**Subscribe Servlet Documentation**

Requirement: In this task i need to acess that when i used to submit the details of the users it should store and retrieve the stored details of the users

* Firstly we have to write the servlet code to

**package** com.digisite.core.servlets;

**import** java.io.IOException; -Handles input/output operations

**import** java.text.SimpleDateFormat; - Formats dates.

**import** java.util.Date; -Represents a specific instant in time.

**import** java.util.HashMap; -Implements a hash table, used for mapping keys to values.

**import** java.util.Map; -Represents a mapping between keys and values.

**import** java.util.concurrent.atomic.AtomicInteger; -Provides atomic operations on integers.

**import** javax.servlet.Servlet; -Defines methods that all servlets must implement.

* **import** javax.servlet.http.HttpServletResponse; -Provides HTTP-specific functionality in sending a response.

**import** org.apache.sling.api.SlingHttpServletRequest; -Represents the request made to the servlet.

**import** org.apache.sling.api.SlingHttpServletResponse; -Represents the response sent by the servlet.

**import** org.apache.sling.api.resource.Resource; -Represents a resource within the Sling content tree.

**import** org.apache.sling.api.resource.ResourceResolver; -Resolves resources within the Sling content tree.

**import** org.apache.sling.api.servlets.SlingAllMethodsServlet; -Base class for all servlets supporting multiple HTTP methods.

**import** org.osgi.service.component.annotations.Component; -Indicates that a class is an OSGi component.

@Component(service = Servlet.**class**, immediate = **true**,

property = {

"sling.servlet.paths=/bin/subscribe1" ---**Immediate Activation:** The servlet is activated immediately upon service registration. We have to register the servlet in system console

}

)

**public** **class** SubscribeServlet **extends** SlingAllMethodsServlet {

**private** **static** **final** **long** ***serialVersionUID*** = 1L;

**private** **static** **final** AtomicInteger ***counter*** = **new** AtomicInteger(0);

**public** **void** doGet(SlingHttpServletRequest req, SlingHttpServletResponse resp) **throws** IOException {

* **SlingHttpServletRequest req**: Represents the request made to the servlet.
* **SlingHttpServletResponse resp**: Represents the response sent by the servlet.

**try** {

String firstname = req.getParameter("First Name");

String lastname = req.getParameter("Last Name");

String email = req.getParameter("Email Id");

These lines populate the map with key-value pairs. Each key corresponds to a field name ("First Name", "Last Name", "Email Id"), and each value is the corresponding user input retrieved from the request parameters.

**ResourceResolver resolver = req.getResourceResolver();**

This line obtains a **ResourceResolver** object named **resolver** from the **SlingHttpServletRequest** object **req**. A **ResourceResolver** is used to resolve resources within the AEM repository.

Map<String, Object> map = **new** HashMap<>();

This line creates a new HashMap named **map**. A **HashMap** is a collection that stores key-value pairs, in this case, to store the user data extracted from the request parameters.

map.put("First Name", firstname);

map.put("Last Name", lastname);

map.put("Email Id", email);

These lines populate the **map** with key-value pairs. Each key corresponds to a field name ("First Name", "Last Name", "Email Id"), and each value is the corresponding user input retrieved from the request parameters.

// Generate a unique identifier for the node name

String identifier = String.*valueOf*(***counter***.getAndIncrement());

This line generates a unique identifier for the node name. It retrieves the current value of the **counter**, increments it atomically, and converts it to a string.

// Generate a unique timestamp for the node name

String timeStamp = **new** SimpleDateFormat("yyyy-MM-dd").format(**new** Date());

This line generates a timestamp in the format "yyyy-MM-dd" representing the current date.

// Construct the node name with the timestamp and identifier

String nodeName = timeStamp + "\_" + identifier;

This line constructs a unique node name by combining the timestamp and the identifier, separated by an underscore.

// Construct the node path

String nodePath = "/content/digisite/us/en/subscribe/jcr:content/" + nodeName;

This line constructs the full node path where the user data will be stored in the AEM repository.

// Create the new node

Resource parentResource = resolver.getResource("/content/digisite/us/en/subscribe/jcr:content");

This line retrieves the parent resource in the AEM repository where the new node will be created.

resolver.create(parentResource, nodeName, map);

This line creates a new node in the AEM repository with the specified node name (**nodeName**) under the parent resource, and sets the properties of the node using the key-value pairs in the **map**.

resp.getWriter().write("Thank you for submitting");

This line writes a response message "Thank you for submitting" to the **SlingHttpServletResponse** object **resp**, indicating successful submission of the user data.

resolver.commit();

This line commits the changes made to the AEM repository using the **ResourceResolver**. It ensures that the changes are persisted and visible to other parts of the system.

} **catch** (IOException e) {

resp.sendError(HttpServletResponse.***SC\_INTERNAL\_SERVER\_ERROR***, "Error processing the request");

}

}

}

* We have to give the htl code for the above servlet code

<div class="cmp-subscribe">

<form method="get" action="/bin/subscribe">

<div class="input group">

<label for ="first-name">First name:</label>

<input id="first-name" type="text" name="First Name"/>

</div>

<div class="input group">

<label for ="last-name">Last name:</label>

<input id="last-name" type="text" name="Last Name"/>

</div>

<div class="input group">

<label for ="email-id">Email id:</label>

<input id="email-id" type="email" name="Email Id"/>

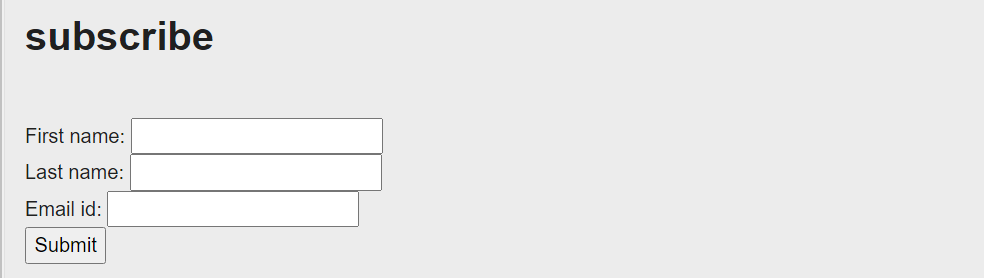
</div>

<button type="submit">Submit</button>

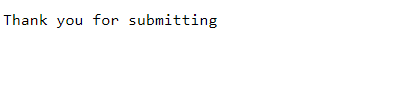
</form>

<div>

* After adding the htl code to the html then the output looks like



* Submitting the details we have to go to view as published page
* In that page we have give the user details and click on the submit button it will submitted
* And after submitting it will gives this message



* The submitted Details will store under this path ‘/content/digisite/us/en/subscribe’

