

# Customising Redmine and Superset to Allow Automatic Login with Gmail through Keycloak

## **Task Description:**

To integrate Redmine and Apache Superset with Keycloak for easy Gmail login, follow these steps:

1. Start by setting up Keycloak. Create a realm and set up Google as an identity provider using OAuth credentials from the Google API Console.
2. Next, create separate clients within Keycloak for Redmine and Superset, making sure to specify the right redirect URLs.
3. Then, install the necessary OAuth plugin in Redmine, set it up with Keycloak client details, and update Redmine's configuration file.
4. For Superset, install the required Python packages and update the `superset_config.py` file to include OIDC configuration for Keycloak.
5. Finally, restart both Redmine and Superset servers to apply the changes.

To verify the integration, access Redmine and Superset using Google credentials, and check that user information is correctly populated.

## **Table of content**

<b>Scenario: Ensure the system is up to date</b>	<b>2</b>
<b>Scenario: Install Podman</b>	<b>3</b>
<b>Scenario: Install Vim</b>	<b>4</b>
<b>Scenario: Create necessary directories and files for Redmine setup</b>	<b>5</b>
<b>Scenario: Define the Dockerfile for setting up Superset with additional packages</b>	<b>6</b>
<b>Scenario: Create and run the <code>redmine.sh</code> script to set up containers using Podman</b>	<b>8</b>
<b>Scenario: Set up the Redmine environment and install the OAuth plugin</b>	<b>9</b>
<b>Scenario: Access the Redmine container to install the OAuth plugin</b>	<b>10</b>
<b>Scenario: Access Redmine web interface and configure OAuth plugin</b>	<b>12</b>
<b>Scenario: Access Keycloak web interface</b>	<b>15</b>
<b>Scenario: Access Keycloak and create a realm</b>	<b>16</b>
<b>Scenario: Create a client in Keycloak</b>	<b>17</b>
<b>Scenario: Configure the client and obtain the client secret</b>	<b>18</b>
<b>Scenario: Configure Identity Provider in Keycloak</b>	<b>19</b>
<b>Scenario: Access Google API Dashboard and create OAuth credentials</b>	<b>20</b>

<b>Scenario: Configure Identity Provider in Keycloak with Google OAuth</b>	<b>22</b>
<b>Scenario: Configure Redmine to use Keycloak OAuth plugin</b>	<b>23</b>
<b>Scenario: Configure Redmine authentication settings</b>	<b>24</b>
<b>Scenario: Create a user in Redmine</b>	<b>25</b>
<b>Scenario: Authenticate with Keycloak and Google OAuth</b>	<b>26</b>
<b>Scenario: Build and run the Superset Docker image</b>	<b>28</b>
<b>Scenario: Create Superset admin user and initialise the database</b>	<b>30</b>
<b>Scenario: Update Superset configuration</b>	<b>32</b>
<b>Scenario: Connect Superset to PostgreSQL</b>	<b>34</b>
<b>Scenario: Configure Superset for Keycloak SSO</b>	<b>34</b>
<b>Scenario: Login to Superset via Keycloak</b>	<b>36</b>
<b>Scenario: Test SSO Integration with Redmine and Superset</b>	<b>38</b>

## Test cases

Task: 1

<b>Scenario: Ensure the system is up to date</b>	
<b>Given:</b> User has a terminal open.	<b>Output:</b> <pre>manish@manish: ~\$ sudo apt update [sudo] password for manish: Get: http://security.ubuntu.com/ubuntu jammy-security InRelease [129 kB] Hit:2 http://in.archive.ubuntu.com/ubuntu jammy InRelease Get:3 http://in.archive.ubuntu.com/ubuntu jammy-updates InRelease [128 kB] Hit:4 http://in.archive.ubuntu.com/ubuntu jammy-backports InRelease Fetched 257 kB in 2s (111 kB/s) Reading package lists... Done Building dependency tree... Done Reading state information... Done All packages are up to date. manish@manish: ~\$ sudo apt upgrade Reading package lists... Done Building dependency tree... Done Reading state information... Done Calculating upgrade... Done The following packages were automatically installed and are no longer required:   libwpe-1.0-1 libwpebackend-fdo-1.0-1 Use 'sudo apt autoremove' to remove them. 0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded. manish@manish: ~\$</pre>
<b>When:</b> User run "sudo apt update"	
<b>And:</b> User run "sudo apt upgrade"	
<b>Then:</b> User system packages updated to the latest versions	
<b>Result:</b> Passed	



## Task: 2

Scenario: Install Podman	
<b>Given:</b> User system packages are up to date	<b>OUTPUT:</b> <pre>root@nanih-pc: ~ \$ sudo systemctl status podman ● podman.service - Podman API Service    Loaded: loaded (/lib/systemd/system/podman.service; enabled; vendor preset: enabled)      Active: inactive (dead) since Thu 2024-06-28 10:13:35 IST; 25min ago        Docs: man:podman-system-service(1)     Process: 658 ExecStart=/usr/bin/podman SLOGGING system service (code=exited, status=0/SUCCESS)       Main PID: 658 (code=exited, status=0/SUCCESS)         CPU: 10ms  Jun 28 10:13:29 nanih systemd[1]: Starting Podman API Service.. Jun 28 10:13:29 nanih systemd[1]: Started Podman API Service.</pre>
<b>When:</b> User run "sudo apt install podman"	<pre>Jun 28 10:13:30 nanih podman[658]: time="2024-06-28T10:13:30+05:30" level=info msg="/usr/bin/podman filtering at log level info" Jun 28 10:13:30 nanih podman[658]: time="2024-06-28T10:13:30+05:30" level=info msg="[graphdriver] using prior storage driver: overlay" Jun 28 10:13:30 nanih podman[658]: time="2024-06-28T10:13:30+05:30" level=info msg="Found CNI network podman (type=bridge) at /etc/cni/net.d/87-podman.yaml" Jun 28 10:13:30 nanih podman[658]: time="2024-06-28T10:13:30+05:30" level=info msg="Setting parallel job count to 13" Jun 28 10:13:30 nanih podman[658]: time="2024-06-28T10:13:30+05:30" level=info msg="using systemd socket activation to determine API endpoint" Jun 28 10:13:30 nanih podman[658]: time="2024-06-28T10:13:30+05:30" level=info msg="using API endpoint: \"" Jun 28 10:13:30 nanih podman[658]: time="2024-06-28T10:13:30+05:30" level=info msg="API service listening on \"/run/podman/podman.sock\"" Jun 28 10:13:35 nanih systemd[1]: podman.service: Deactivated successfully. (lines 1-19/19 (END))</pre>
<b>Then:</b> Podman installed on the system	
<b>Result:</b> Passed	

### Task: 3

#### Task: 4

<b>Scenario: Create necessary directories and files for Redmine setup</b>	
<b>Given:</b> User Unix-like operating system environment	<b>Output:</b> <code>exit manish@manish:~/redmine\$ ls Dockerfile redmine.sh superset manish@manish:~/redmine\$</code>
<b>When:</b> User run the following commands:  mkdir redmine touch redmine.sh vim redmine.sh cat redmine.sh touch Dockerfile	
<b>Then:</b> User following conditions met:  A directory named redmine is created. An empty file named redmine.sh is created. The file redmine.sh can be edited using vim. The contents of redmine.sh can be displayed using cat. An empty file named Dockerfile is created.	
<b>Result: Passed</b>	

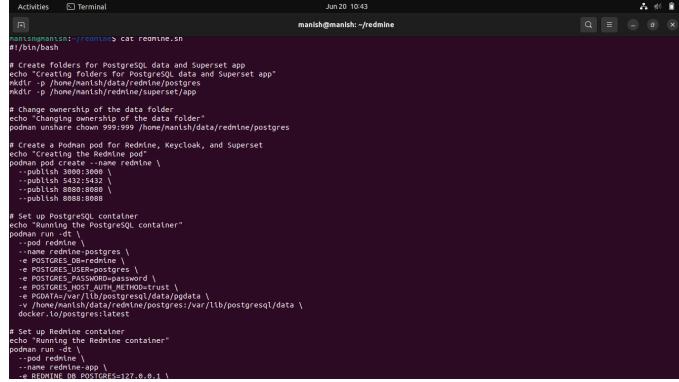
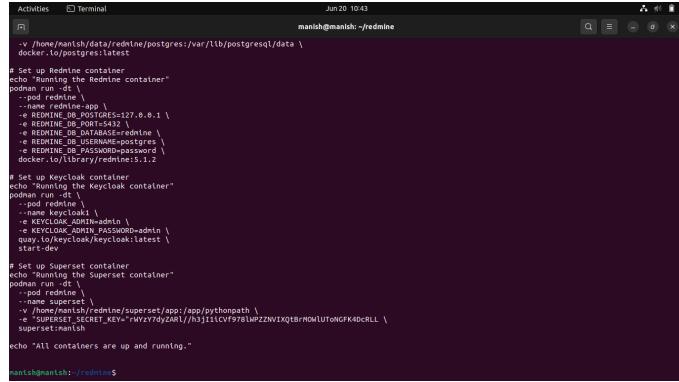
## Task: 5

<b>Scenario: Define the Dockerfile for setting up Superset with additional packages</b>	
<b>Given:</b> User empty <code>Dockerfile</code>	<b>Output:</b> <pre>manish@manish:~/redmine\$ cat Dockerfile # syntax=docker/dockerfile:1  FROM docker.io/apache/superset  # Switching to root to install the required packages USER root  # if you prefer Postgres, you may want to use `psycopg2-binary` instead RUN pip install psycopg2-binary RUN pip install flask-oidc==1.3.0 RUN pip install flask openid RUN pip install python-keycloak RUN pip install itsdangerous==2.0.1  # Switching back to using the `superset` user USER superset</pre>
<b>When:</b> User add the following content to the Dockerfile with ss in the output box.	
<b>Then:</b> User Dockerfile up a Superset environment with the required packages installed.	
<b>Result:</b> Passed	

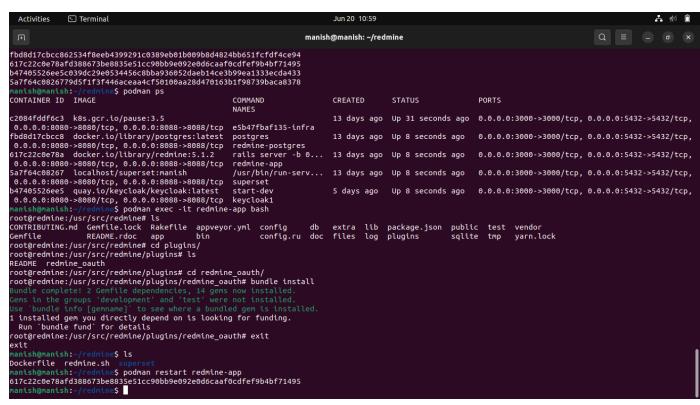
## Task: 6

<b>Scenario: Create and run the redmine.sh script to set up containers using Podman</b>	
<p><b>Given:</b> User properly configured redmine.sh script with the following content in the output box.</p> <pre>echo "All containers are up and running."</pre>	<p><b>Output:</b></p> <pre>[root@ip-172-31-10-1 ~]# sh redmine.sh Creating Folders for PostgreSQL data and Superset app Changing ownership of the data folder Creating the Redmine pod Error: Error adding pod to state: name "redmine" is in use: pod already exists Running the PostgreSQL container Error: error creating container storage: the container name "redmine-postgres" is already in use by "fbd8d17cc8d02534f8eeb4399291c8309eb01b009b8d4824b651fcfd4ce94". You have to remove that container to be able to reuse that name.: that name is already in use Running the Redmine container Error: error creating container storage: the container name "redmine-app" is already in use by "617c722c0e70af388673be8835e51cc90bb9e092e0dcaaf0cdfe95abf71495". You have to remove that container to be able to reuse that name.: that name is already in use Running the Keycloak container Error: error creating container storage: the container name "keycloak" is already in use by "b47405526e5c839dc29e0534450c8bba936052daeb14ce3b99ea1333edcd431". You have to remove that container to be able to reuse that name.: that name is already in use Running the Superset container Error: error creating container storage: the container name "superset" is already in use by "5a7f64cd026779df1f3f446aceaa4cf5010aa2d470163b1f98739baca8378". You have to remove that container to be able to reuse that name.: that name is already in use All containers are up and running. [root@ip-172-31-10-1 ~]#</pre>
<p><b>When:</b> User execute <code>./redmine.sh</code></p>	
<p><b>Then:</b> User following conditions met:</p> <ul style="list-style-type: none"> <li>Directories for PostgreSQL data and Superset app are created.</li> <li>Ownership of the PostgreSQL data directory is changed.</li> <li>A Podman pod named <b>redmine</b> is created with the specified ports published.</li> <li>Containers for PostgreSQL, Redmine, Keycloak, and Superset are created and running within the <b>redmine</b> pod.</li> <li>A message indicating "All containers are up and running" is displayed.</li> </ul>	<pre>[root@ip-172-31-10-1 ~]# podman ps CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS c2084ffdfcc3 kbs.grc.l0/pause:1.5 0.0.0.0:8080-&gt;8080/tcp, 0.0.0.0:8088-&gt;8088/tcp es4afffbaf135-infra 13 days ago Up 31 seconds ago 0.0.0.0:1942-&gt;5432/tcp, 0.0.0.0:8080-&gt;8080/tcp, 0.0.0.0:8088-&gt;8088/tcp 617c722c0e70af388673be8835e51cc90bb9e092e0dcaaf0cdfe95abf71495-docker.io/library/redmine:5.1.2 rails server -b 0... 13 days ago Up 8 seconds ago 0.0.0.0:3000-&gt;3000/tcp, 0.0.0.0:5432-&gt;5432/tcp, 0.0.0.0:8080-&gt;8080/tcp, 0.0.0.0:8088-&gt;8088/tcp 5a7f64cd026779df1f3f446aceaa4cf5010aa2d470163b1f98739baca8378-localhost/superset:main:sh /usr/bin/run-serv... 13 days ago Up 8 seconds ago 0.0.0.0:3000-&gt;3000/tcp, 0.0.0.0:5432-&gt;5432/tcp, 0.0.0.0:8080-&gt;8080/tcp, 0.0.0.0:8088-&gt;8088/tcp superset 5a7f64cd026779df1f3f446aceaa4cf5010aa2d470163b1f98739baca8378-keycloak:1.0 keycloak 5 days ago Up 8 seconds ago 0.0.0.0:3000-&gt;3000/tcp, 0.0.0.0:5432-&gt;5432/tcp, 0.0.0.0:8080-&gt;8080/tcp, 0.0.0.0:8088-&gt;8088/tcp [root@ip-172-31-10-1 ~]#</pre>
<p><b>Result: Passed</b></p>	

