Auth0 Configuration for Spring Boot Application

Introduction:

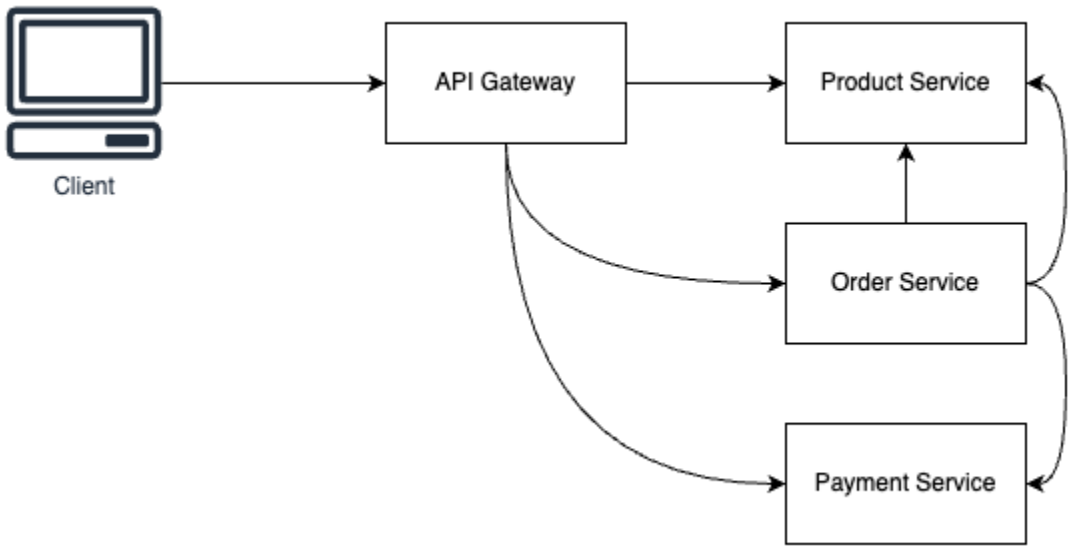
For Auth0 Configuration, you need to create an Auth0/Okta account. Go to the below URL to create your developer account with the free tier.

Got to : https://auth0.com/api/auth/login?redirectTo=dashboard

Once the account is created, you must configure the Auth0 dashboard to work with your Spring Boot Microservices.

Below is the architecture of Spring Boot Microservices we are working with.

Architecture:



In this Architecture, you can see we have multiple services. I have only included the service that needs Security integration.

1. API gateway → This has the API to login to the system and return the access token to be used to invoke other services

2. Product Service → This has the API to manage Products
3. Payment Service → This has the API to manage Payments
4. Order Service → This has the API to manage Orders. Internally, this Service will also invoke the API's of the Product Service and the Payment Service.

Spring Security works with different components:

1. 0Auth Client → This is the Client Application, which interacts with Auth0 and Authenticates and authorizes the User and generates the token.
2. 0Auth Resource → This is the Resource that serves the resources (API). It can validate the token with proper authorization and allow access.

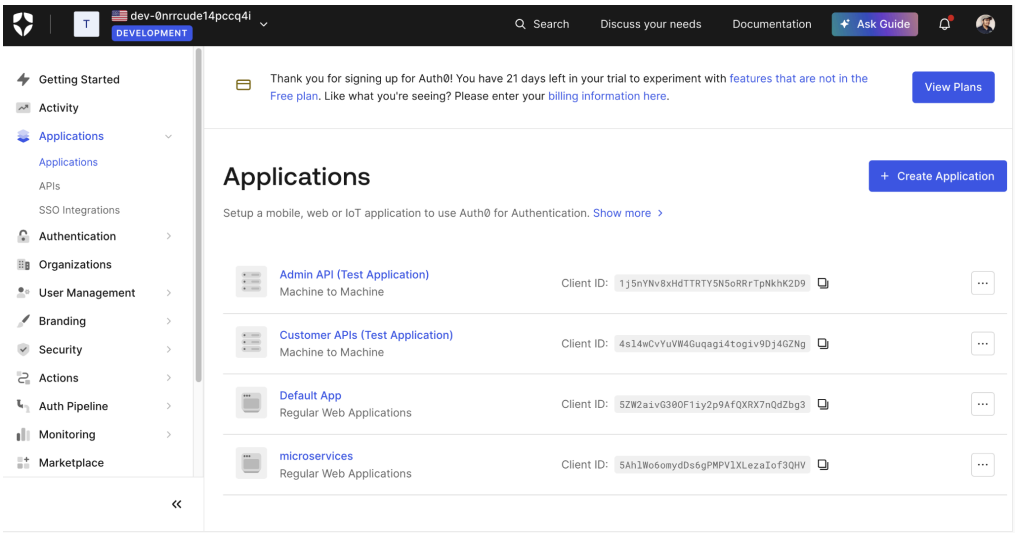
As per above Architecture

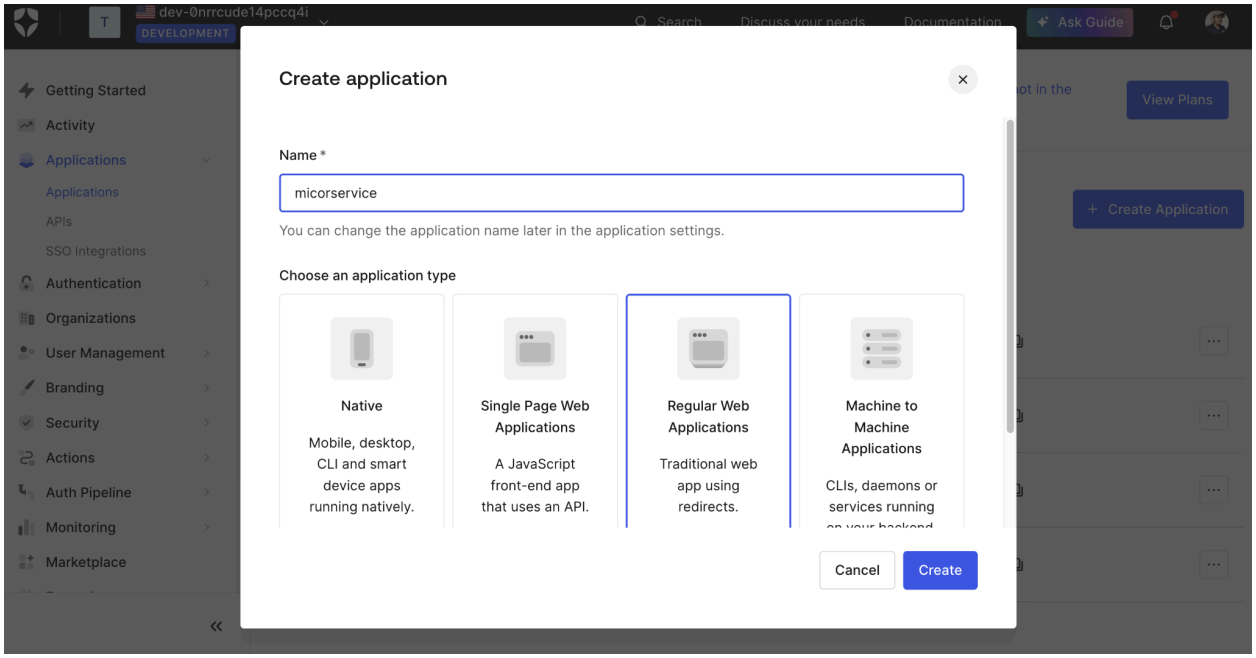
Below are the details containing which security component should be added to each service.

Service	Component
API Gateway	Client, Resource
Product Service	Resource
Order Service	Resource
Payment Service	Resource

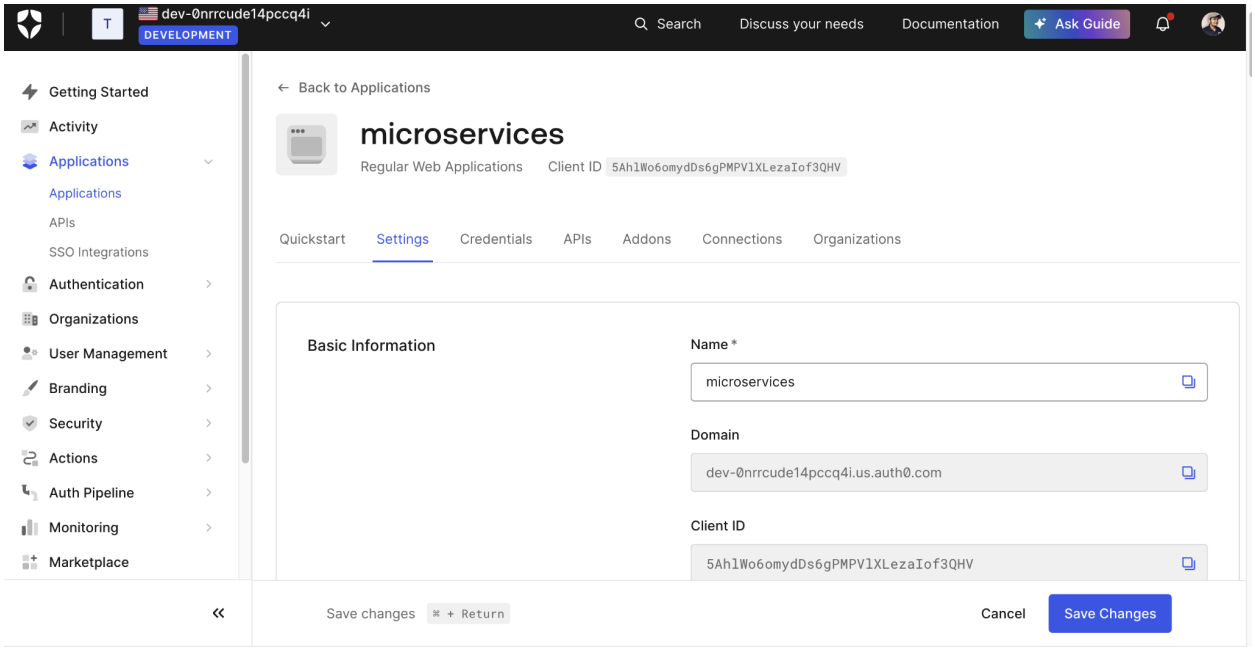
Auth0 dashboard Configuration

- Login to Auth0 dashboard and go to Applications → Applications
- Click on Create Application

