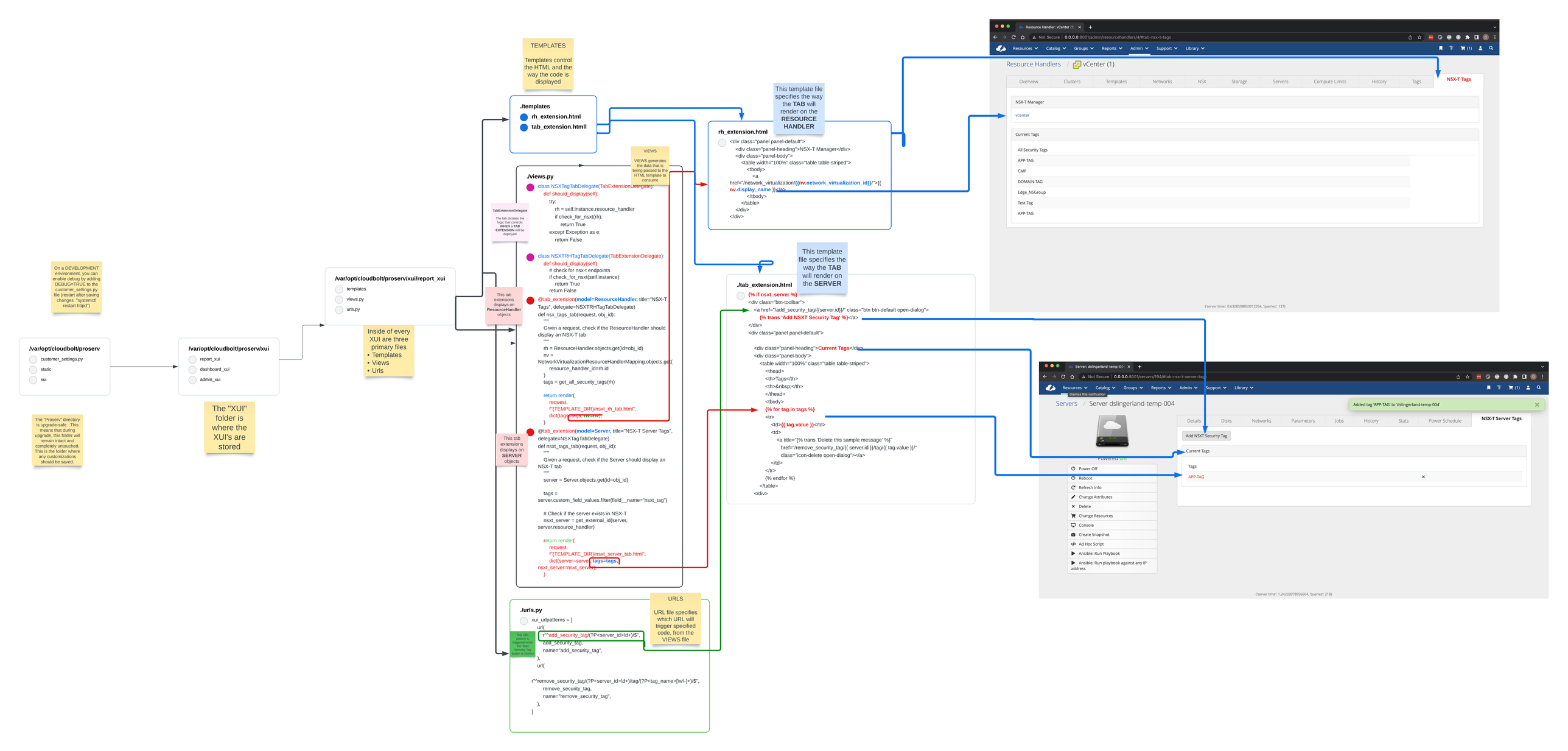
TEMPLATES
Templates control the HTML and the way the code is displayed
./templates
rh_extension.html
tab_extension.htmlll
This template file specifies the way the TAB will render on the RESOURCE HANDLER
rh_extension.html

NSX-T Manager

<table width="100%" class="table table-striped">
<tbody>
<a
href="/network_virtualization/{{nv.network_virtualization_id}}/">{{
nv.display_name }}
</tbody>
</table>