## Task: 7

Scenario: Set up the Redmine environment and install the OAuth plugin	
<b>Given:</b> User Unix-like operating system with Podman installed	<b>Output:</b>
<b>And:</b> User <code>redmine.sh</code> script that sets up necessary directories and containers	
<b>And:</b> User Redmine container connected to a PostgreSQL database	
<b>When:</b> User execute the command <code>sh redmine.sh</code>	
<b>Then:</b> User the script should:	
<ul style="list-style-type: none"> <li>Create directories for PostgreSQL data and the Superset app</li> <li>Change ownership of the PostgreSQL data directory</li> <li>Create a Podman pod named <code>redmine</code> with specified ports published</li> <li>Run containers for PostgreSQL, Redmine, Keycloak, and Superset within the <code>redmine</code> pod</li> <li>Display a message indicating "All containers are up and running"</li> </ul>	
<b>Result: Passed</b>	

**Task: 8**

<b>Scenario: Access the Redmine container to install the OAuth plugin</b>	
<b>Given:</b> User redmine.sh script has been executed and all containers are running	<b>Output:</b> <pre>root@manish:~/redmine\$ podman exec -it redmine-app bash root@redmine:/usr/src/redmine# ls CONTRIBUTING.md Gemfile.lock Rakefile appveyor.yml config db extra lib package.json public test vendor Gemfile README.rdoc app bin config.ru doc files log plugins sqlite tmp yarn.lock root@redmine:/usr/src/redmine# cd plugins/ root@redmine:/usr/src/redmine/plugins# ls README redmine_oauth root@redmine:/usr/src/redmine/plugins# cd redmine_oauth/ root@redmine:/usr/src/redmine/plugins/redmine_oauth# bundle install bundle completed: 2 Gemfile dependencies, 14 gems now installed. Gems in the groups 'development' and 'test' were not installed. Use <code>bundle info [gemname]</code> to see where a bundled gem is installed. 1 installed gem you directly depend on is looking for funding. Run <code>bundle fund</code> for details root@redmine:/usr/src/redmine/plugins/redmine_oauth#</pre>
<b>When:</b> User run the command <code>podman exec -it redmine-app bash</code>	
<b>Then:</b> User gain interactive shell access to the Redmine container	
<b>When:</b> User navigate to the plugins directory with <code>cd plugins/</code>	
<b>And:</b> User list the contents of the directory with <code>ls</code>	
<b>Then:</b> User directory empty or show the current plugins	
<b>When:</b> User clone the OAuth plugin with git clone <a href="https://github.com/kontron/redmine_oauth.git">https://github.com/kontron/redmine_oauth.git</a>	
<b>And:</b> User list the contents of the directory with <code>ls</code>	

**Then:** User redmine\_oauth directory present

**When:** User navigate into the redmine\_oauth directory with cd redmine\_oauth/

**And:** User install the bundle with bundle install

**Then:** User necessary dependencies for the redmine\_oauth plugin installed

**When:** User navigates back to the plugins directory with cd ..

**And:** User install the bundle in the plugins directory with bundle install

**Then:** User necessary dependencies for all plugins installed

**When:** User restart the Redmine container with podman restart redmine-app

```
Activities Terminal Jun 20 10:59
manish@manish:~/redmine
$ podman ps
CONTAINER ID IMAGE COMMAND NAMES STATUS PORTS
c200ff4df0fc k8s.gcr.io/pause:3.1 0.0.0.0:8088->8088/tcp e8a7bfaf135-lnfra 13 days ago Up 31 seconds ago 0.0.0.0:3000->3000/tcp, 0.0.0.0:5432->5432/tcp, 0.0.0.0:8088->8088/tcp, 0.0.0.0:8088->8088/tcp, 0.0.0.0:8088->8088/tcp, 0.0.0.0:8088->8088/tcp
f0dd17bc8c docker.io/library/postgres:latest postgres 13 days ago Up 8 seconds ago 0.0.0.0:3000->3000/tcp, 0.0.0.0:5432->5432/tcp, 0.0.0.0:5432->5432/tcp, 0.0.0.0:8088->8088/tcp, 0.0.0.0:8088->8088/tcp, 0.0.0.0:8088->8088/tcp
617c22cbe78a docker.io/library/redmine:5.1.2 rails server -b 0... 13 days ago Up 8 seconds ago 0.0.0.0:3000->3000/tcp, 0.0.0.0:5432->5432/tcp, 0.0.0.0:8088->8088/tcp, 0.0.0.0:8088->8088/tcp, 0.0.0.0:8088->8088/tcp, 0.0.0.0:8088->8088/tcp
373e2319a1f1 docker.io/library/keycloak:15.1.2 keycloak run-serv... 13 days ago Up 8 seconds ago 0.0.0.0:3000->3000/tcp, 0.0.0.0:5432->5432/tcp, 0.0.0.0:8088->8088/tcp, 0.0.0.0:8088->8088/tcp, 0.0.0.0:8088->8088/tcp, 0.0.0.0:8088->8088/tcp
047405320eef quay.io/keycloak/keycloak:latest superset 5 days ago Up 8 seconds ago 0.0.0.0:3000->3000/tcp, 0.0.0.0:5432->5432/tcp, 0.0.0.0:8088->8088/tcp, 0.0.0.0:8088->8088/tcp, 0.0.0.0:8088->8088/tcp, 0.0.0.0:8088->8088/tcp
manish@manish:~$ podman exec -it redmine-app bash
root@redmine:/# cd /src/redmine/plugins
root@redmine:/src/redmine/plugins# ls
README redmine_oauth
root@redmine:/src/redmine/plugins# cd redmine_oauth/
root@redmine:/src/redmine/plugins/redmine_oauth# bundle install
bundle completed 2 Gemfile dependencies, 14 gems now installed.
Gemfile.lock is updated.
To see which gems are bundled, run `bundle lock` or
use `bundle info [gemname]` to see where a bundled gem is installed.
I installed gem you directly depend on to looking for funding.
Run `bundle config --local source` to change this.
root@redmine:/src/redmine/plugins/redmine_oauth# exit
exit
root@redmine:~# ls
dockerfile redmine.sh superset
root@redmine:~# podman restart redmine-app
617c22cbe78afcd388673be883165cc9b8b092zeddcaaf0cdfe9bf71495
manish@manish:~/redmine$
```

**Then:** User Redmine application restart with the newly installed **redmine\_oauth** plugin

**Result: Passed**

### Task: 9

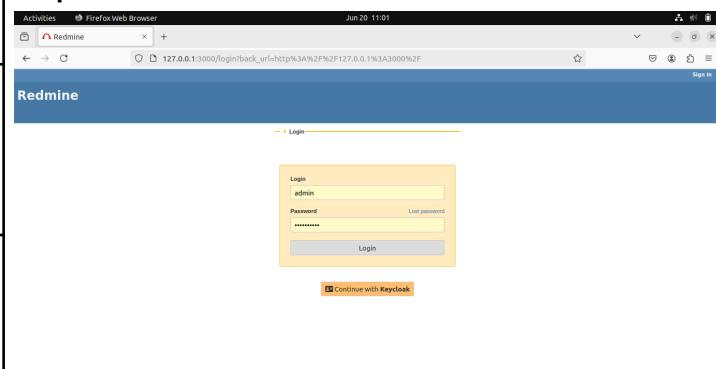
#### Scenario: Access Redmine web interface and configure OAuth plugin

**Given:** User Redmine and Keycloak containers are running

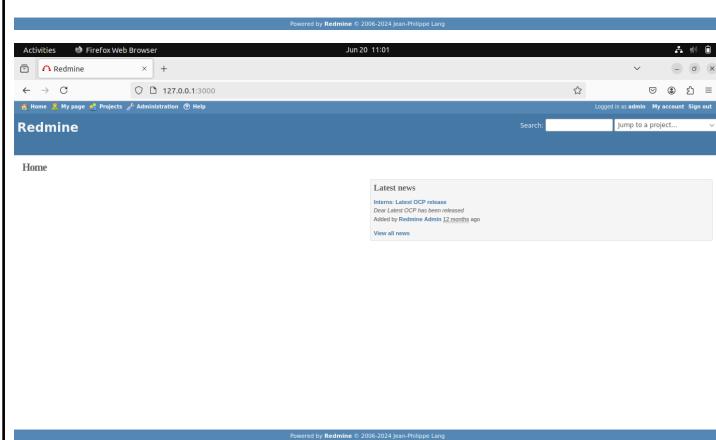
**When:** User open a web browser and navigate to **http://127.0.0.1:3000**

**Then:** User Redmine login page displayed

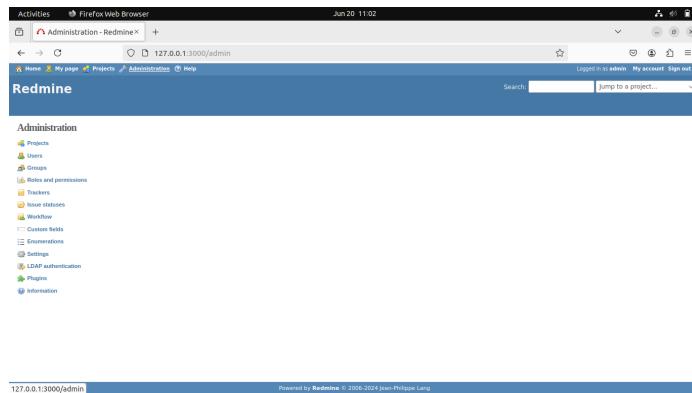
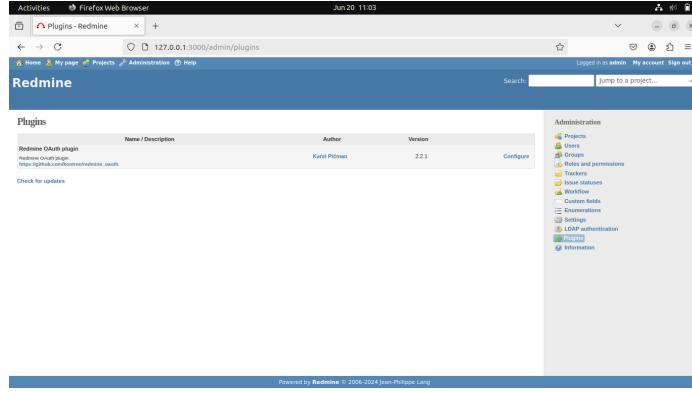
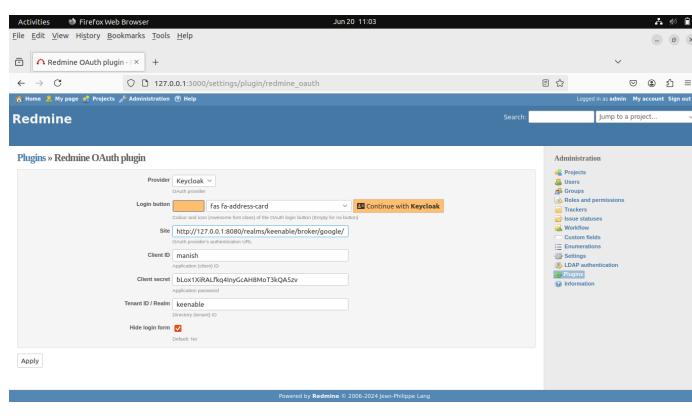
#### Output:

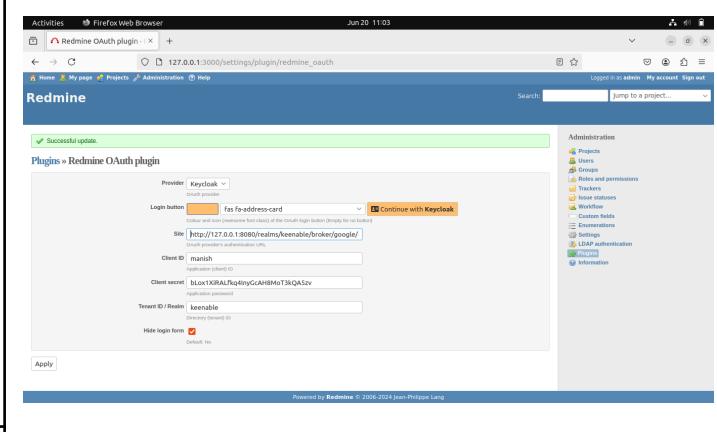
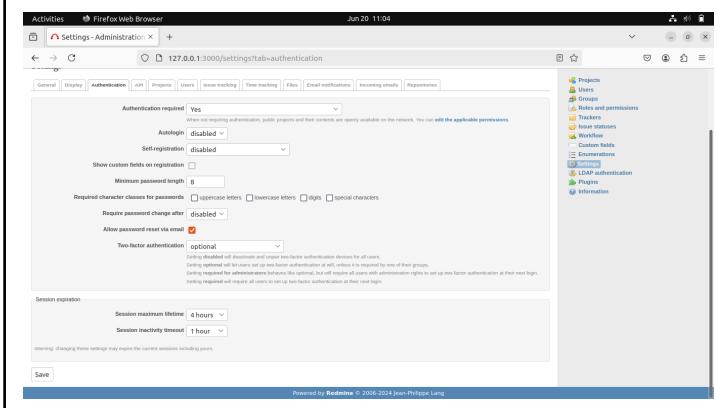
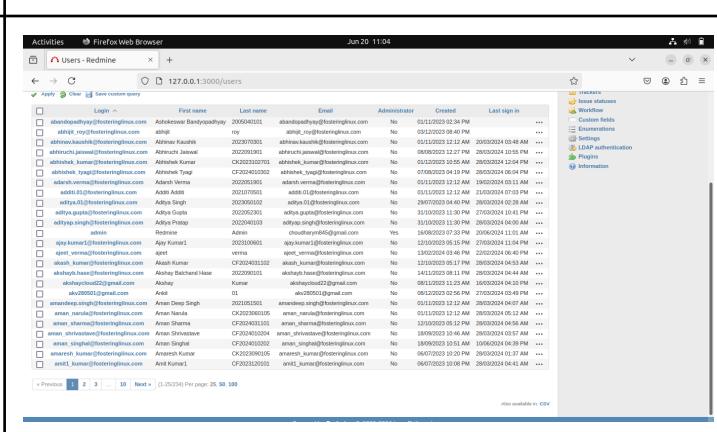


The screenshot shows a Firefox browser window with the Redmine login page. The URL is 127.0.0.1:3000/login/back\_url=http%3A%2F%2F127.0.0.1%3A3000%2F. The login form has 'admin' in the username field and a masked password. Below the form is a 'Login' button and a 'Continue with Keycloak' button, which is highlighted with a yellow box.