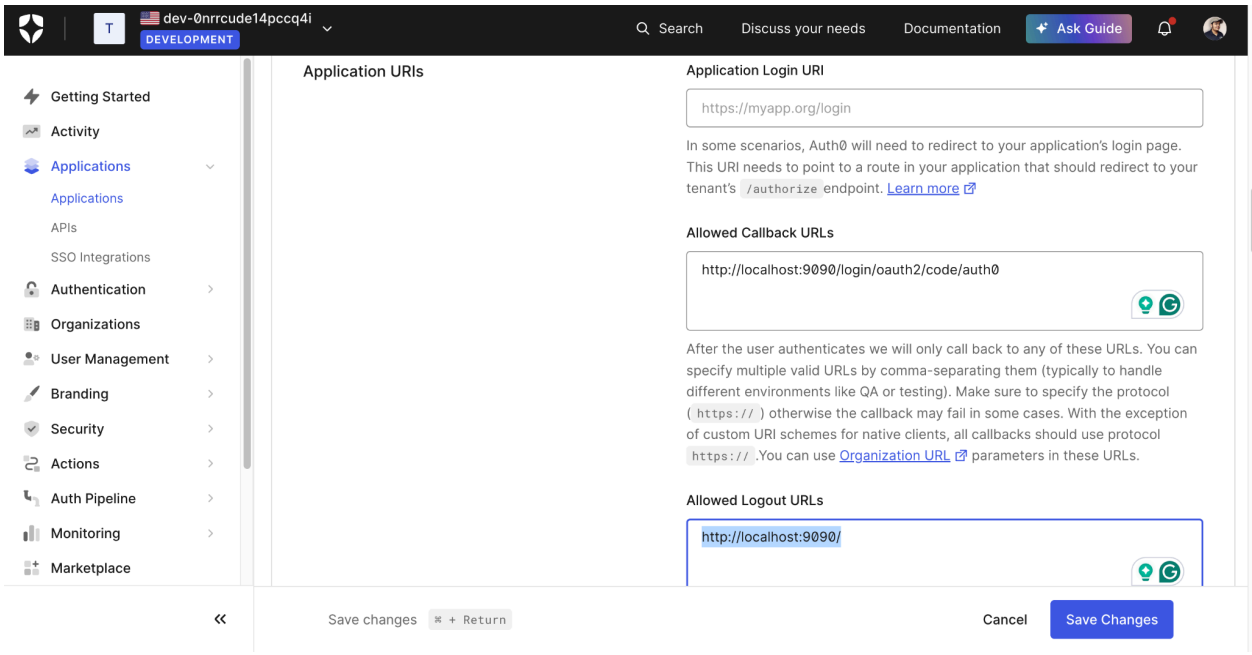




Once the application is created, go to the application for managing the settings.



- Now once the application is created, you need to configure the application URLs
- Configure Application callback/redirect URL
    - http://localhost:9090/login/oauth2/code/auth0
- Configure logout URL
    - http://localhost:9090/



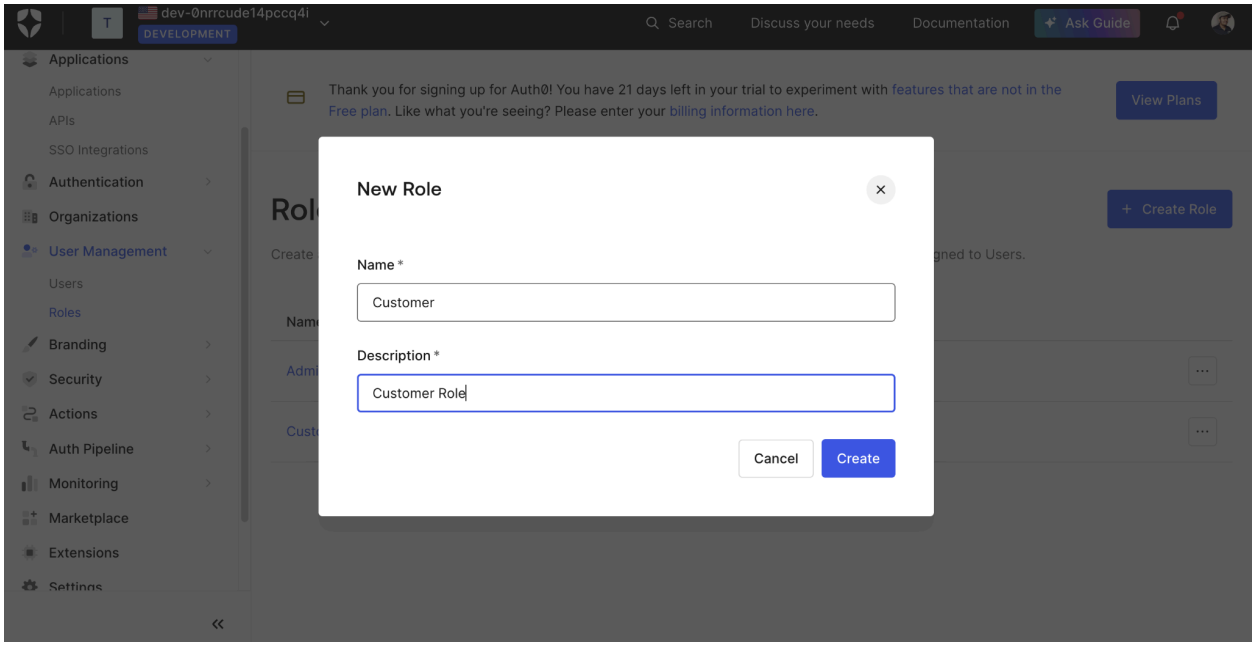
Save the application by clicking Save Changes.

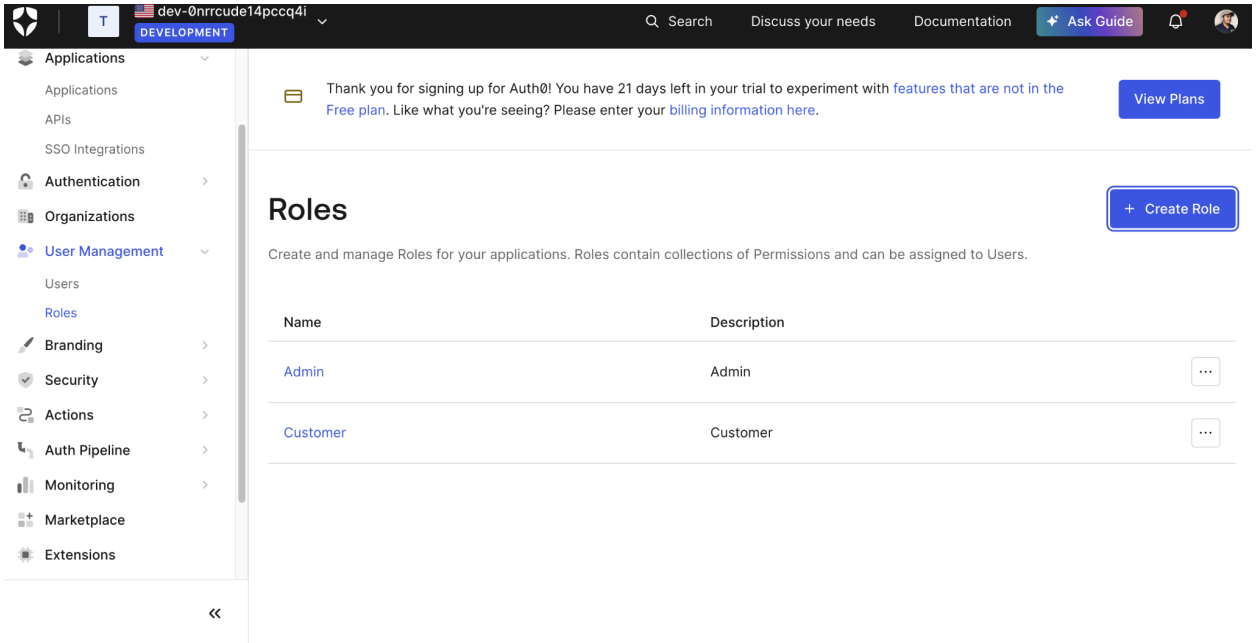
Create Users and Roles

Once our application is created, we need Users and Roles, which help to log in to the application.

Create Roles

- Roles allow us to have Role-based Authentication control for our application.
- Customer Role:
    - This Role allows the user to be a Customer and allows the application to: Place an Order, View Order, and View Products.
- Admin Role:
    - This role is for Admin, which helps to maintain the application.
    - To maintain Products.
    - To maintain the Payment reports
    - And other management related APIs, which the Customer should not have access to.
- In Auth0, each Role should be tied to the API (Audience), which helps the application to decide what permissions the user has assigned.
- Then the Role should be tied up to Users.
- Go to User Management → Roles to create new Roles.
- Click on Crate Role to create a new Role.

