Instructions

The next step to the recruitment process is technical evaluation. To qualify for a technical interview, you must upload your take on this practical exercise to your Github account. If you don't have a Github account, please sign-up for one and push your code to it.

When done, kindly reply to this e-mail together with the link to your Github repository of your code where we can clone it. You have 72 hours from the receipt of this e-mail to finish the exercise and to push your code to Github.

Requirements

You must demonstrate your competence as a full-stack developer who is confident working on both client-side and server-side code. Using the programming language and web application development framework of your choice, use an empty starter development template. If the default template includes any default code (like controllers and views) please delete them.

Basically, there are two pages that you'll need to implement:

- A login page, that consumes an external API to validate the account credentials from a standard HTML form submission
- A home page, that is only accessible when a valid user is logged in. If the user attempts to go to this page directly without logging in, the user must be redirected to the login page.

Objectives

- Authenticate users with a login page by consuming an external JSON API endpoint from server side.
- Restrict unauthenticated users from accessing the home page
- The home page must show a hierarchical tree of regions, cities and barangays, data can be pulled from an external JSON API.

External Dependencies

Aside from your framework-specific server-side (and possibly client-side) dependencies, you'll need to consume an external JSON API endpoint to accomplish the objectives.

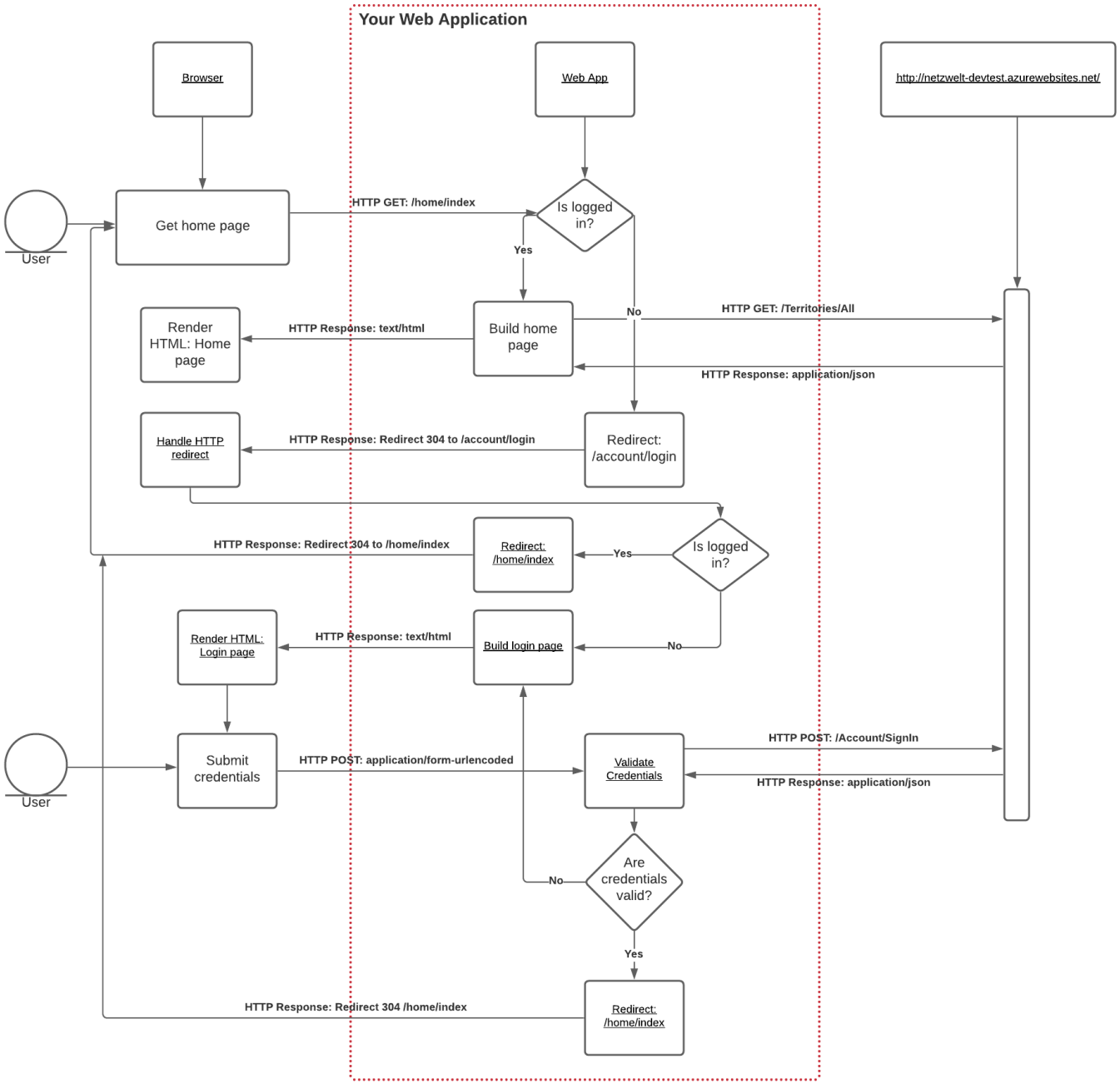
Basically, there are two API endpoints on https://netzwelt-devtest.azurewebsites.net :