VIEWS
VIEWS generates the data that is being passed to the HTML template to consume
./views.py
class NSXTagTabDelegate(TabExtensionDelegate):
def should_display(self):
try:
rh = self.instance.resource_handler
if check_for_nsxt(rh):
return True
except Exception as e:
return False
TabExtensionDelegate
The tab dictates the logic that controls WHEN a TAB EXTENSION will be deployed
class NSXTRHTagTabDelegate(TabExtensionDelegate):
def should_display(self):
# check for nsx-t endpoints
if check_for_nsxt(self.instance):
return True
return False
@tab_extension(model=ResourceHandler, title="NSX-T Tags", delegate=NSXTRHTagTabDelegate)
def nsx_tags_tab(request, obj_id):
"""
Given a request, check if the ResourceHandler should display an NSX-T tab
"""
rh = ResourceHandler.objects.get(id=obj_id)
nv = NetworkVirtualizationResourceHandlerMapping.objects.get(
resource_handler_id=rh.id
)
tags = get_all_security_tags(rh)
return render(
request,
f"{TEMPLATE_DIR}/nsxt_rh_tab.html",
dict(tags=tags, nv=nv),
)
This tab extensions displays on ResourceHandler objects
@tab_extension(model=Server, title="NSX-T Server Tags", delegate=NSXTagTabDelegate)
def nsxt_tags_tab(request, obj_id):
"""
Given a request, check if the Server should display an NSX-T tab
"""
server = Server.objects.get(id=obj_id)
tags = server.custom_field_values.filter(field__name="nsxt_tag")
# Check if the server exists in NSX-T
nsxt_server = get_external_id(server, server.resource_handler)
return render(
request,
f"{TEMPLATE_DIR}/nsxt_server_tab.html",
dict(server=server, tags=tags, nsxt_server=nsxt_server),
)
This tab extensions displays on SERVER objects
This template file specifies the way the TAB will render on the SERVER
./tab_extension.html
{% if nsxt_server %}

{% trans 'Add NSXT Security Tag' %}

Current Tags

<table width="100%" class="table table-striped">
<thead>
<th>Tags</th>
<th> </th>
</thead>
<tbody>
{% for tag in tags %}
<tr>
<td>{{ tag.value }}</td>
<td>
<a title="{% trans 'Delete this sample message' %}"
href="/remove_security_tag/{{ server.id }}/tag/{{ tag.value }}/"
class="icon-delete open-dialog">
</td>
</tr>
{% endfor %}
</table>

./urls.py
URLS
URL file specifies which URL will trigger specified code, from the VIEWS file
xui_urlpatterns = [
url(
r"^add_security_tag/(?P<server_id>\d+)/$",
add_security_tag,
name="add_security_tag",
),
url(
r"^remove_security_tag/(?P<server_id>\d+)/tag/(?P<tag_name>[\w\-]+)/$",
remove_security_tag,
name="remove_security_tag",
),
]
This URL pattern is triggered when the "Add Security Tag" button is clicked
/var/opt/cloudbolt/proserv/xui/report_xui
templates
views.py
urls.py
Inside of every XUI are three primary files
• Templates
• Views
• Urls
On a DEVELOPMENT environment, you can enable debug by adding DEBUG=TRUE to the customer_settings.py file (restart after saving changes "systemctl restart httpd")
/var/opt/cloudbolt/proserv
customer_settings.py
static
xui
/var/opt/cloudbolt/proserv/xui
report_xui
dashboard_xui
admin_xui
The "XUI" folder is where the XUI's are stored
The "Proserv" directory is upgrade-safe. This means that during upgrade, this folder will remain intact and completely untouched. This is the folder where any customizations should be saved.
Resource Handlers / vCenter (1)
Overview
Clusters
Templates
Networks
NSX
Storage
Servers
Compute Limits
History
Tags
NSX-T Tags
NSX-T Manager
vcenter
Current Tags
All Security Tags
APP-TAG
CMP
DOMAIN-TAG
Edge_NSGroup
Test-Tag
APP-TAG
Servers / Server dslingerland-temp-004
Added tag 'APP-TAG' to 'dslingerland-temp-004'
Details
Disks
Networks
Parameters
Jobs
History
Stats
Power Schedule
NSX-T Server Tags
Add NSXT Security Tag
Current Tags
Tags
APP-TAG
Power Off
Reboot
Refresh Info
Change Attributes
Delete
Change Resources
Console
Create Snapshot
Ad Hoc Script
Ansible: Run Playbook
Ansible: Run playbook against any IP address