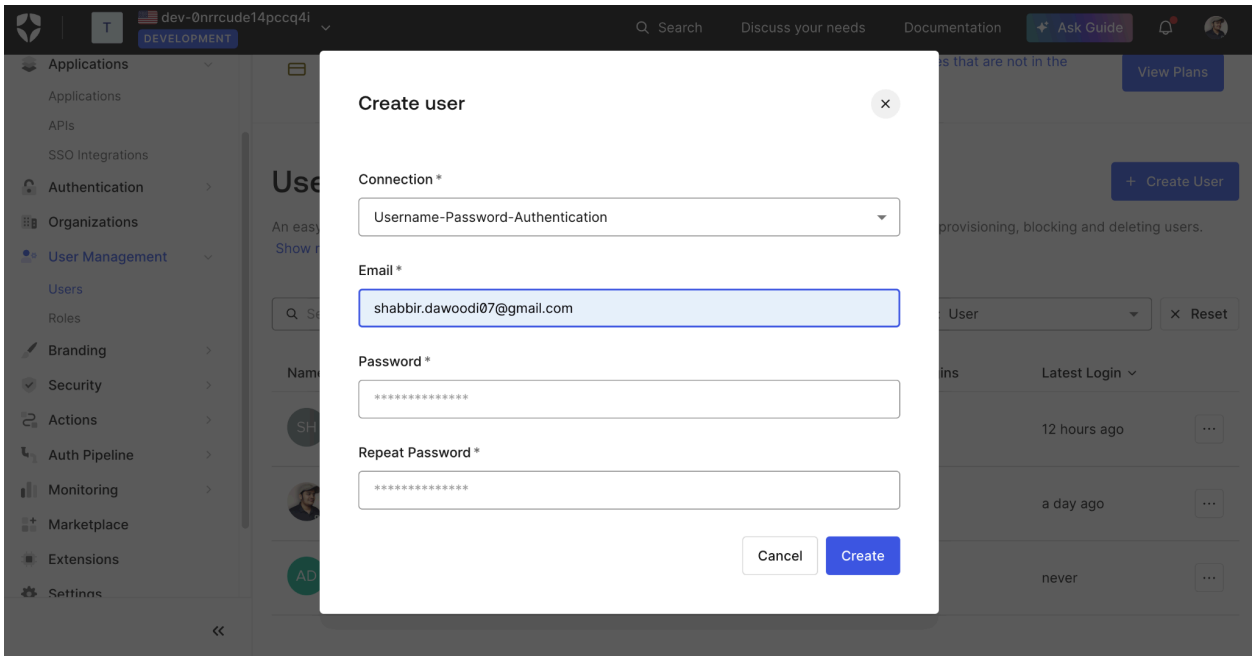




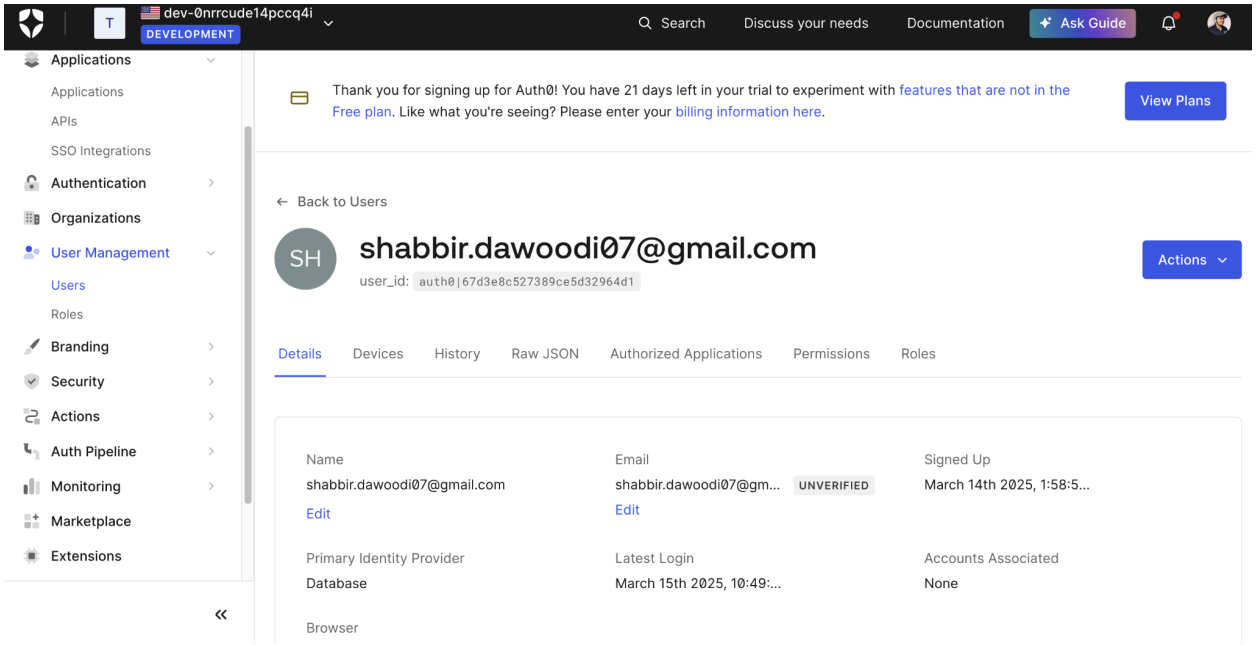
Create User

- Go to User Management → Users to create Users.

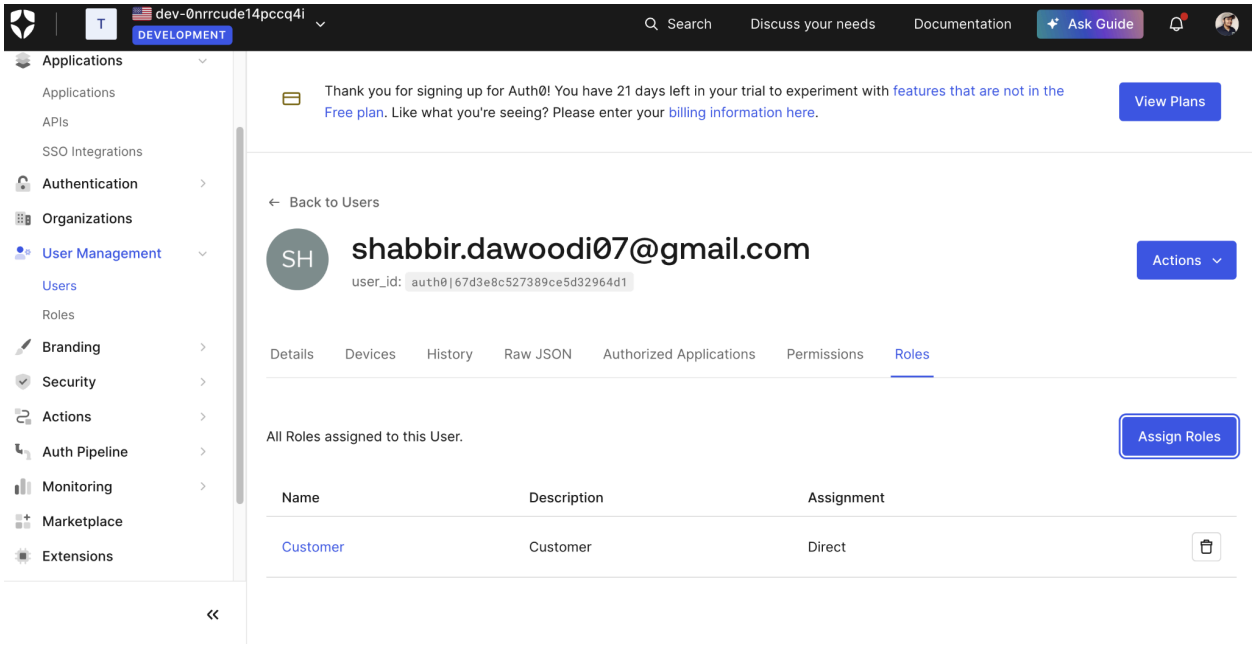
- Click on Create User to create a new User.



Click on the User created to assign Permission and Roles.



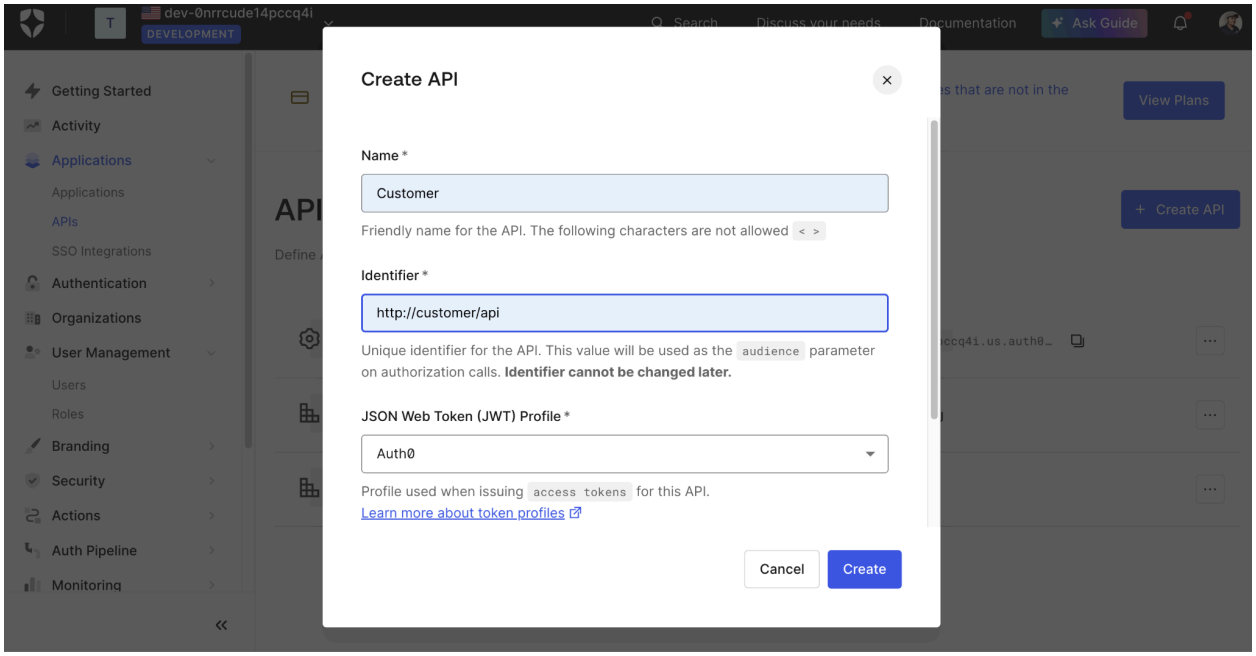
- Go to the Roles Section to assign a Role to the User created.
- For this User, assign the Customer Role as shown in the below Screenshot.