The screenshot shows the Redmine home page after logging in as 'admin'. The top navigation bar shows 'Logged in as admin'. The main content area displays 'Latest news' with a single entry: 'Interne: Latest OCP release' by 'Dove Latte OCP' from 'Admin' 12 months ago. There is also a 'View all news' link.

<p><b>When:</b> User login to Redmine with the username "admin" and password "admin"</p>	
<p><b>Then:</b> User gain access to the Redmine dashboard</p>	
<p><b>When:</b> User click the "Administration" option</p>	
<p><b>Then:</b> User administration page should be displayed</p>	

<p><b>When:</b> User click the "Plugins" option</p>	
<p><b>Then:</b> User list of installed plugins displayed</p>	
<p><b>When:</b> User click the "Configure" option for the OAuth plugin</p>	
<p><b>Then:</b> User plugin configuration page displayed</p>	
<p><b>When:</b> User click the "Provider" dropdown and select the "Keycloak" option</p>	

**Then:** User Keycloak option selected as the OAuth provider

**Result: Passed**

## Task: 10

### Scenario: Access Keycloak web interface

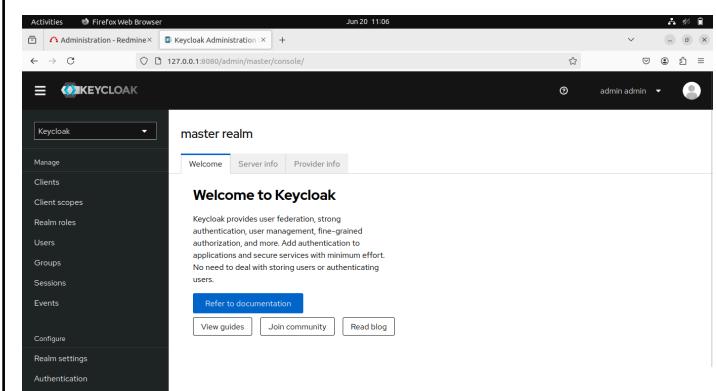
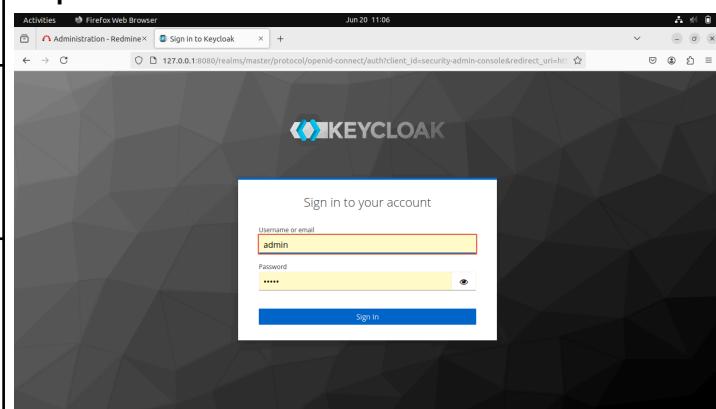
**Given:** User Keycloak container is running

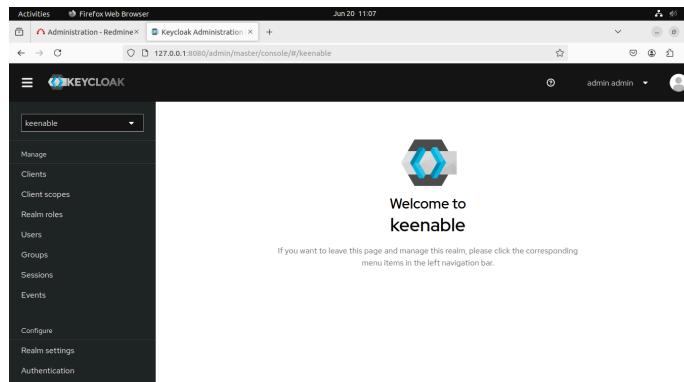
**When:** User open a web browser and navigate to <http://127.0.0.1:8080>

**Then:** User Keycloak login page displayed

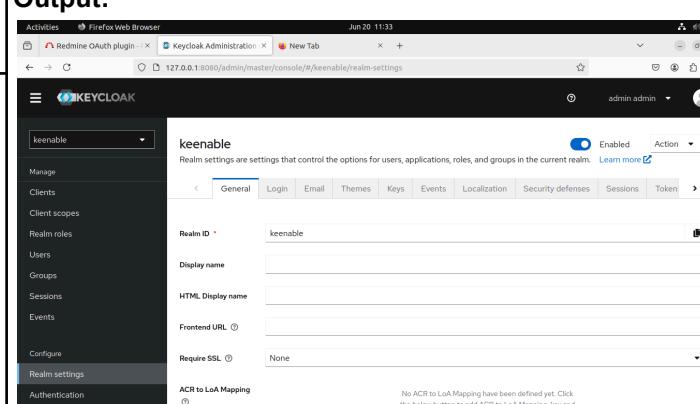
**When:** User log in to Keycloak with the username "admin" and password "admin"

#### Output:



<p><b>Then:</b> User gain access to the Keycloak admin console</p>	 <p>The screenshot shows the Keycloak Admin Console interface. The left sidebar has a dark theme with white text and icons. It lists several menu items: Manage, Clients, Client scopes, Realm roles, Users, Groups, Sessions, Events, Configure, Realm settings (which is currently selected), Authentication, and Identity providers. The main content area has a light background. At the top right, there is a logo for 'KEYCLOAK' and the text 'Welcome to keenable'. Below that, a message says, 'If you want to leave this page and manage this realm, please click the corresponding menu items in the left navigation bar.' The URL in the browser's address bar is '127.0.0.1:8080/admin/master/console/#/keenable'.</p>
<b>Result: Passed</b>	

## Task: 11

<b>Scenario: Access Keycloak and create a realm</b>	
<p><b>Given:</b> User logged in to the Keycloak admin console</p>	<p><b>Output:</b></p>  <p>The screenshot shows the Keycloak Admin Console interface. The left sidebar has a dark theme with white text and icons. It lists several menu items: Manage, Clients, Client scopes, Realm roles, Users, Groups, Sessions, Events, Configure, Realm settings (which is currently selected), Authentication, and Identity providers. The main content area has a light background. The 'General' tab is selected in the top navigation bar. The 'Realm ID' field is set to 'keenable'. There are other fields for 'Display name', 'HTML Display name', 'Frontend URL', 'Require SSL', and 'ACR to LoA Mapping'. A note at the bottom says, 'No ACR to LoA Mapping have been defined yet. Click the below button to add ACR to LoA Mapping key and value are required for a key pair.' The URL in the browser's address bar is '127.0.0.1:8080/admin/master/console/#/keenable/realm-settings'.</p>
<p><b>When:</b> User navigate to the "Realms" section</p>	
<p><b>And:</b> User create a new realm with the necessary details</p>	

**Then:** User new realm created and listed in the realms section

**Result: Passed**

## Task: 12

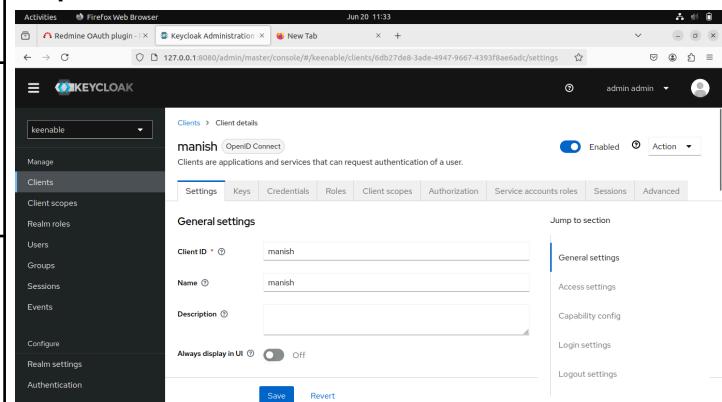
### Scenario: Create a client in Keycloak

**Given:** User have created a realm in Keycloak

**When:** User navigate to the "Clients" section within the realm

**And:** User create a new client with the necessary details

### Output:

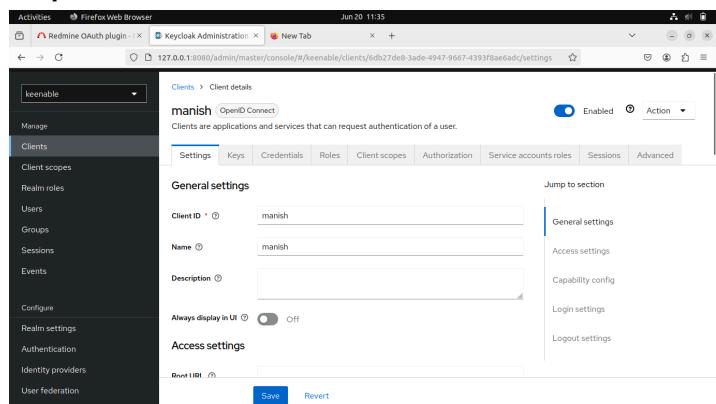
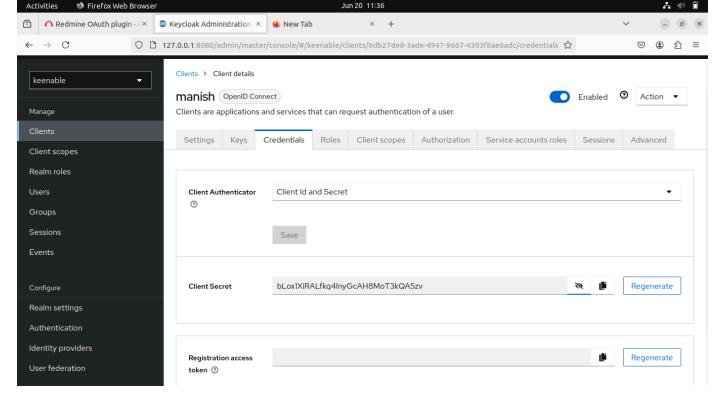


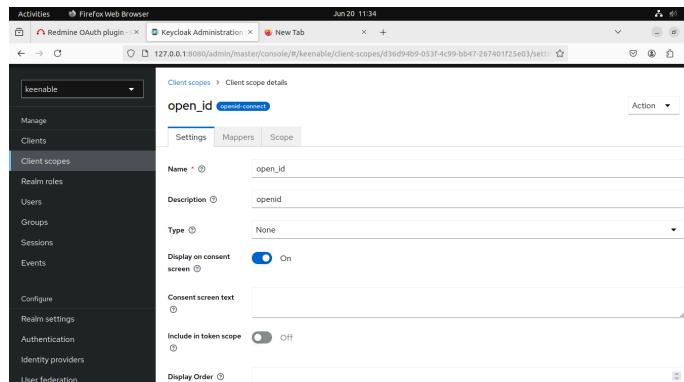
The screenshot shows the Keycloak Administration interface. The left sidebar is titled 'keenable' and has sections for Manage, Clients, Client scopes, Realm roles, Users, Groups, Sessions, Events, Configure, Realm settings, Authentication, and Identity providers. The 'Clients' section is currently selected. The main panel shows a list of clients with one entry: 'manish' (OpenID Connect). Below the list, there is a 'Client details' form with tabs for Settings, Keys, Credentials, Roles, Client scopes, Authorization, Service accounts roles, Sessions, and Advanced. The 'General settings' tab is active, showing fields for Client ID (manish), Name (manish), and Description. A note says 'Clients are applications and services that can request authentication of a user.' On the right side, there is a sidebar with sections for General settings, Access settings, Capability config, Login settings, and Logout settings. At the bottom of the main panel, there are 'Save' and 'Revert' buttons.