URL Path	Method	Remarks
/Account/SignIn	POST	Returns the authenticated user's username and set of roles
/Territories/All	GET	Returns a list of territories that need to be presented in a hierarchical tree structure

API usage documentation can be viewed online: https://netzwelt-devtest.azurewebsites.net/swagger/index.html

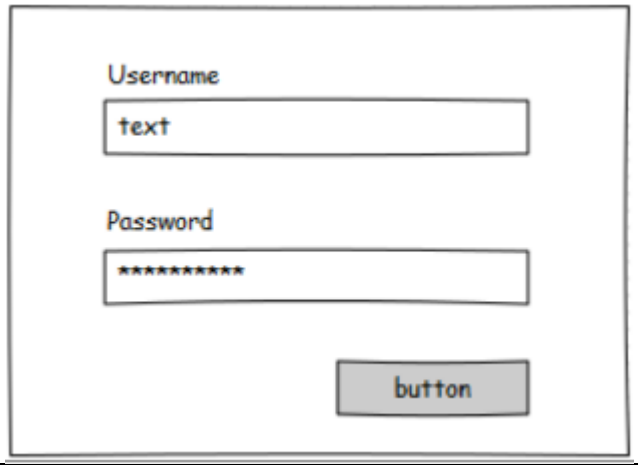
High-level Interaction Diagram

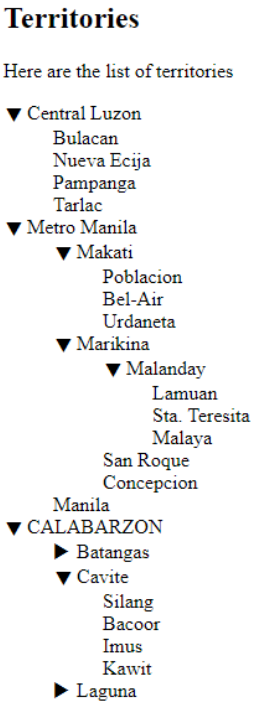
Please refer to the diagram below to see how your application is expected to interact with the user and the API endpoints that need to be consumed to achieve the objectives:



netzwelt

Use Cases

Use-case Name	Login
Actor	Unauthenticated User
Pre-condition	User is not logged in
Expected URL	GET http://your.webserver.host/account/login POST http://your.webserver.host/account/login
Story Details	As a user, I should be able to see the login screen where I can enter my account credentials to be authenticated by the system. The login screen should be able to accept a username and a password.
Exceptions	E1: Invalid credentials provided The login page must be shown again with the error message: "Invalid username or password" E2: User is already logged in The user must be redirected to the home page (/home/index)
UI Mockup	
Technical Requirements	Incoming credentials must be validated on the server side by consuming the API on https://netzwelt-devtest.azurewebsites.net/Account/SignIn You may use the following test credentials: U: foo P: bar Refer to the API endpoint documentation for more details.