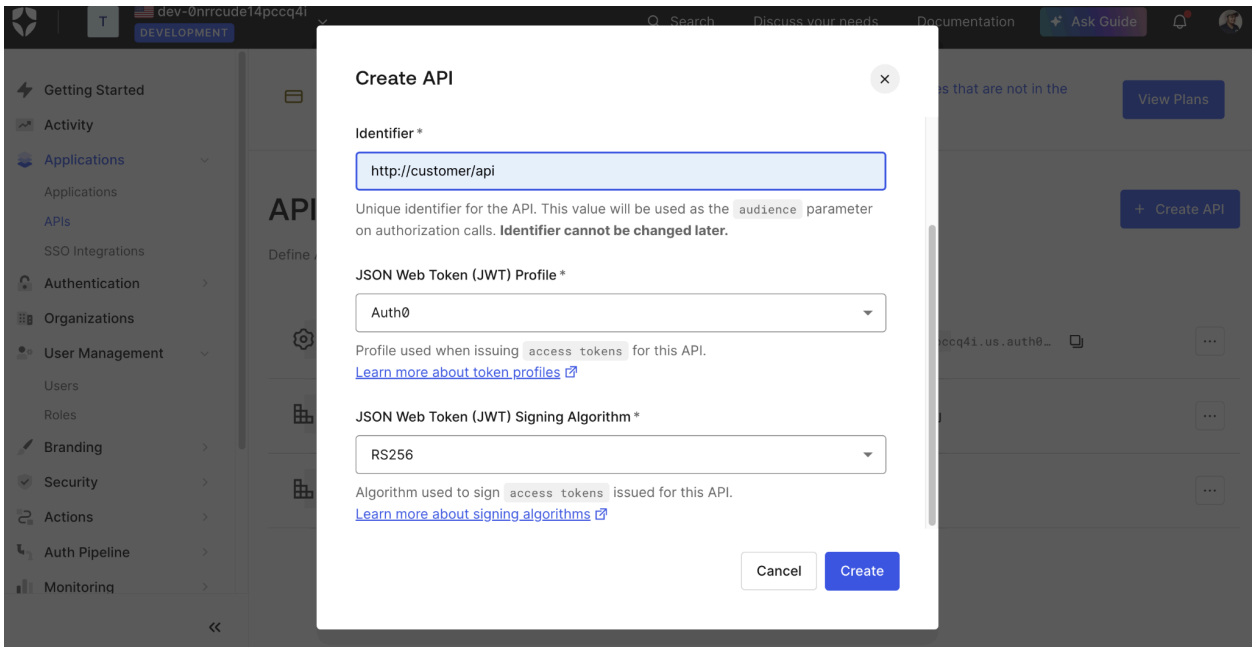


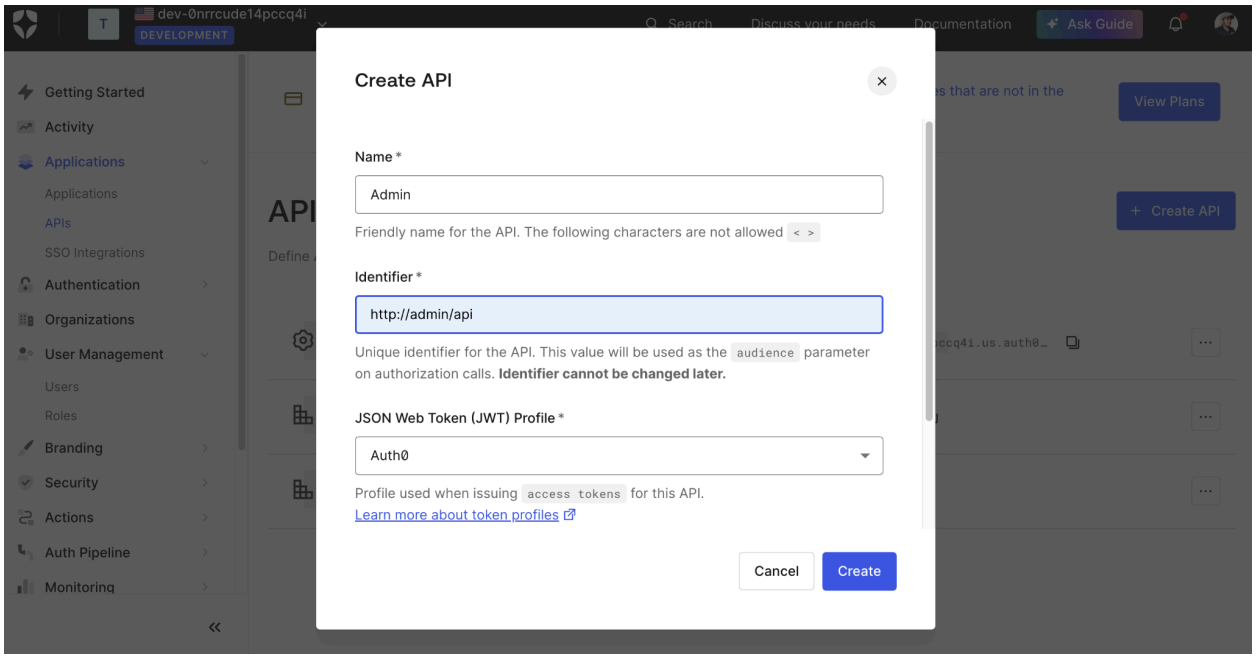
- Now, we have assigned the Roles to the User
- This User should also have the Permissions assigned.
- Permissions allow the particular User to use the different API's
- Create an API (Audience) that defines the permissions and add those permissions to users.

Create API

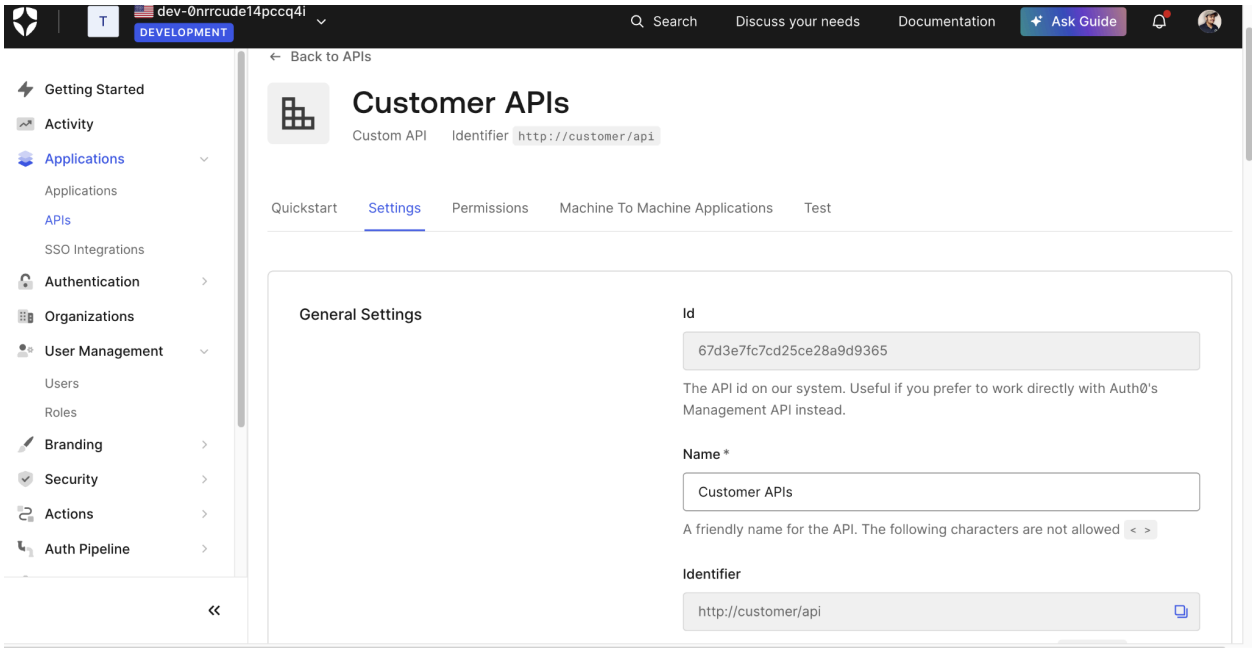
- Let's create 2 API's for different purposes of the application
- Customer API to have customer-related permissions
- Admin API to have admin-related permissions
- Go to Applications → APIs to create the API
- Click on Create API to create a new API
- Give a unique name to the API
- Give a unique identifier of the API. Give an Endpoint which is unique, it may not be a valid endpoint.
- Keep JSON Web Token Profile as "Auth0"
- Keep Signing Algorithm as Rs256
- Click on Create to create an API
- Repeat the process for Admin API

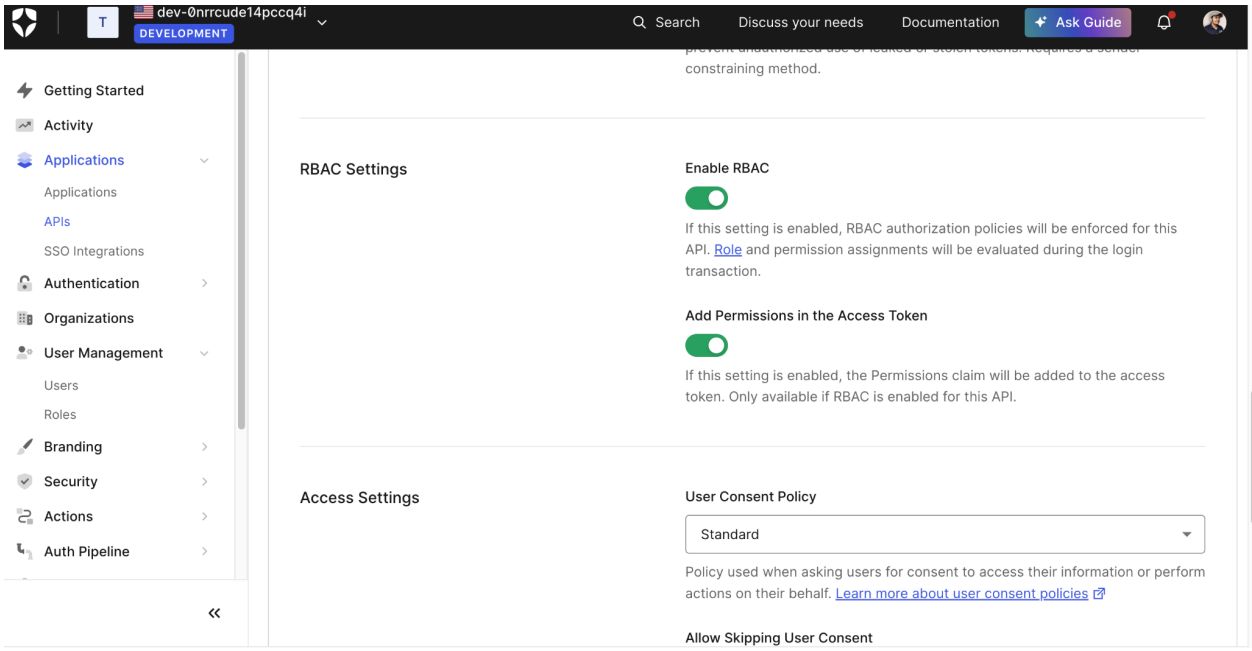

dev-0nrrcude14pccq4i
DEVELOPMENT
Search
Discuss your needs
Documentation
Ask Guide
Getting Started
Activity
Applications
Applications
APIs
SSO Integrations
Authentication
Organizations
User Management
Users
Roles
Branding
Security
Actions
Auth Pipeline
Monitoring
View Plans
Create API
Create API
Name *
Customer
Friendly name for the API. The following characters are not allowed < >
Identifier *
http://customer/api
Unique identifier for the API. This value will be used as the audience parameter on authorization calls. Identifier cannot be changed later.
JSON Web Token (JWT) Profile *
Auth0
Profile used when issuing access tokens for this API.
Learn more about token profiles
Cancel
Create