**Then:** User new client created and listed in the clients section

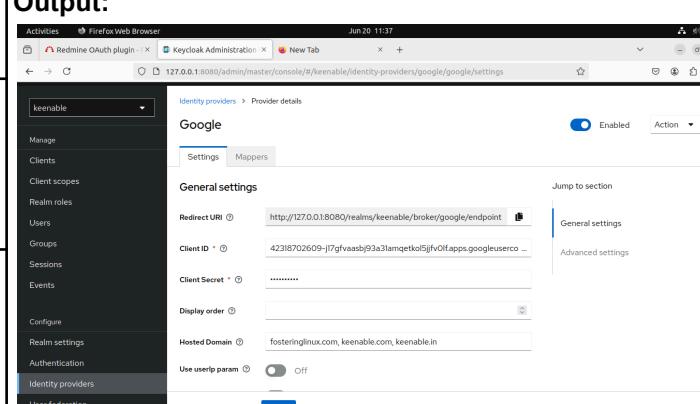
**Result: Passed**

### Task: 13

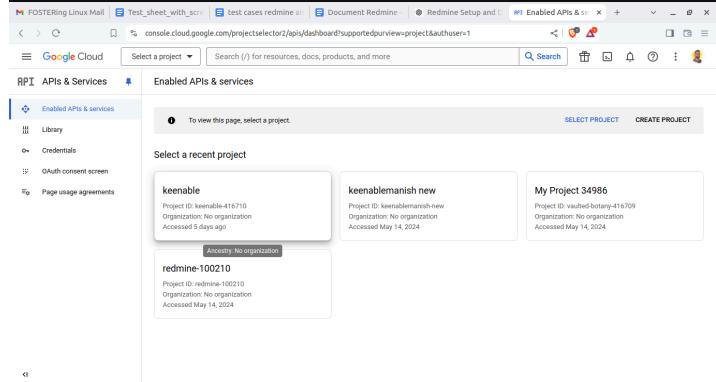
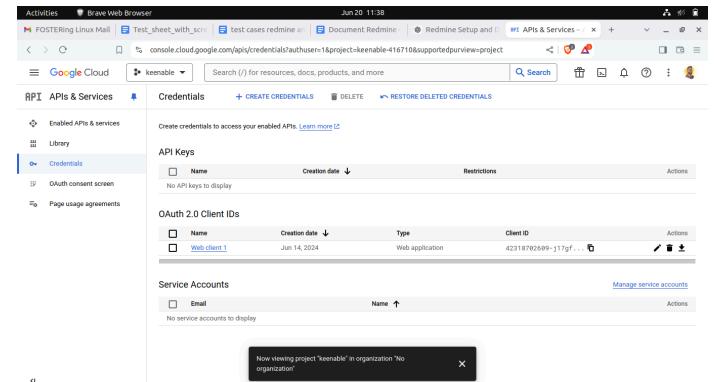
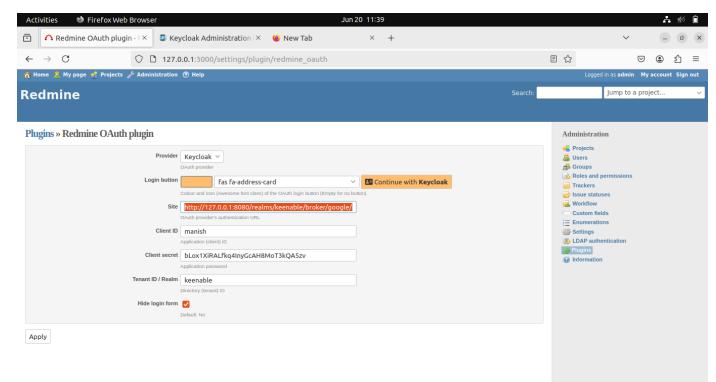
<b>Scenario: Configure the client and obtain the client secret</b>	
<b>Given:</b> User have created a client in Keycloak	<b>Output:</b> 
<b>When:</b> User click the "Next" button to proceed with the client configuration	
<b>Then:</b> User taken to the client settings page	
<b>When:</b> User navigate to the "Credentials" tab for the client	
<b>Then:</b> User see the client secret	

<p><b>And:</b> User copy the client secret</p>	 <p>The screenshot shows the 'Client scopes' section in Keycloak. A new scope named 'open_id' is being configured. The 'Settings' tab is selected, showing the following details:</p> <ul style="list-style-type: none"> <li>Name: open_id</li> <li>Description: openid</li> <li>Type: None</li> <li>Display on consent screen: On</li> <li>Consent screen text: (empty)</li> <li>Include in token scope: Off</li> <li>Display Order: (empty)</li> </ul>
<b>Result: Passed</b>	

## Task: 14

<b>Scenario: Configure Identity Provider in Keycloak</b>	
<p><b>Given:</b> User have copied the client secret</p>	<p><b>Output:</b></p>  <p>The screenshot shows the 'Identity providers' section for the 'Google' provider. The 'Settings' tab is selected, showing the following details:</p> <ul style="list-style-type: none"> <li>Enabled: On</li> <li>General settings:       <ul style="list-style-type: none"> <li>Redirect URI: http://127.0.0.1:8080/realm/keenable/broker/google/endpoint</li> <li>Client ID: 423b702609-jl7gfvaasd93a3lamgetkol5jfv0lf.apps.googleusercontent.com</li> <li>Client Secret: (redacted)</li> <li>Display order: (empty)</li> <li>Hosted Domain: fosteringlinux.com, keenable.com, keenable.in</li> <li>User login param: Off</li> </ul> </li> </ul>
<p><b>When:</b> User navigate to the "Identity Providers" section within the realm</p>	
<p><b>Then:</b> User able to configure the identity provider with the necessary details</p>	
<b>Result: Passed</b>	

## Task: 15

Scenario: Access Google API Dashboard and create OAuth credentials	
<b>Given:</b> User have access to the Google API Dashboard using my personal Gmail account	<b>Output:</b> 
<b>When:</b> User navigate to <a href="https://console.cloud.google.com/projectselector2/apis/dashboard?supportedpurview=project&amp;authuser=1">https://console.cloud.google.com/projectselector2/apis/dashboard?supportedpurview=project&amp;authuser=1</a>	
<b>Then:</b> User see a list of projects associated with my account	
<b>When:</b> User select the keenable project	
<b>Then:</b> User project dashboard should be displayed	
<b>And:</b> User select "Create credentials"	

**And:** User choose "OAuth 2.0 Client IDs"

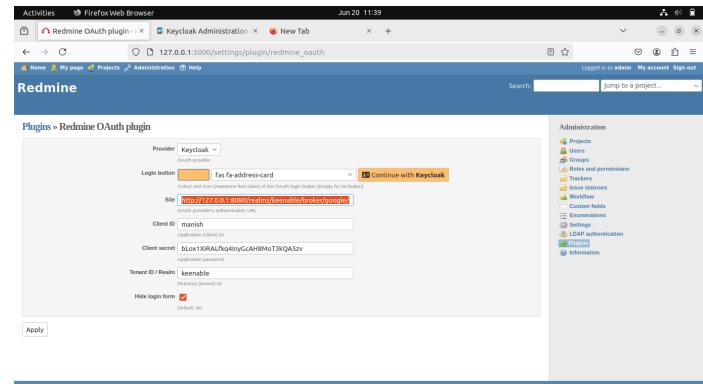
**Then:** User prompted to configure OAuth credentials

**When:** User select "Web application" as the application type

**And:** User configure the required fields with the necessary information, including the redirect URI

**http://127.0.0.1:3000/oauth2callback**

**And:** User create the credentials



**Then:** User see the client ID and client secret

**When:** User copy the client ID and client secret

**Then:** User able to paste these credentials into Keycloak

**Result: Passed**

## Task: 16

### Scenario: Configure Identity Provider in Keycloak with Google OAuth

**Given:** User have the client ID and client secret from Google

**When:** User navigate to the "Identity Providers" section in Keycloak

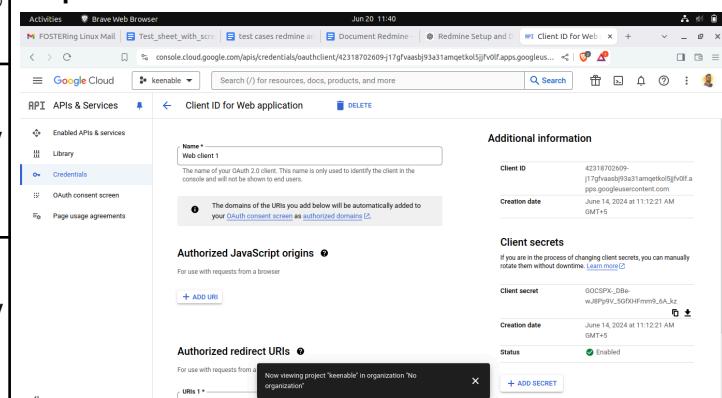
**And:** User select "Google" as the identity provider

**And:** User paste the client ID and client secret into the appropriate fields in Keycloak

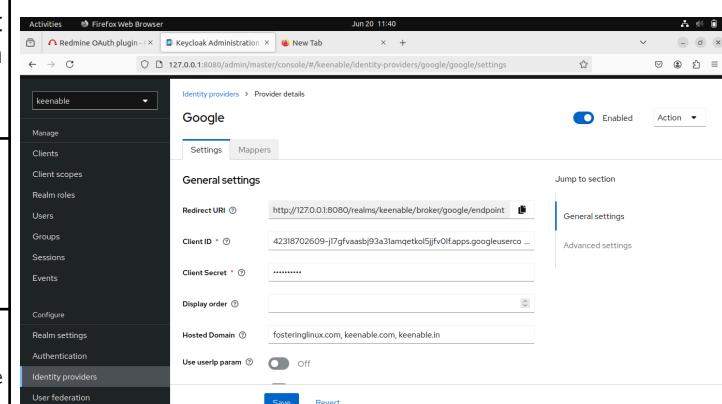
**And:** User save the configuration

**Then:** Keycloak configured to use Google OAuth for authentication

#### Output:



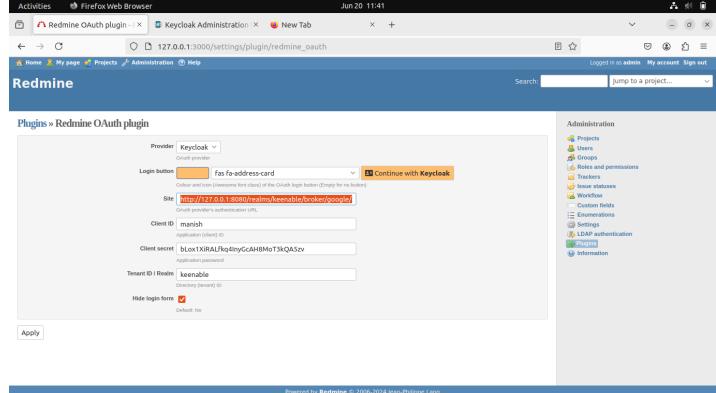
The screenshot shows the 'RPI APIs & Services' section in the Keenable web interface. A new client ID for a web application is being created. The client ID is set to 'Web-client 1'. The client secret is listed as 'GOISIPw\_DBw-vJfFpyvJ\_5GtfHfmrm9\_LA\_kz'. The creation date is 'June 14, 2024 at 11:12:21 AM' and the status is 'Enabled'.



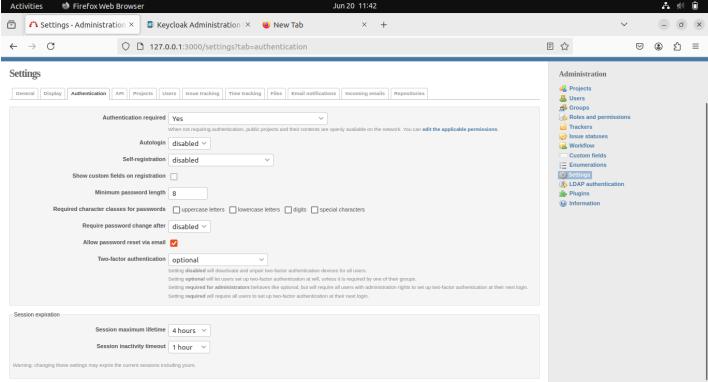
The screenshot shows the 'Identity providers' section in the Keycloak Administration interface. A new provider for 'Google' is being configured. The 'General settings' tab is selected. The Redirect URI is set to 'http://127.0.0.1:8080/realm/keenable/broker/google/endpoint'. The Client ID is '42318702609-j7gfaasbj93a3lamqekol5jfv0f.apps.googleusercontent.com'. The Client Secret is redacted. The Hosted Domain is 'fosteringlinux.com, keenable.com, keenable.in'. The 'Use userip param' option is set to 'Off'. The provider is marked as 'Enabled'.