Use-case Name	Home Page
Actor	Authenticated User
Pre-condition	User is logged in
Expected URL	GET http://your.webserver.host/home/index -or- GET http://your.webserver.host/ This is the default route.
Story Details	As an authenticated user, I should be able to see a home page that will show a user-friendly hierarchical list of territories.
Exceptions	E1: User attempts to go directly to the home page The user must be redirected back to the login page (/Account/Login)
UI Mockup	
Technical Requirements	The list of credentials can be fetched from an external API endpoint: HTTP GET https://netzwelt-devtest.azurewebsites.net/Territories/All The API returns a flat list of territories. The application must be able to take the list and arrange it using the records' correlation between the fields "id" and "parent." Note that the depth of the hierarchy is unknown at runtime and the list is unsorted. For example: some nodes in the tree may have 10 levels, while others just 1 level. Refer to the API endpoint documentation for more details.

netzwelt

Sample code listing: Simple Treeview in HTML, CSS and Javascript

To show the data in hierarchical form, it must be presented in a tree-like structure. There are many ways to achieve this but here's a very rudimentary approach for reference:

```
<!DOCTYPE html>
<html>
<head>
<meta name="viewport" content="width=device-width, initial-scale=1">
<style>
ul, #myUL {
  list-style-type: none;
}

#myUL {
  margin: 0;
  padding: 0;
}

.caret {
  cursor: pointer;
  -webkit-user-select: none; /* Safari 3.1+ */
  -moz-user-select: none; /* Firefox 2+ */
  -ms-user-select: none; /* IE 10+ */
  user-select: none;
}

.caret::before {
  content: "\25B6";
  color: black;
  display: inline-block;
  margin-right: 6px;
}

.caret-down::before {
  -ms-transform: rotate(90deg); /* IE 9 */
  -webkit-transform: rotate(90deg); /* Safari */'
  transform: rotate(90deg);
}

.nested {
  display: none;
}

.active {
  display: block;
}
</style>
</head>
<body>

<h2>Territories</h2>
<p>Here are the list of territories</p>

<ul id="myUL">
        <li>
        <span class="caret">Central Luzon</span>
        <ul class="nested">
            <li>Bulacan</li>
            <li>Nueva Ecija</li>
            <li>Pampanga</li>
            <li>Tarlac</li>
        </ul>
    </li>
        <li>
         <span class="caret">Metro Manila</span>
        <ul class="nested">
                <li>
                <span class="caret">Makati</span>
                <ul class="nested">
                    <li>Poblacion</li>
```

```
                    <li>Bel-Air</li>
                    <li>Urdaneta</li>
                </ul>
            </li>
            <li>
                <span class="caret">Marikina</span>
                <ul class="nested">
                        <li>
                        <span class="caret">Malanday</span>
                        <ul class="nested">
                            <li>Lamuan</li>
                            <li>Sta. Teresita</li>
                            <li>Malaya</li>
                        </ul>
                    </li>
                    <li>San Roque</li>
                    <li>Concepcion</li>
                </ul>
            </li>
            <li>Manila</li>
        </ul>
    </li>
    <li>
        <span class="caret">CALABARZON</span>
        <ul class="nested">
                <li>
                <span class="caret">Batangas</span>
                <ul class="nested">
                    <li>Lipa</li>
                    <li>Bauan</li>
                    <li>Sto. Tomas</li>
                </ul>
            </li>
            <li>
                <span class="caret">Cavite</span>
                <ul class="nested">
                    <li>Silang</li>
                    <li>Bacoor</li>
                    <li>Imus</li>
                    <li>Kawit</li>
                </ul>
            </li>
            <li>
                <span class="caret">Laguna</span>
                <ul class="nested">
                    <li>Calamba</li>
                    <li>Sta. Rosa</li>
                    <li>San Pedro</li>
                </ul>
             </li>
        </ul>
    </li>
</ul>

<script>
var toggler = document.getElementsByClassName("caret");
var i;

for (i = 0; i < toggler.length; i++) {
  toggler[i].addEventListener("click", function() {
    this.parentElement.querySelector(".nested").classList.toggle("active");
    this.classList.toggle("caret-down");
  });
}
</script>

</body>
</html>
```

Selection Criteria

We are very interested to see how often you commit code. We suggest you commit often so that we can see the code history.

Plus points if:

- You used a Javascript frontend framework (like Vue, ReactJS, Svelte, Astro, Qwik, HTMX or jQuery)
- You kept it clean and simple. We don't want to work with Ivory Tower architects who just love to show-off their big brains 😊
- The working app is hosted online using any of the following free hosting services:
    - Microsoft Azure
    - Amazon AWS
    - Vercel
    - Netlify
    - Cloudflare Workers
    - Fly.io
    - Render