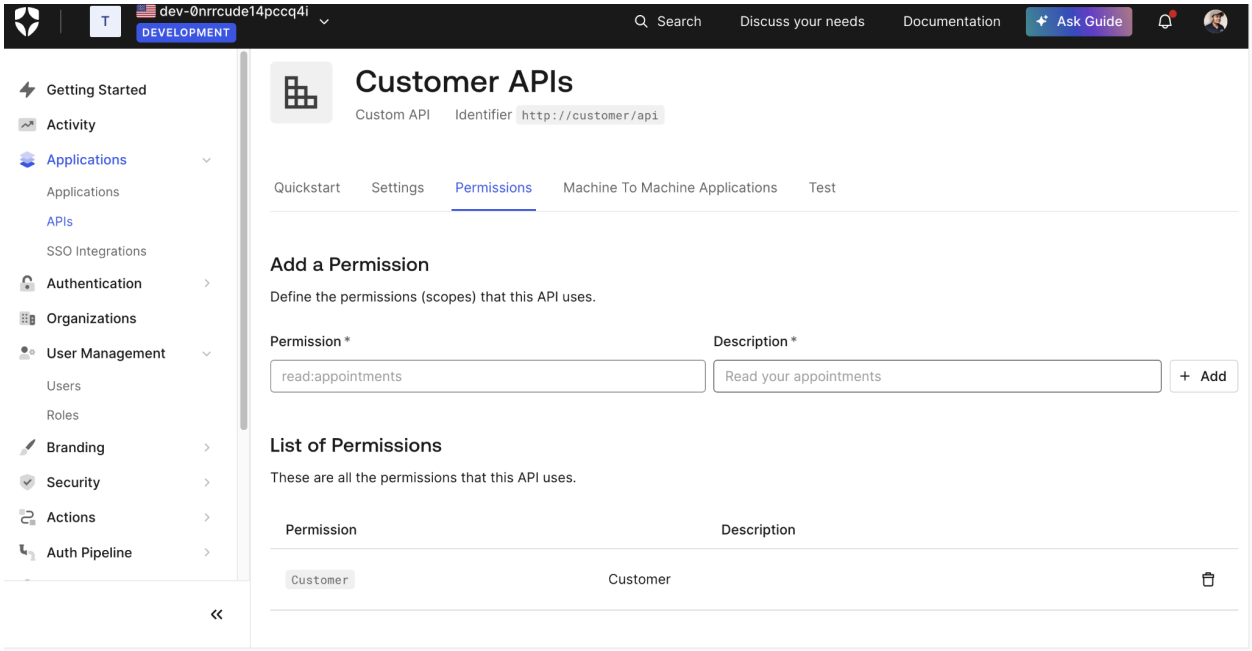

dev-0nrrcude14pccq4i
DEVELOPMENT
Search
Discuss your needs
Documentation
Ask Guide
Getting Started
Activity
Applications
Applications
APIs
SSO Integrations
Authentication
Organizations
User Management
Users
Roles
Branding
Security
Actions
Auth Pipeline
Monitoring
View Plans
Create API
Create API
Identifier *
http://customer/api
Unique identifier for the API. This value will be used as the audience parameter on authorization calls. Identifier cannot be changed later.
JSON Web Token (JWT) Profile *
Auth0
Profile used when issuing access tokens for this API.
Learn more about token profiles
JSON Web Token (JWT) Signing Algorithm *
RS256
Algorithm used to sign access tokens issued for this API.
Learn more about signing algorithms
Cancel
Create



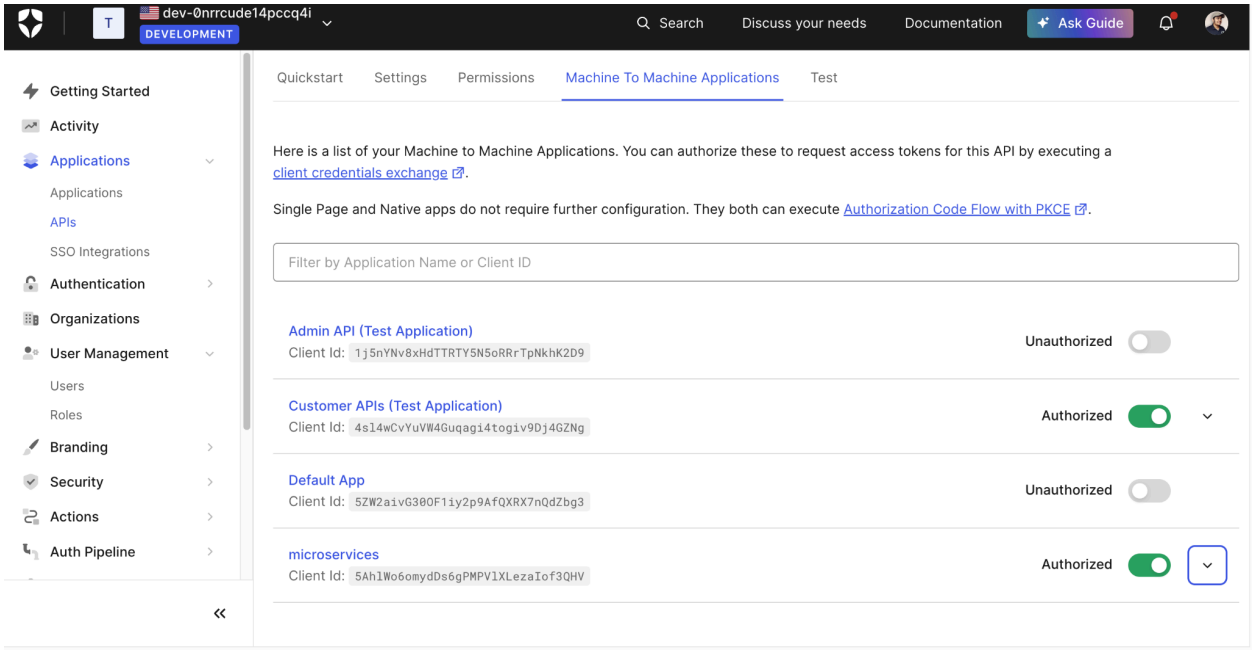
- For each API, enable Role-based Access Control (RBAC)
- Enable RBAC and add permissions in the Access token
- Now, For Each API created, we need to define permissions.