**Result: Passed**

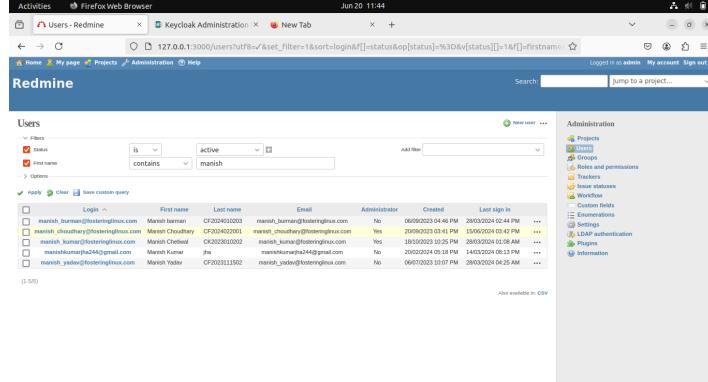
## Task: 17

<b>Scenario: Configure Redmine to use Keycloak OAuth plugin</b>	
<b>Given:</b> User logged in to Redmine as an admin	<b>Output:</b> 
<b>When:</b> User navigate to the "Administration" section	
<b>And:</b> User select the "Plugins" option	
<b>And:</b> User fill in the necessary configuration details for the Keycloak OAuth plugin	
<b>Then:</b> User plugin configured correctly	
<b>Result:</b> Passed	

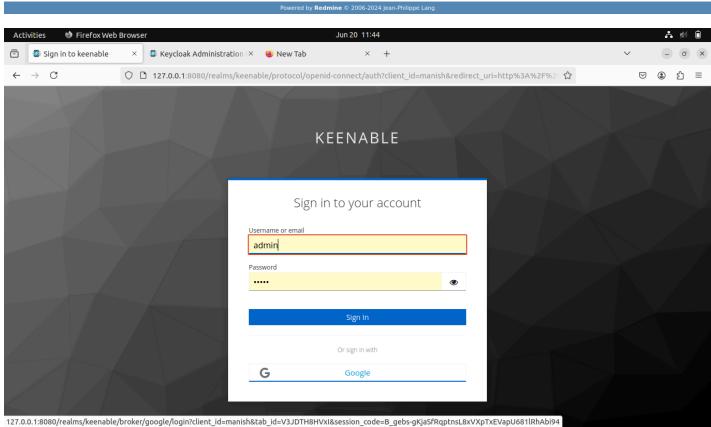
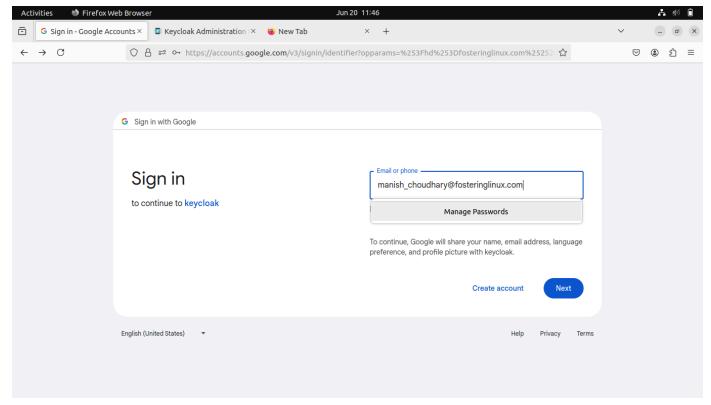
## Task: 18

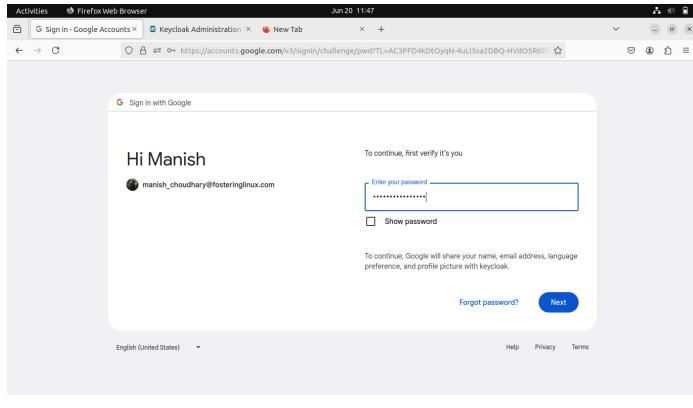
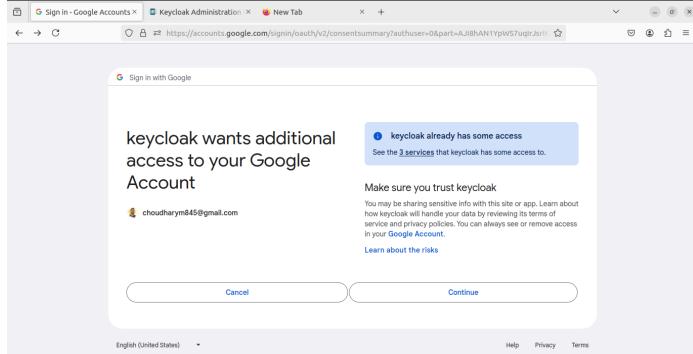
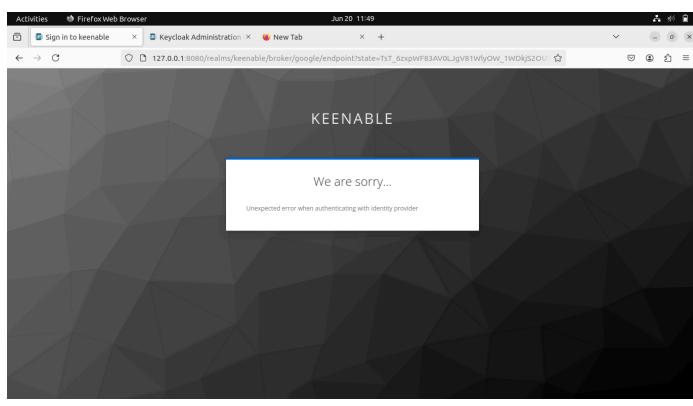
Scenario: Configure Redmine authentication settings	
<b>Given:</b> User have configured the Keycloak OAuth plugin in Redmine	<b>Output:</b> 
<b>When:</b> User navigate to the "Settings" section	
<b>And:</b> User select the "Authentication" tab	
<b>And:</b> User configure the authentication settings to use Keycloak	
<b>Then:</b> User authentication settings updated correctly	
<b>Result:</b> Passed	

### Task: 19

<b>Scenario: Create a user in Redmine</b>																																																
<b>Given:</b> User am logged in to Redmine as an admin	<b>Output:</b>  <table border="1"> <thead> <tr> <th>Login</th> <th>First name</th> <th>Last name</th> <th>Email</th> <th>Administrator</th> <th>Created</th> <th>Last sign in</th> </tr> </thead> <tbody> <tr> <td>manish_burnam@fosteringlinux.com</td> <td>Manish</td> <td>burnam</td> <td>CF202402003</td> <td>manish_burnam@fosteringlinux.com</td> <td>No</td> <td>06/05/2023 04:46 PM</td> <td>28/03/2024 02:44 PM</td> </tr> <tr> <td>manish_choudhary@fosteringlinux.com</td> <td>Marsh</td> <td>Choudhary</td> <td>CF202402003</td> <td>manish_choudhary@fosteringlinux.com</td> <td>Yes</td> <td>20/05/2023 03:41 PM</td> <td>15/06/2024 03:42 PM</td> </tr> <tr> <td>manish_kumar@fosteringlinux.com</td> <td>Marsh</td> <td>Chetan</td> <td>OK202802003</td> <td>manish_kumar@fosteringlinux.com</td> <td>Yes</td> <td>18/03/2023 10:25 PM</td> <td>26/03/2024 01:09 AM</td> </tr> <tr> <td>manish_kumar94@gmail.com</td> <td>Marsh</td> <td>Kumar</td> <td>Pa</td> <td>manish_kumar94@gmail.com</td> <td>No</td> <td>04/02/2024 08:04 PM</td> <td>14/03/2024 08:10 PM</td> </tr> <tr> <td>manish_yadav@fosteringlinux.com</td> <td>Manish</td> <td>Yadav</td> <td>CP2023111602</td> <td>manish_yadav@fosteringlinux.com</td> <td>No</td> <td>06/05/2023 03:07 PM</td> <td>30/03/2024 14:21 AM</td> </tr> </tbody> </table>	Login	First name	Last name	Email	Administrator	Created	Last sign in	manish_burnam@fosteringlinux.com	Manish	burnam	CF202402003	manish_burnam@fosteringlinux.com	No	06/05/2023 04:46 PM	28/03/2024 02:44 PM	manish_choudhary@fosteringlinux.com	Marsh	Choudhary	CF202402003	manish_choudhary@fosteringlinux.com	Yes	20/05/2023 03:41 PM	15/06/2024 03:42 PM	manish_kumar@fosteringlinux.com	Marsh	Chetan	OK202802003	manish_kumar@fosteringlinux.com	Yes	18/03/2023 10:25 PM	26/03/2024 01:09 AM	manish_kumar94@gmail.com	Marsh	Kumar	Pa	manish_kumar94@gmail.com	No	04/02/2024 08:04 PM	14/03/2024 08:10 PM	manish_yadav@fosteringlinux.com	Manish	Yadav	CP2023111602	manish_yadav@fosteringlinux.com	No	06/05/2023 03:07 PM	30/03/2024 14:21 AM
Login	First name	Last name	Email	Administrator	Created	Last sign in																																										
manish_burnam@fosteringlinux.com	Manish	burnam	CF202402003	manish_burnam@fosteringlinux.com	No	06/05/2023 04:46 PM	28/03/2024 02:44 PM																																									
manish_choudhary@fosteringlinux.com	Marsh	Choudhary	CF202402003	manish_choudhary@fosteringlinux.com	Yes	20/05/2023 03:41 PM	15/06/2024 03:42 PM																																									
manish_kumar@fosteringlinux.com	Marsh	Chetan	OK202802003	manish_kumar@fosteringlinux.com	Yes	18/03/2023 10:25 PM	26/03/2024 01:09 AM																																									
manish_kumar94@gmail.com	Marsh	Kumar	Pa	manish_kumar94@gmail.com	No	04/02/2024 08:04 PM	14/03/2024 08:10 PM																																									
manish_yadav@fosteringlinux.com	Manish	Yadav	CP2023111602	manish_yadav@fosteringlinux.com	No	06/05/2023 03:07 PM	30/03/2024 14:21 AM																																									
<b>When:</b> User navigate to the "Users" section																																																
<b>And:</b> User create a new user with the email "manish_choudhary@fosteringlinux.com" and other required details																																																
<b>Then:</b> user created successfully																																																
<b>Result:</b> Passed																																																

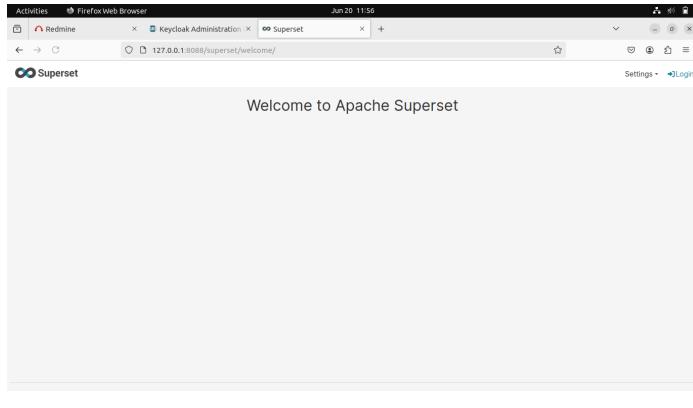
## Task: 20

Scenario: Authenticate with Keycloak and Google OAuth	
<b>Given:</b> User have logged out of Redmine	
<b>When:</b> User navigate to the Redmine login page	
<b>And:</b> User select "Continue with Keycloak"	
<b>Then:</b> User redirected to the Keycloak login page	
<b>When:</b> User select the Google authentication option in Keycloak	
<b>Then:</b> User redirected to the Google login page	

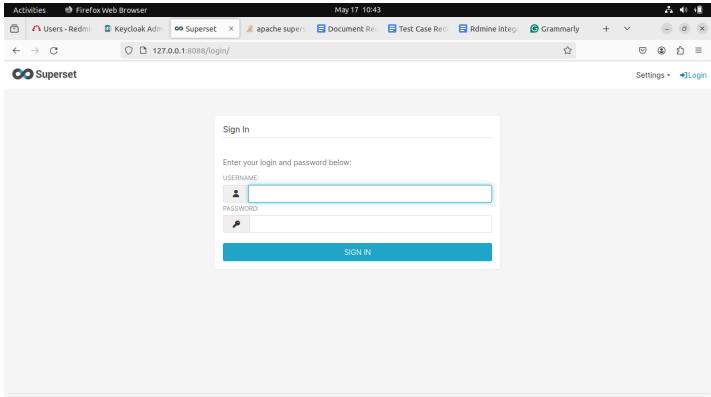
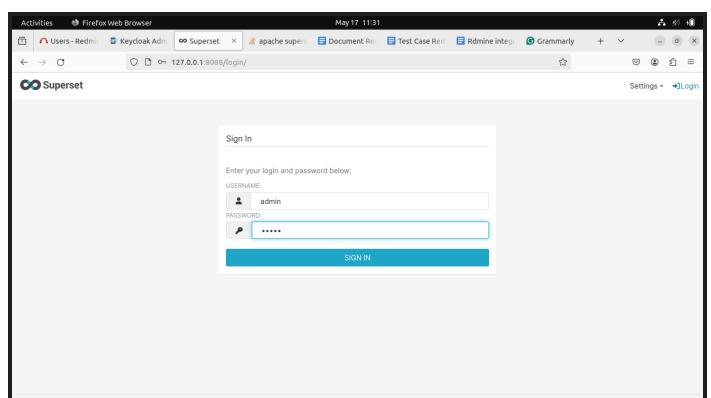
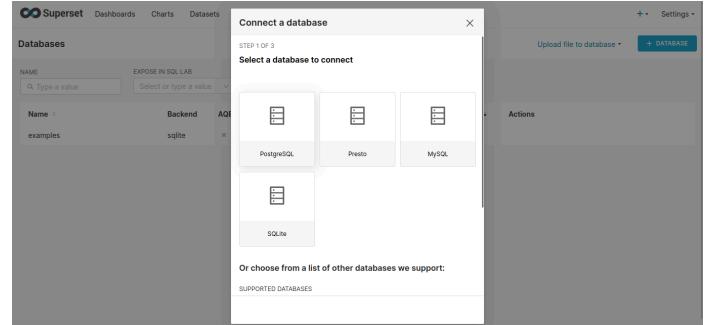
	 <p><b>When:</b> User log in with the email "manish_choudhary@fosteringlinux.com"</p>
	 <p><b>When:</b> User log in with the email "choudharym845@gmail.com"</p>
<p><b>Then:</b> User see an error indicating that the user does not exist in Redmine</p>	 <p><b>keycloak wants additional access to your Google Account</b></p> <p>keycloak already has some access See the 3 services that keycloak has some access to.</p> <p>Make sure you trust keycloak You may be sharing sensitive info with this site or app. Learn about how keycloak will handle your data by reviewing its terms of service and privacy policies. You can always see or remove access in your Google Account.</p> <p>Cancel Continue</p>
<p><b>Result: Passed</b></p>	 <p>KEENABLE</p> <p>We are sorry... Unexpected error when authenticating with identity provider</p>

