Here's how you can do it:

1. **`website_route_rules`:** Define the custom routing. This tells Frappe that when a user navigates to a certain URL, it should render a different HTML or Python file than what the URL suggests.

```
# Example of website_route_rules in hooks.py

website_route_rules = [
    {"from_route": "/submitted-orders", "to_route": "order_list_submitted"},
    {"from_route": "/draft-orders", "to_route": "order_list_draft"},
]
```

In this example, when a user goes to `/submitted-orders`, Frappe will render the `order_list_submitted.html` or `order_list_submitted.py` file. Likewise, navigating to `/draft-orders` will render `order_list_draft.html` or `order_list_draft.py`.

1. **`standard_portal_menu_items`:** Add the links to the portal's navigation menu. This doesn't control routing but simply adds entries to the portal menu with the given titles and routes.

```
# Example of standard_portal_menu_items in hooks.py

standard_portal_menu_items = [
    # ... other menu items ...
    {
        "title": "Submitted Orders",
        "route": "/submitted-orders",
        "reference_doctype": "SHQ Order",
        "role": "SHQ User"
    },
    {
        "title": "Draft Orders",
        "route": "/draft-orders",
        "reference_doctype": "SHQ Order",
        "role": "SHQ User"
    },
]
```

Here, two new items with titles "Submitted Orders" and "Draft Orders" are added to the sidebar. They will show up for users with the role "SHQ User". The `route` tells the browser where to go when the link is clicked, and the `reference_doctype` is for context about which type of document is associated with the link (this is used for permissions and other Frappe internal logic).

Remember, for this to work properly:

- The routes defined in `website_route_rules` must match the `route` you specify in `standard_portal_menu_items`.
- You need to have the corresponding HTML or Python files (`order_list_submitted` and `order_list_draft`) in your application's `www` folder or the appropriate module's `www` folder.