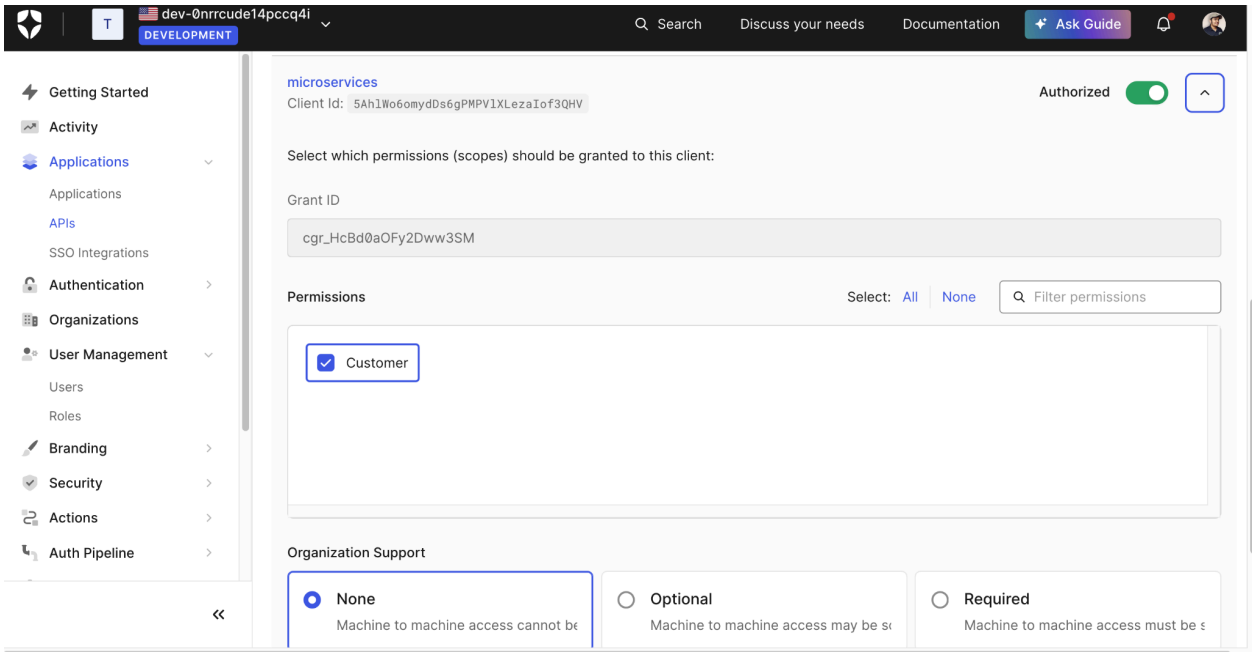




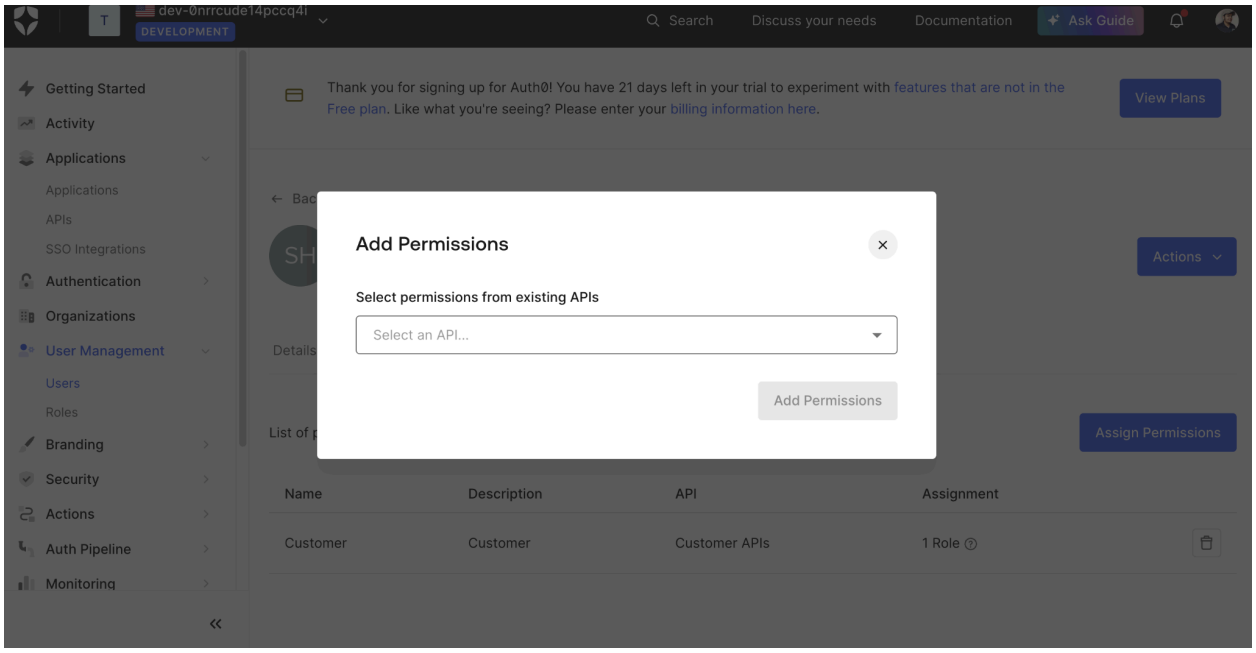


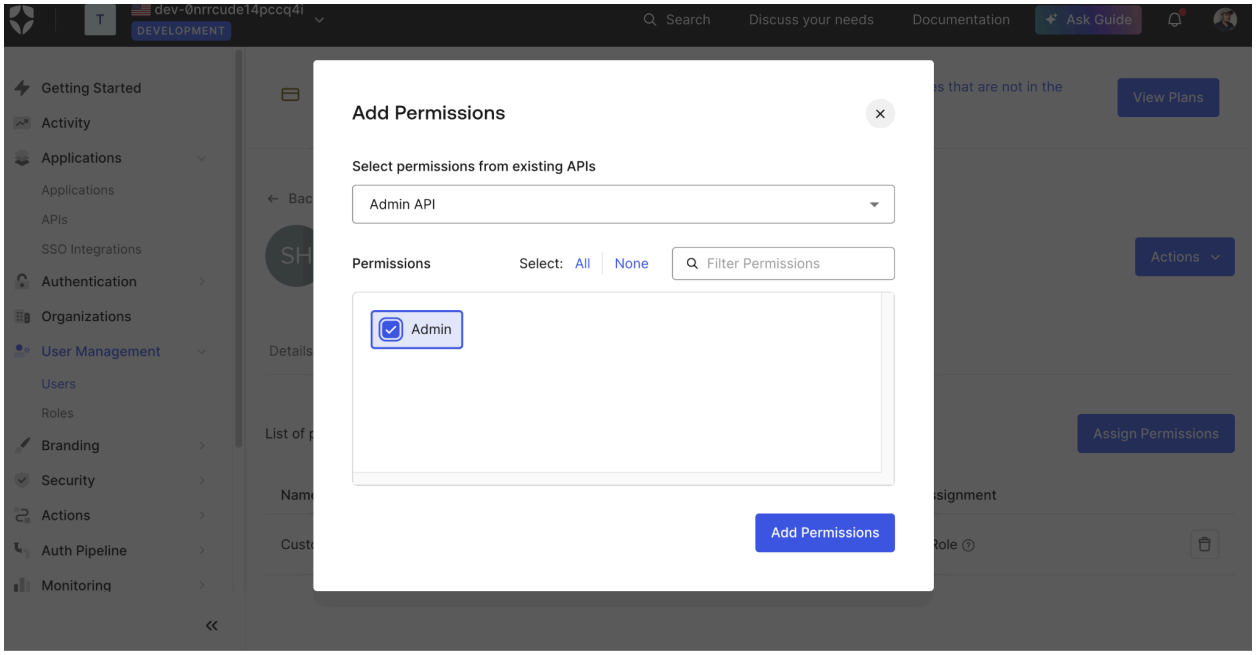
- Add Permission and Description for the API
- Multiple permissions can be added to the API, which can be used to add for the Roles.
- Go to Machine To Machine Application to add the Application we have created.





- Now, go to Users by User Management → Users to add permissions to the User
- Click on the User for which permission needs to be added.
- Go to Permission tab to add the permission
- Click on Assign Permission to assign permission.
- Select the API from which you need to add permission.
- Select all permissions, click on Save to save the permissions.

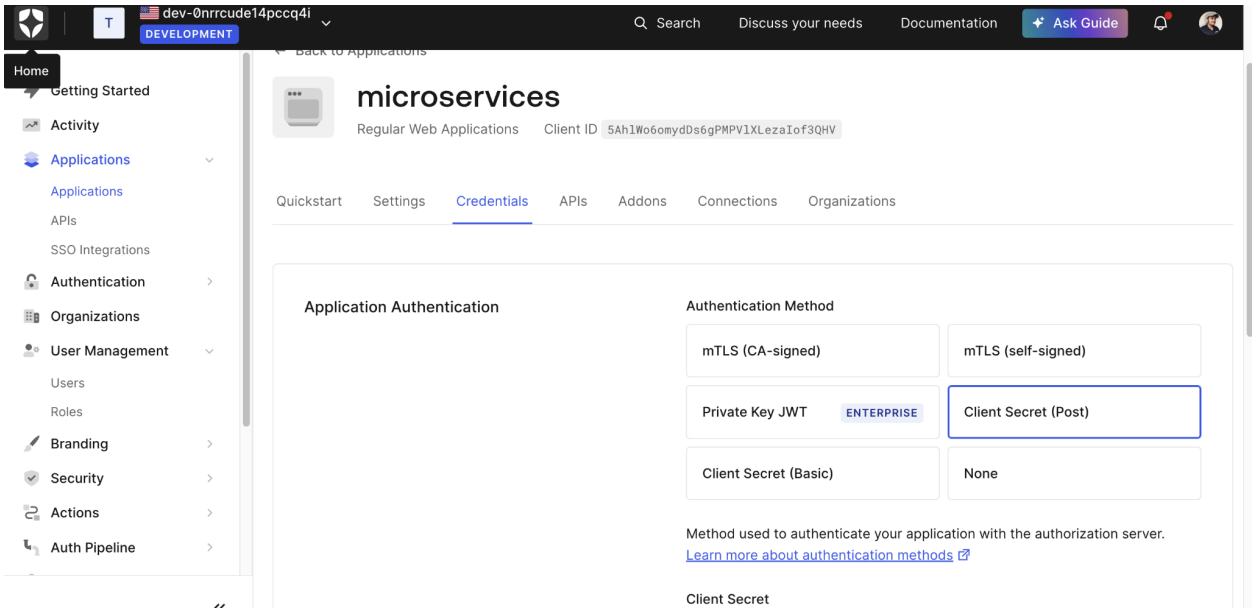




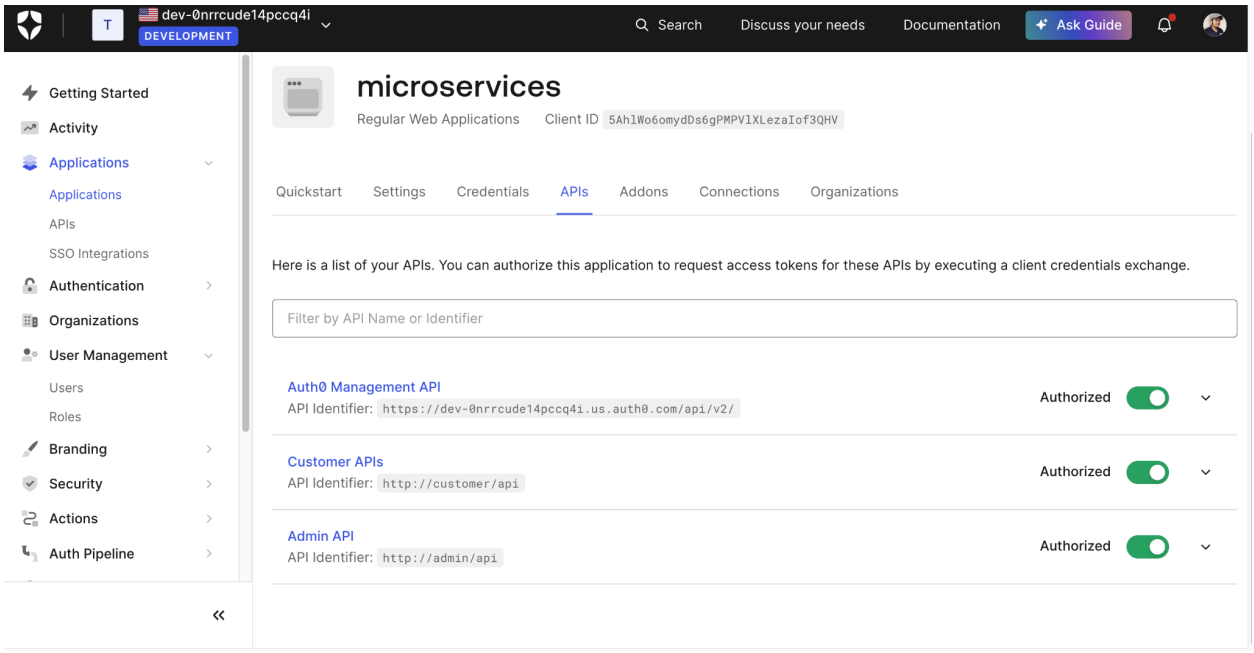
Add Customer Permissions. Based on the Users that you have created, add the permissions accordingly.