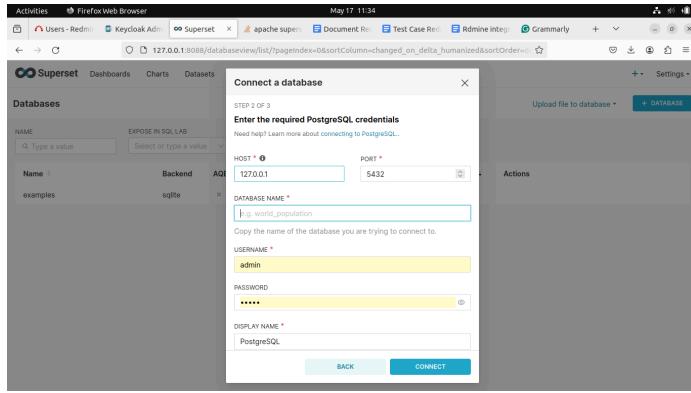
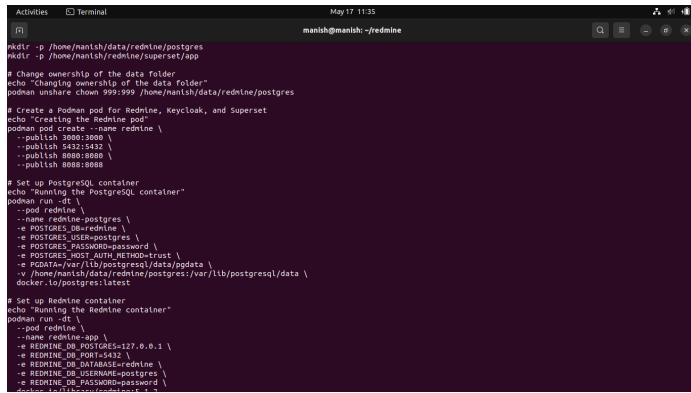
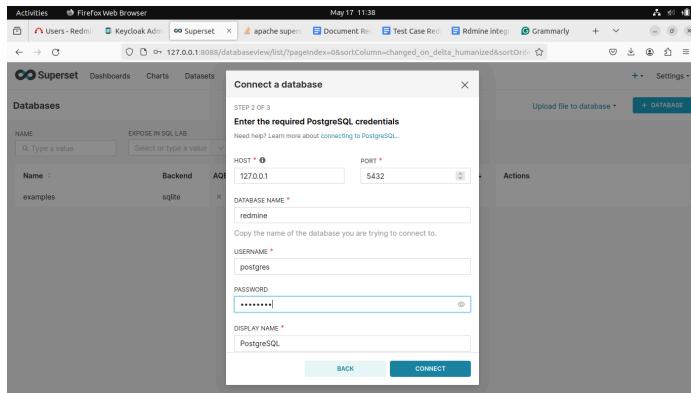
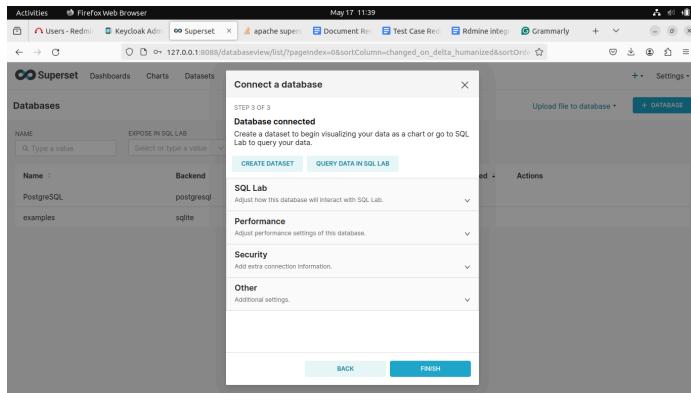
**Task: 21**

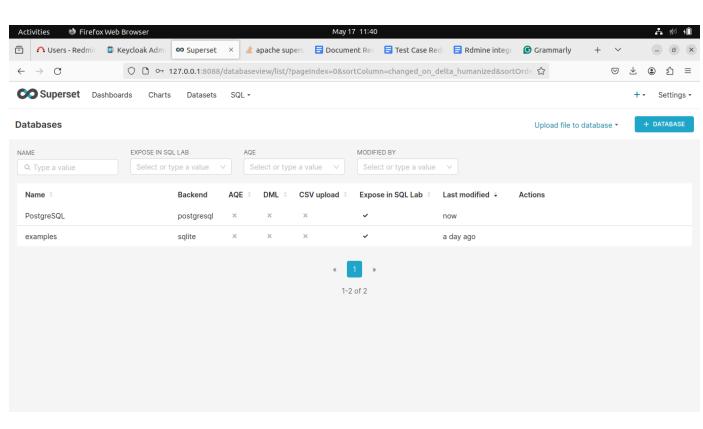
<b>Scenario: Build and run the Superset Docker image</b>	
<b>Given:</b> User have a Dockerfile for Superset	<b>Output:</b> <pre>manish@manish:~/redmine\$ cat Dockerfile # syntax=docker/dockerfile:1  FROM docker.io/apache/superset  # Switching to root to install the required packages USER root  # if you prefer Postgres, you may want to use `psycopg2-binary` instead RUN pip install psycopg2-binary RUN pip install flask-oidc==1.3.0 RUN pip install flask_openid RUN pip install python-keycloak RUN pip install itsdangerous==2.0.1  # Switching back to using the `superset` user USER superset  manish@manish:~/redmine\$</pre>
<b>When:</b> User run the command "podman build -t superset:manish ."	
<b>Then:</b> The Docker image for Superset built and tagged as "superset:manish"	
<b>When:</b> User run the command "podman images"	<pre>manish@manish:~/redmine\$ podman build -t superset:manish . STEP 1/8: FROM docker.io/apache/superset STEP 2/8: USER root --&gt; Using cache 30e175c728f443baa7df0aaa87167271e3cbbb6b0e13d5012272f452898fd5a --&gt; 30e175c728f STEP 3/8: RUN pip install psycopg2-binary --&gt; Using cache ddfbe70cb58ab327cd0ab2fcc9dad1168f84d451d6a7a1e56516e9cc51c99bd7 --&gt; ddfbe70cb58 STEP 4/8: RUN pip install flask-oidc==1.3.0 --&gt; Using cache d5ca2db9520f90e4ce86f92626284eea14dd3fe18119c325bf20aa08e40c3882 --&gt; d5ca2db9520 STEP 5/8: RUN pip install flask_openid --&gt; Using cache 6a5b581a1d1d1557ee3b98add0d7dd929f3567c3ddfa730787ae8d9efde4ba0fc --&gt; 6a5b581a1d STEP 6/8: RUN pip install python-keycloak --&gt; Using cache eea8bddcfe716d36f2cf54b4586e4730c6324a563231d9fb5533b17f28d5897 --&gt; eea8bddcfe7 STEP 7/8: RUN pip install itsdangerous==2.0.1 --&gt; Using cache 03a7cf64f9ed03a2731371565ba1ab3b07a2e7881bc4869db2bf2177960d5fd2 --&gt; 03a7cf64f9e STEP 8/8: USER superset --&gt; Using cache e8693c41fde86be0a0b04e37e6ef93986ff3accc76976a81134dcfd7d148150b COMMIT superset:manish --&gt; e8693c41fde Successfully tagged localhost/superset:manish e8693c41fde86be0a0b04e37e6ef93986ff3accc76976a81134dcfd7d148150b manish@manish:~/redmine\$</pre>
	<b>openssl rand -base64 42</b>
<b>Then:</b> User see the "superset:manish" image listed	"N/m/iGhhcsoadUnvIHMqHiRbApy7Z1CT9ymQXwb nBSQI50c7xNaH10+"
<b>When:</b> User generate a secret key with the command "openssl rand -base64 42"	<pre>manish@manish:~/redmine/superset/app\$ podman images REPOSITORY          TAG      IMAGE ID   CREATED    SIZE localhost/superset  manish   e8693c41fde8  13 days ago  1.38 GB quay.io/keycloak/keycloak  latest   07caf7ea07ac  2 weeks ago  462 MB docker.io/apache/superset  latest   bf71819e68c5  5 weeks ago  1.37 GB docker.io/library/postgres  latest   cff0b68a194a  5 weeks ago  439 MB docker.io/library/redmine  5.1.2    375ba4478aa8  2 months ago  653 MB k8s.gcr.io/pause        3.5     ed210e3e4a5b  3 years ago  690 kB manish@manish:~/redmine/superset/app\$</pre>
<b>Then:</b> User receive a secret key string	

<p><b>When:</b> User navigate to "http://127.0.0.1:8088" in a web browser</p>	
<p><b>Then:</b> The Superset login page displayed</p>	
<p><b>Result: Passed</b></p>	

## Task 22:

<b>Scenario: Create Superset admin user and initialise the database</b>	
<p><b>Given:</b> The Superset container is running</p>	<p><b>Output:</b></p> 
<p><b>When:</b> User run the command "podman exec -it superset superset fab create-admin --username admin --firstname Superset --lastname Admin --email admin@superset.com --password admin"</p>	
<p><b>Then:</b> Then an admin user with the username "admin" and password "admin" created</p>	
<p><b>When:</b> User run the command "podman exec -it superset superset db upgrade"</p>	
<p><b>Then:</b> The local database migrated to the latest version</p>	

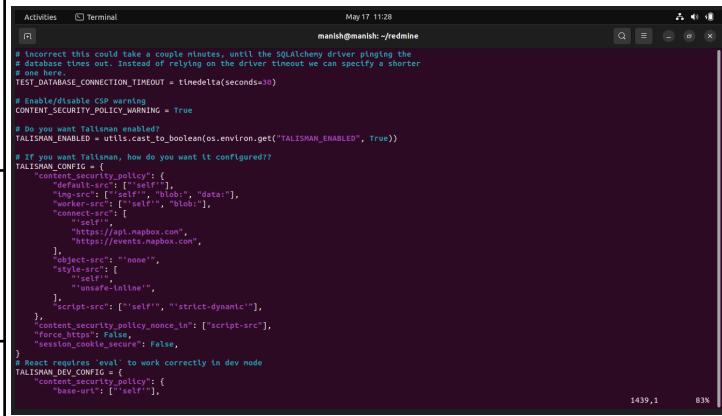
<p><b>When:</b> User run the command "podman exec -it superset superset init"</p>	
<p><b>Then:</b> The Superset instance initialised</p>	
<p><b>When:</b> User log in to Superset with the username "admin" and password "admin"</p>	
<p><b>Then:</b> User gain access to the Superset dashboard</p>	

Result: Passed


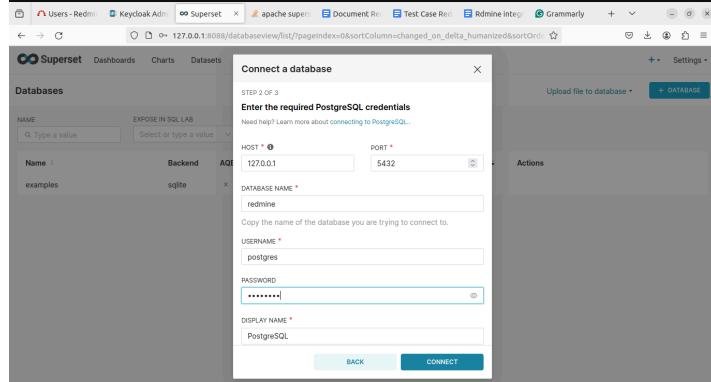
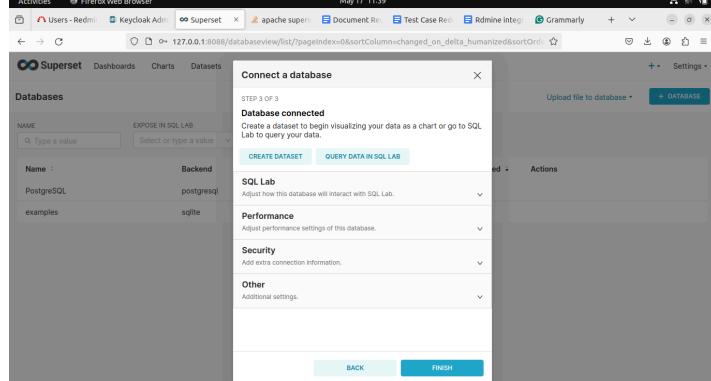
Name	Backend	AQE	DML	CSV upload	Expose in SQL Lab	Last modified	Actions
PostgreSQL	postgresql	x	x	x	✓	now	
examples	sqlite	x	x	x	✓	a day ago	