Application Configuration to add API’s and Application Authentication

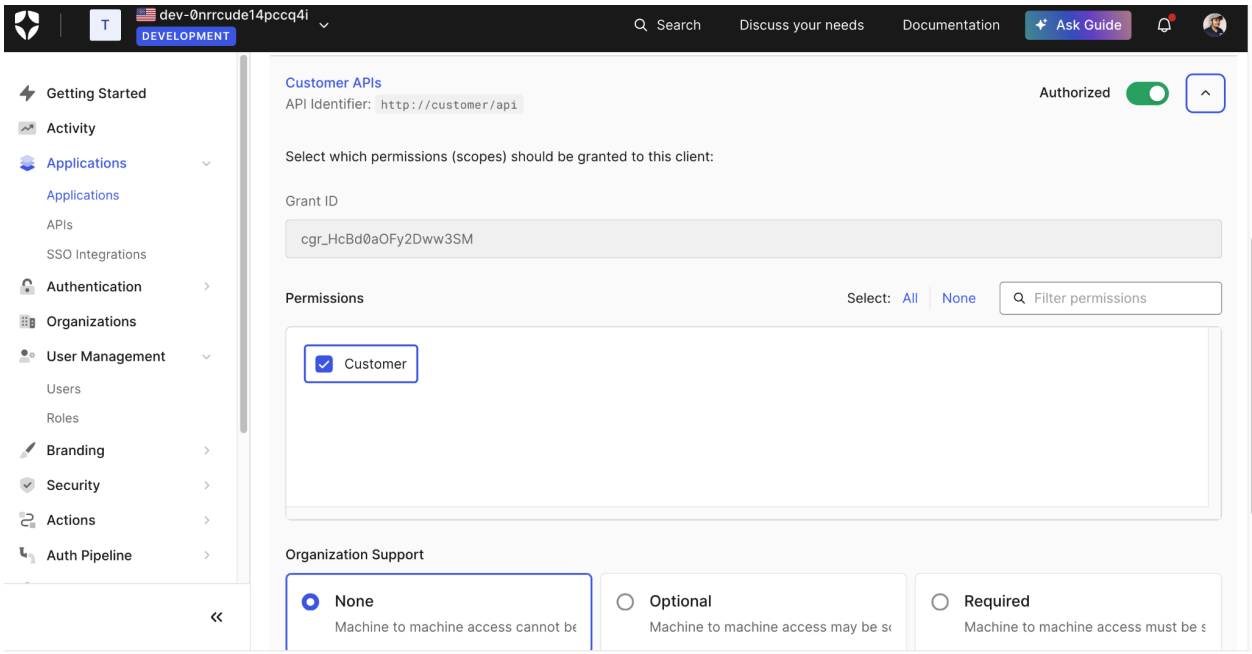
- Open Application and go to credentials tab to configure credentials.
- Set Application Authentication to “Client Secret (Post)”

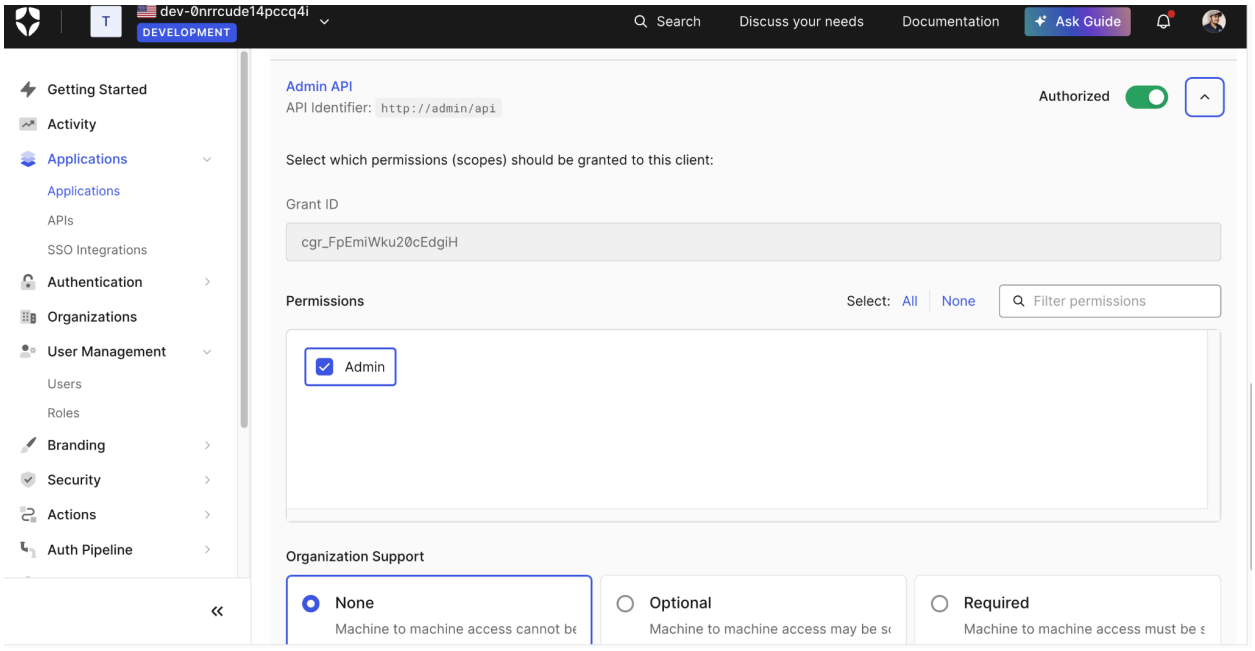


Go to the API tab and enable all API’s



For each API added, add the Permissions available from that API as below.