## Task: 23

Scenario: Update Superset configuration	
<p><b>Given:</b> User need to disable the CSP in Superset</p>	<p><b>Output:</b></p> <pre>manish@manish:~/redmine/superset/app\$ podman exec -it superset superset db upgrade Loaded your LOCAL configuration at [/app/pythonpath/superset_config.py] logging was configured successfully 2024-05-17 06:19:23,351:INFO:superset.utils.logging_configurator:logging was configured successfully 2024-05-17 06:19:23,355:INFO:root:Configured event logger of type &lt;class 'superset.utils.log.DBEventLogger'&gt; /usr/local/lib/python3.10/site-packages/flask_limiter/extension.py:293: UserWarning: Using the in-memory storage for tracking rate limits as no storage was explicitly specified. This is not recommended for production use. See: https://flask-limiter.readthedocs.io#configuring-a-storage-backend for documentation about configuring the storage backend. warnings.warn( WARNI [alembic.env] SQLite Database support for metadata databases will be removed in a future version of Superset. INFO [alembic.runtime.migration] Context impl SQLiteImpl. INFO [alembic.runtime.migration] Will assume transactional DDL.  manish@manish:~/redmine/superset/app\$ podman exec -it superset superset db upgrade Loaded your LOCAL configuration at [/app/pythonpath/superset_config.py] logging was configured successfully 2024-05-17 06:21:00,555:INFO:superset.utils.logging_configurator:logging was configured successfully 2024-05-17 06:21:00,560:INFO:root:Configured event logger of type &lt;class 'superset.utils.log.DBEventLogger'&gt; /usr/local/lib/python3.10/site-packages/flask_limiter/extension.py:293: UserWarning: Using the in-memory storage for tracking rate limits as no storage was explicitly specified. This is not recommended for production use. See: https://flask-limiter.readthedocs.io#configuring-a-storage-backend. _ warnings.warn( WARNI [alembic.env] SQLite Database support for metadata databases will be removed in a future version of Superset.</pre>
<p><b>When:</b> User run the command "podman exec -it -u root superset bash"</p>	
<p><b>And:</b> User run "apt-get update" and "apt install vim" inside the container</p>	
<p><b>And:</b> User navigate to the "superset/" directory and edit "config.py" using "vim"</p>	
<p><b>And:</b> User set "TALISMAN_ENABLED" to "False" in "config.py"</p>	

<p><b>And:</b> User save the changes and exit</p>	 <pre> Activities Terminal May 17 11:28 manish@manish:~/redmine # Uncorrect this could take a couple minutes, until the SQLAlchemy driver pings the # database times out. Instead of relying on the driver timeout we can specify a shorter # one here. TEST_DATABASE_CONNECTION_TIMEOUT = timedelta(seconds=30)  # Enable/disable CSP warning CONTENT_SECURITY_POLICY_WARNING = True  # Do you want Talisman enabled? TALISMAN_ENABLED = utils.cast_to_boolean(os.environ.get("TALISMAN_ENABLED", True))  # If you want Talisman, how do you want it configured? TALISMAN_CONFIG = {     "content_security_policy": [         "default-src": ["'self'"],         "img-src": ["blob:", "data:"],         "script-src": ["'self'", "blob:"],         "connect-src": [             "'self'",             "https://api.mapbox.com",             "https://events.mapbox.com",         ],         "object-src": "none",         "style-src": [             "'self'",             "'unsafe-inline'",         ],         "script-src": ["'self'", "'strict-dynamic'"],         "content_security_policy_nonce_in": ["script-src"],         "report-uri": "/csp-report",         "session_cookie_secure": False,     ] } # Exec requires `eval` to work correctly in dev mode TALISMAN_DEV_CONFIG = {     "content_security_policy": [         "base-uri": ["'self'"],     ] } </pre>
<p><b>And:</b> User restart the Superset container with "podman restart superset"</p>	
<p><b>Then:</b> Then the CSP should be disabled</p>	
<p><b>When:</b> User navigate to "http://127.0.0.1:8088" in a web browser</p>	
<p><b>Then:</b> The Superset login page should be displayed</p>	
<p><b>Result:</b> Passed</p>	

## Task: 24

<b>Scenario: Connect Superset to PostgreSQL</b>	
<b>Given:</b> PostgreSQL is running and configured with the details from "redmine.sh"	<b>Output:</b> 
<b>When:</b> User log in to Superset and navigate to the "Settings" section	
<b>And:</b> User add a new database connection	
<b>And:</b> User select "Postgres" as the database type	
<b>And:</b> User fill in the connection details (hostname, port, database name, username, and password)	
<b>Then:</b> The PostgreSQL database connected successfully	
<b>Result:</b> Passed	



Task: 25

**Scenario: Configure Superset for Keycloak SSO**

**Given:** User have created the "client\_secret.json" file with Keycloak client details

**And:** User have created the "keycloak\_security\_manager.py" file with custom security manager code

**And:** User have created the "superset\_config.py" file with Keycloak configuration

**When:** User navigate to the Superset application directory

**And:** User update the database with "podman exec -it superset superset db upgrade"

**And:** User initialise the Superset instance with "podman exec -it superset superset init"

**And:** User restart the Superset container with "podman restart superset"

**Then:** Superset configured to use Keycloak for authentication

## Output:

```
Activities ☐ Terminal manish@manish:~/redmine/superset/app
root@manish:~/redmine/superset/app$ cat client_secret.json
{
  "web": {
    "issuer": "http://127.0.0.1:8080/realm/keenable/broker/google/endpoint",
    "audience": "http://127.0.0.1:8080/realm/keenable/protocol/openid-connect/auth",
    "client_id": "manish",
    "client_secret": "blob:ix18Afkq4InyGzAHMg13kQASzv",
    "redirect_uris": [
      "http://127.0.0.1:8088"
    ],
    "userinfo_url": "http://127.0.0.1:8080/realm/keenable/protocol/openid-connect/userinfo",
    "token_url": "http://127.0.0.1:8080/realm/keenable/protocol/openid-connect/token",
    "token_introspection_url": "http://127.0.0.1:8080/realm/keenable/protocol/openid-connect/token/introspect"
  }
}

root@manish:~/redmine/superset/app$ cat keycloak_security_manager.py
from flask import redirect, request, flash
from flask_appbuilder.security.manager import AUTH_OID
from superset.security import SupersetSecurityManager
from flask_oidc import OpenIDConnect
from flask_appbuilder.security.views import AuthOIDView
from flask_login import login_user
from urllib.parse import quote
from flask_appbuilder.views import expose
import logging

class AuthOIDView(AuthOIDView):
    @expose('/login', methods=['GET', 'POST'])
    def login(self, flag=True):
        if self._handle_login():
            logging.debug("Attempting to authenticate user via OIDC.")
            email = self.appbuilder.sm.auth_user_getfield('email')
            user = self.appbuilder.sm.find_user(email)
            if user:
                self._handle_login_for(user)
            else:
                logging.error("User not found for email (%s)" % email)

        if self._handle_login():
            logging.debug("Attempting to authenticate user via OIDC.")
            email = self.appbuilder.sm.auth_user_getfield('email')
            user = self.appbuilder.sm.find_user(email)
            if user:
                self._handle_login_for(user)
            else:
                logging.error("User not found for email (%s)" % email)

        if not user:
            info = self.appbuilder.sm.auth_user_getinfo(['preferred_username', 'given_name', 'family_name', 'email'])
            user = self.appbuilder.sm.add_user(info['given_name'], info['family_name'],
                                              info['preferred_username'], info.get('email'))
            self._handle_login_for(user)
            logging.debug("Created new user: (%s)" % user)

        if isinstance(user, UserModel):
            # Ensure 'user' is an instance of the user model
            login_user(user, remember=False)
            logging.info("User (%s) logged in successfully." % user.username)
            return redirect(self.appbuilder.get_url_for_index)
        else:
            logging.error("User authentication failed. Invalid user object.")
            flash("Authentication failed.")

    return self._handle_login()

    @expose('/logout', methods=['GET', 'POST'])
    def logout(self):
        self._handle_logout()
        self.appbuilder.sm.logout()
        redirect_url = request.url_root + self.appbuilder.get_url_for_login
        return redirect(self.appbuilder.get_url_for_logout?url=quote(redirect_url))

class OIDCSecurityManager(SupersetSecurityManager):
    authoidview = AuthOIDView

    def __init__(self, appbuilder):
        super().__init__(appbuilder)
        if self.auth_type == AUTH_OID:
            self.old = OpenIDConnect(self.appbuilder.get_app)
```

```
Activities ☐ Terminal manish@manish:~/redmine/superset/app
root@manish:~/redmine/superset/app$ cat superset_config.py
from keycloak.security_manager import OAuthSecurityManager
from flask_appbuilder.security.manager import AUTH_OID
import os

#-----KEYCLOAK-----
curr = os.path.abspath(os.getcwd())
AUTH_TYPE = AUTH_OID
REDIRECT_URI = curr + "/superset/login/callback"
OIDC_CLIENT_SECRET = os.path.join(curr, 'pythonpath', 'client_secret.json')
OIDC_ID_TOKEN_COOKIE_SECURE = False
OIDC_OPENID_PROVIDERS = []
OIDC_OPENID_REALM = 'keenable'
OIDC_INTROSPECTION_AUTH_METHOD = 'client_secret_post'
OIDC_SCOPES = ['openid', 'profile']
AUTH_USER_REGISTRATION = True
AUTH_USER_REGISTRATION_ROLE = 'Public'

#-----
SUPERSET_WEBSERVER_DOMAINS = ['http://@fosteringlinux.com, http://@keenable.in']
ENABLE_PROXY_FIX = True
SUPERSET_WEBSERVER_PROTOCOL = "http"
SUPERSET_WEBSERVER_PORT = 8888
PUBLIC_ROLE_LIKE = "gamma"

#-----Emails for email (if needed)-----
# MAIL_SERVER = "smtp.your-mail.com"
# MAIL_PORT = 587
# MAIL_USE_TLS = True
# MAIL_USERNAME = 'your-email@example.com'
# MAIL_PASSWORD = 'your-email-password'
# MAIL_DEFAULT_SENDER = 'your-email@example.com'
# ...

# Additional Superset configurations
```

**Result: Passed**

## Task: 26

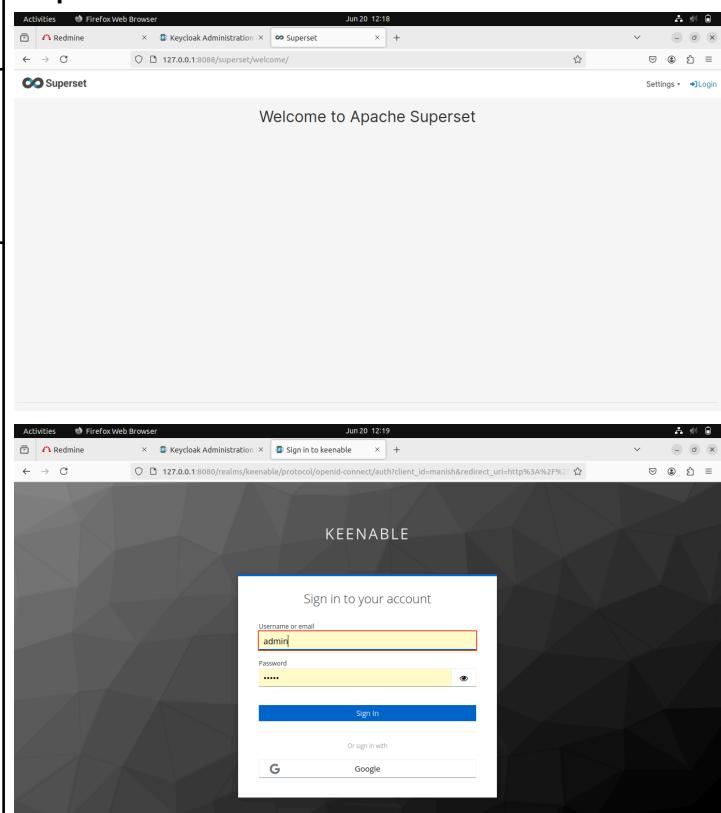
### Scenario: Login to Superset via Keycloak

**Given:** The Superset container is running and configured for Keycloak

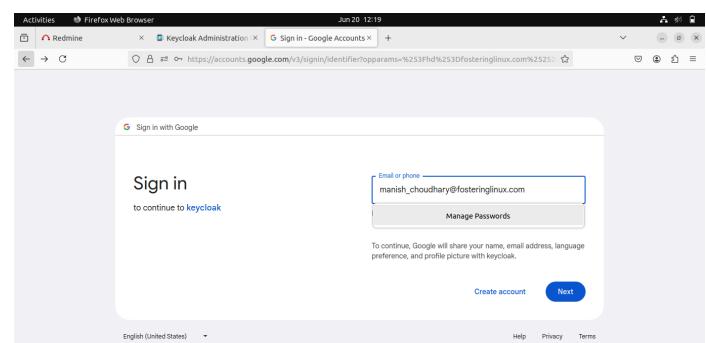
**When:** User navigate to "http://127.0.0.1:8088" in a web browser

**Then:** User redirected to the Keycloak sign-in page

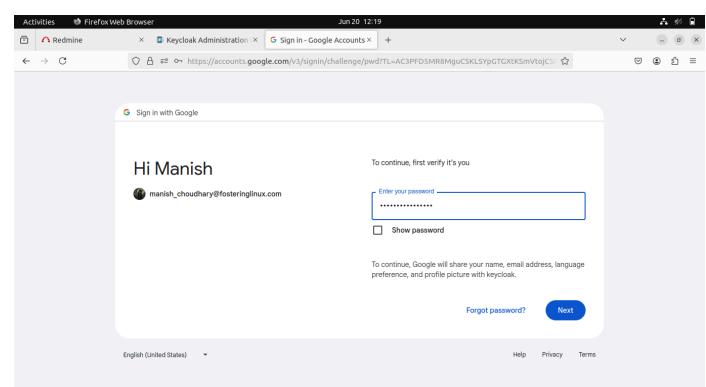
#### Output:



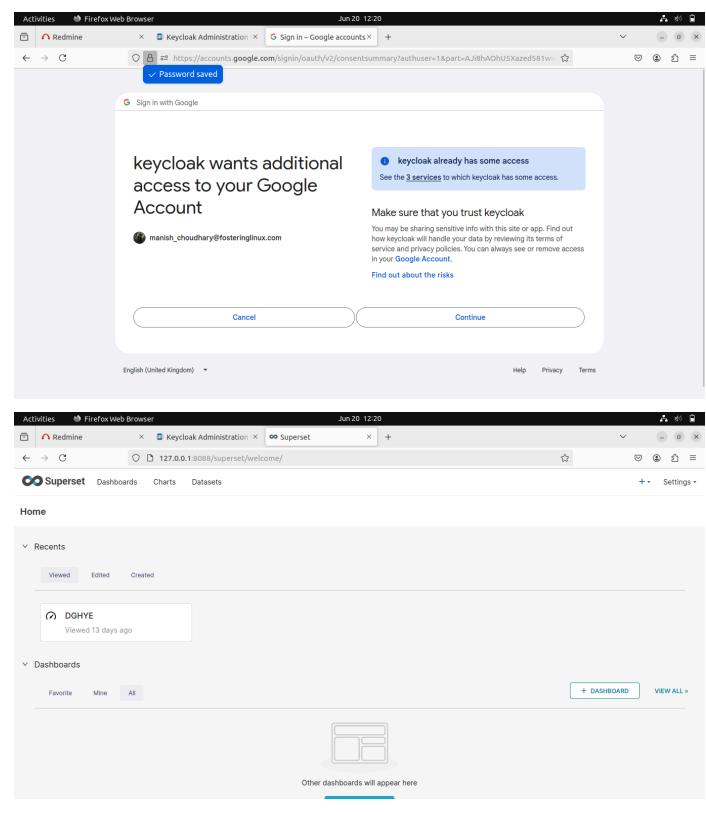
**When:** User click on the Google login option



**And:** User authenticate using my Google account



**Then:** User redirected back to Superset and be logged in



<b>Result: Passed</b>	
-----------------------	--

### Task: 27

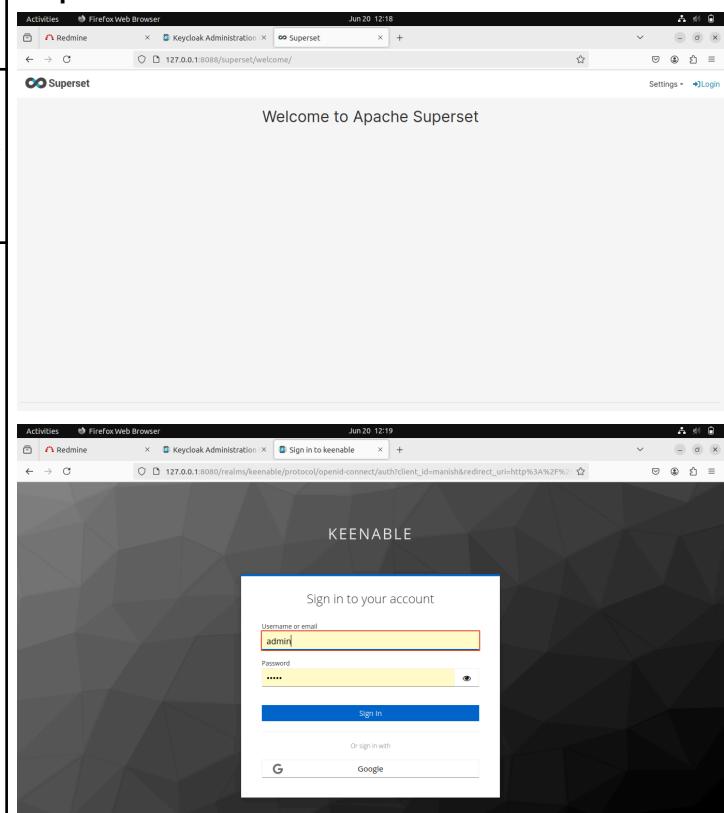
#### Scenario: Test SSO Integration with Redmine and Superset

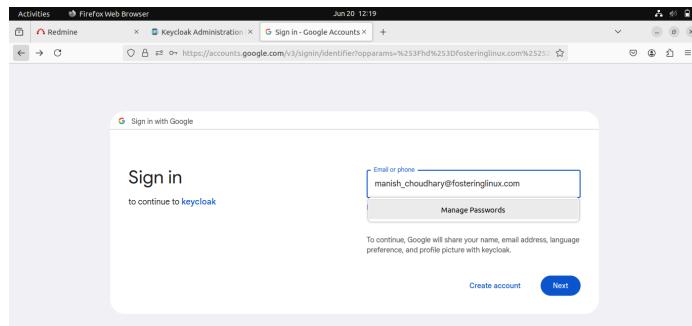
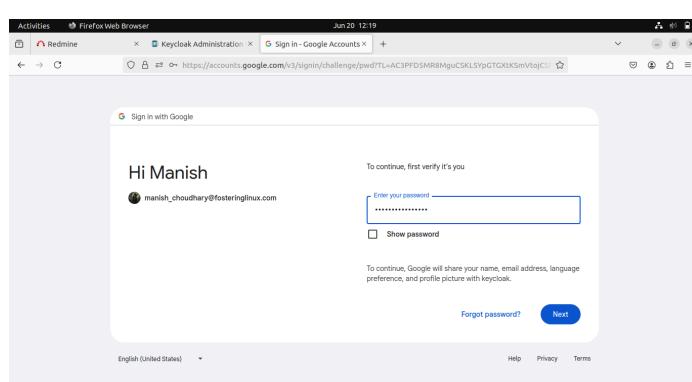
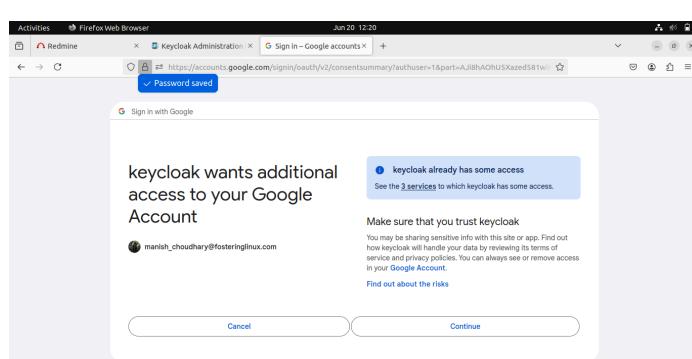
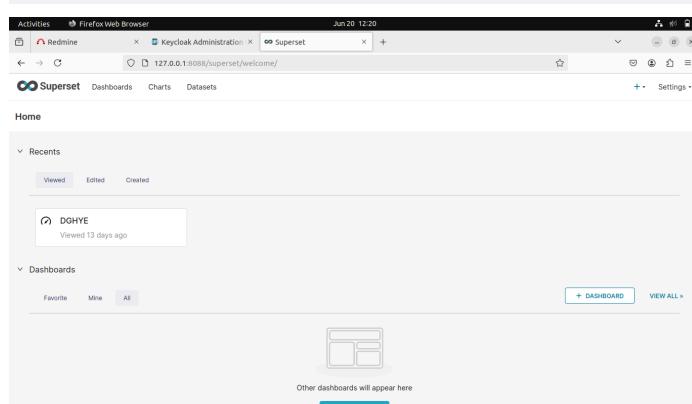
**Given:** Both Redmine and Superset are configured to use Keycloak for SSO

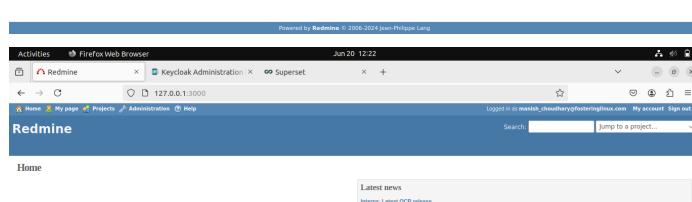
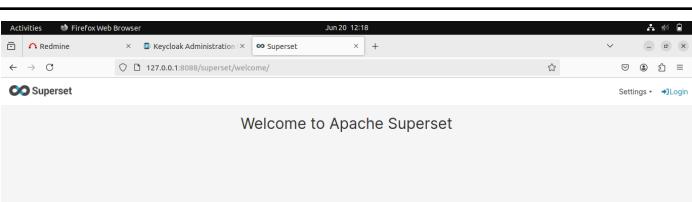
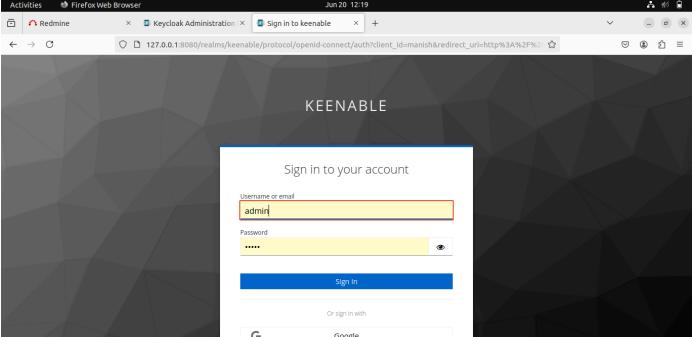
**When:** I log in to Redmine with Keycloak and Google authentication

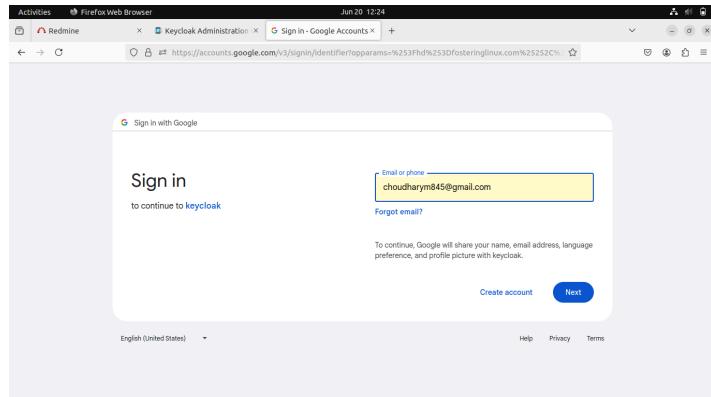
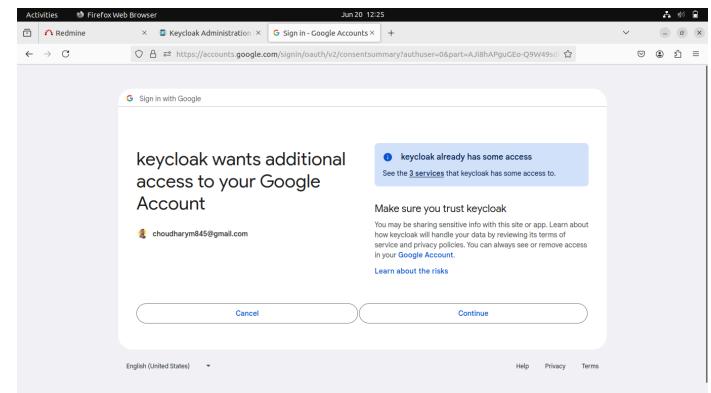
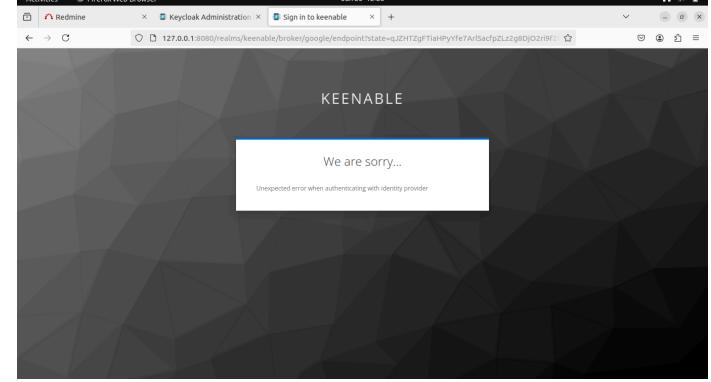
**Then:** User logged in successfully if the user exists in Redmine

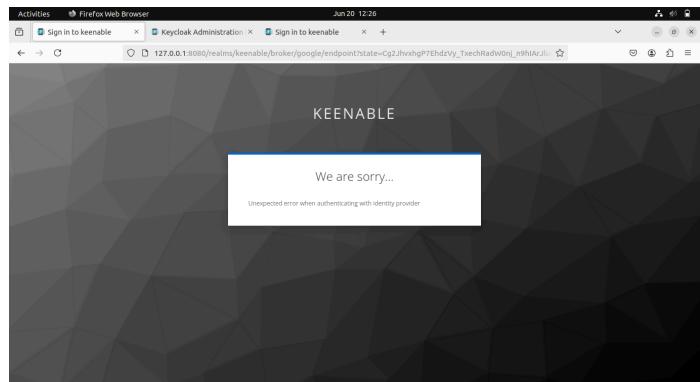
#### Output:



<p><b>When:</b> User log in to Superset with Keycloak and Google authentication</p>	
<p><b>Then:</b> User logged in successfully if the user exists in Superset</p>	
<p><b>When:</b> User try to log in to Redmine with an email not registered in Redmine</p>	
	

	
<p><b>Then:</b> User see an authentication error</p>	
<p><b>When:</b> User try to log in to Superset with an email not registered in Superset</p>	
<p><b>Then:</b> User see an authentication error</p>	

	 <p>The screenshot shows the Google sign-in page. A yellow box highlights the email input field containing "choudharym845@gmail.com". Below the input field are "Create account" and "Next" buttons.</p>
<b>Then:</b> User logged in successfully	 <p>The screenshot shows the Google consent screen. It asks for additional access for "keycloak" and provides a link to see the services. Buttons for "Cancel" and "Continue" are at the bottom.</p>
<b>When:</b> User try to log in to Superset with the email "manish_choudhary@fosteringlinux.com"	 <p>The screenshot shows the Superset sign-in page. A message box says "We are sorry... Unexpected error when authenticating with identity provider".</p>
<b>Then:</b> User logged in successfully	
<b>When:</b> User try to log in to Redmine with the email "choudharym845@gmail.com"	

	
<b>Then:</b> User see an authentication error	
<b>When:</b> User try to log in to Superset with the email "choudharym845@gmail.com"	
<b>Then:</b> User see an authentication error	
<b>Result: Passed</b>	