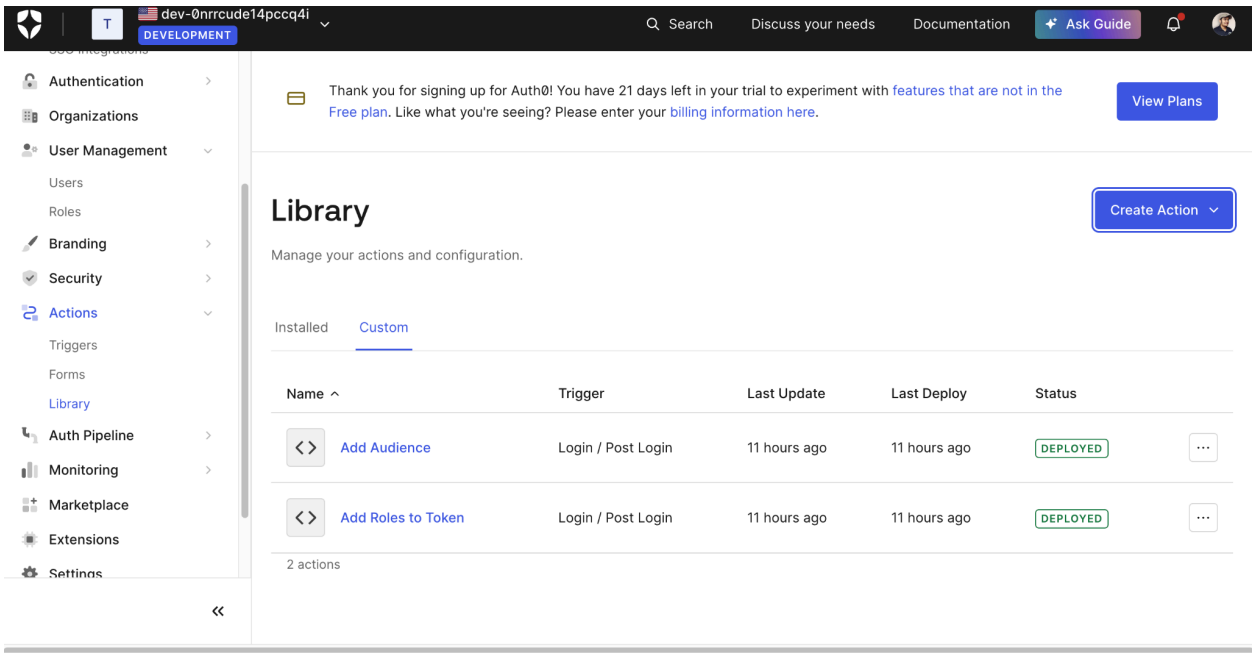


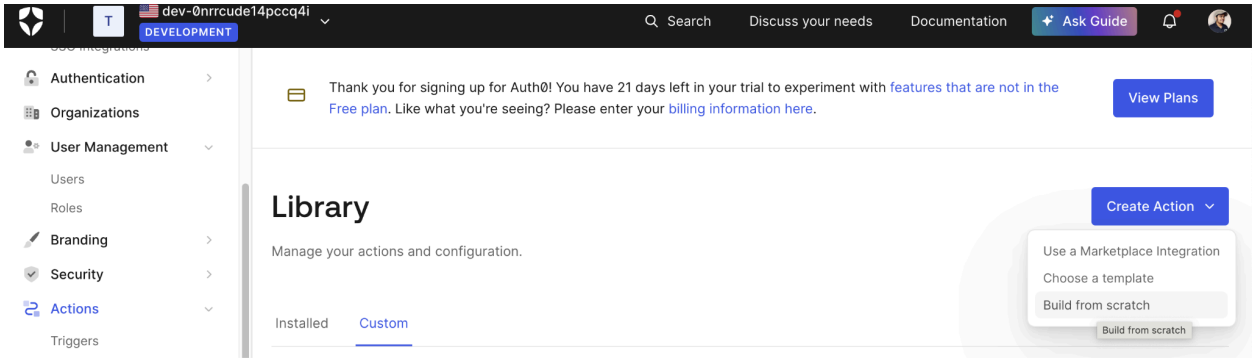


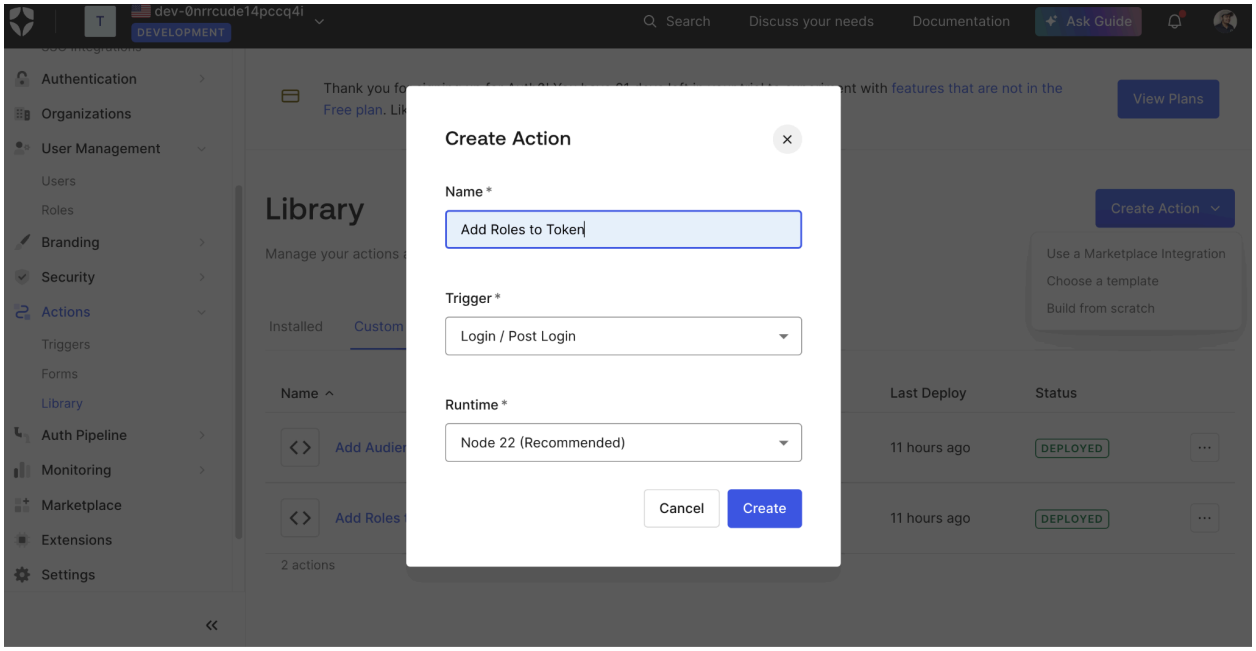
Now, your Application is configured to use Users, Roles, and Permissions.

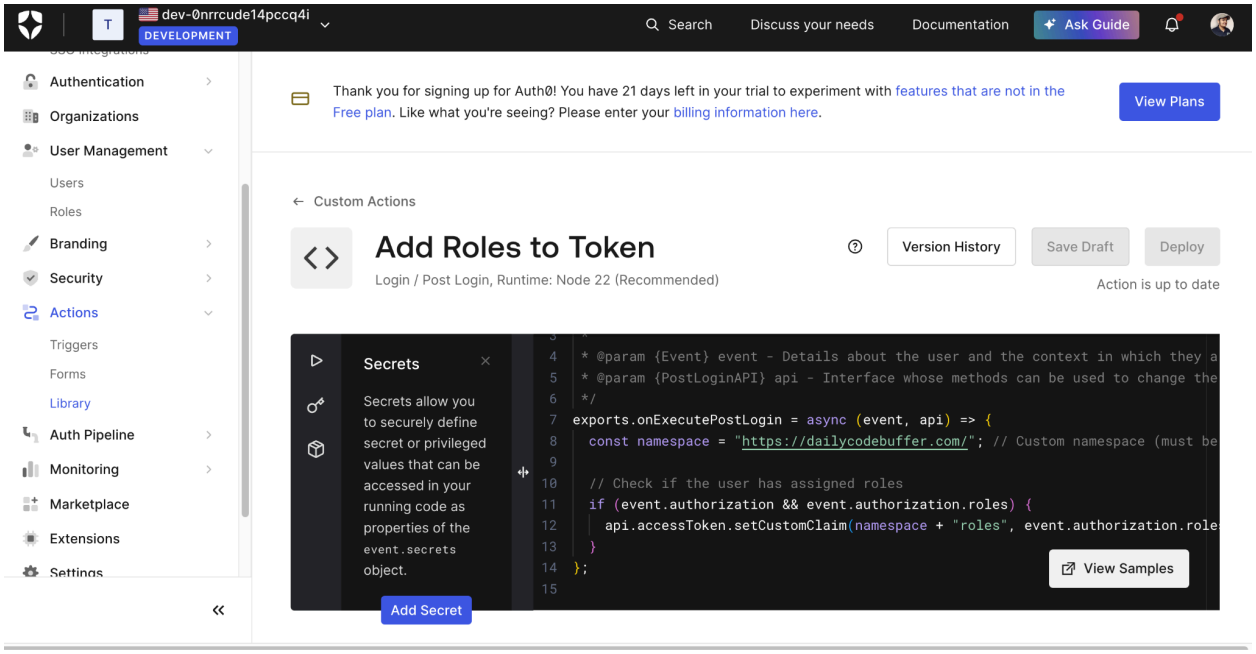
Add Roles and Audience in Access Token

- Auth0 by default does not add Roles and Audience information in the access token.
- When the access token is created, this information is needed to authorize the request.
- To add this information in the Access token, Triggers need to be set up in the Auth0 flow when the user is logged in.
- So, Create the Library which has code to add the details post Login.
- Add Trigger flow to invoke the Custom Library created.
- Go to Actions → Library to create Custom Library
- Go to Custom tab, click on Crate Action, and select Build from Scratch
- Give the appropriate name of the Library.
- Select “Login/post Login” trigger from drop-down.
- Select any Node to run the code.
- Click on Create to create the library.
- This will open the code editor to add the code.
- If the same is not opened, select the library created from the list and it will open the code editor.
- Add the code below and click on deploy to deploy the library.




dev-0nrrcude14pccq4i
DEVELOPMENT
Search
Discuss your needs
Documentation
Ask Guide
Authentication
Organizations
User Management
Users
Roles
Branding
Security
Actions
Triggers
Thank you for signing up for Auth0! You have 21 days left in your trial to experiment with features that are not in the Free plan. Like what you're seeing? Please enter your billing information here.
View Plans
Library
Manage your actions and configuration.
Create Action
Use a Marketplace Integration
Choose a template
Build from scratch
Installed
Custom


Create Action
Name *
Add Roles to Token
Trigger *
Login / Post Login
Runtime *
Node 22 (Recommended)
Cancel
Create
Library
Name
Last Deploy
Status
11 hours ago
DEPLOYED
2 actions



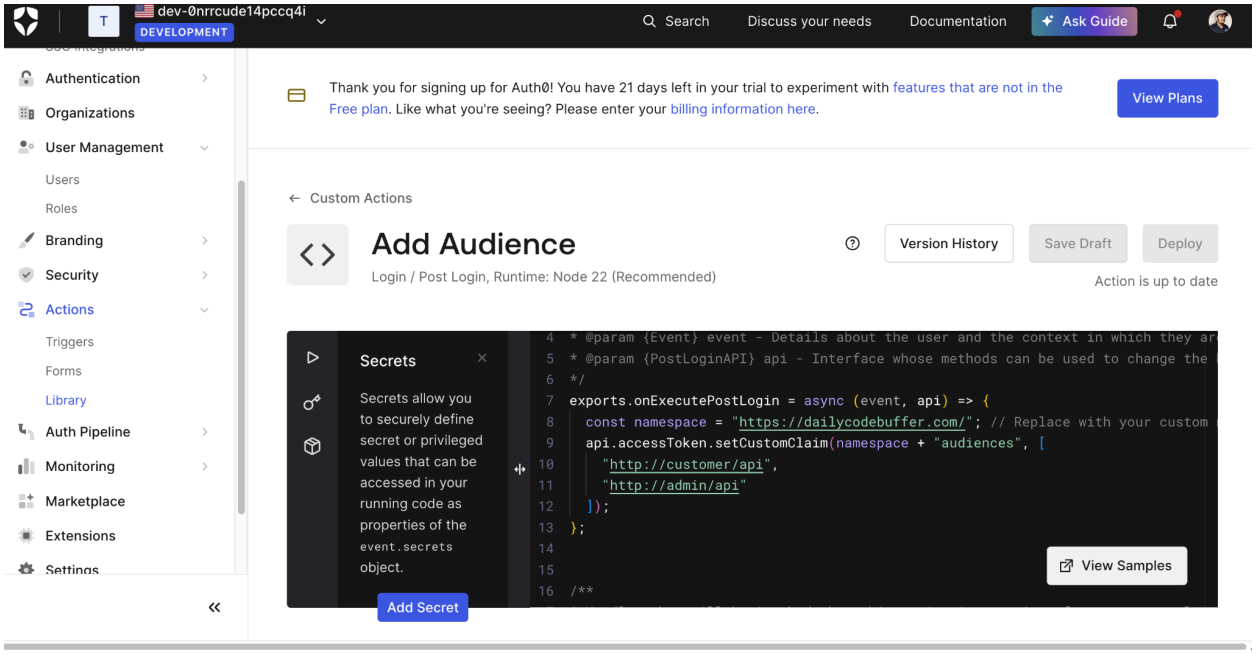

```
exports.onExecutePostLogin = async (event, api) => {
 const namespace = "https://dailycodebuffer.com/"; // Custom namespace
(must be unique)
  // Check if the user has assigned roles
 if (event.authorization && event.authorization.roles) {
   api.accessToken.setCustomClaim(namespace + "roles",
event.authorization.roles);
 }
};
```

Here, the namespace can be any unique namespace for the application. Here I have added my domain as the namespace. This may not be a valid domain or endpoint.

Click on Deploy to deploy the changes.

This Library will add the roles in the access token when the token is generated post login flow.

Create a similar Library for adding the Audience to the token as well as below.

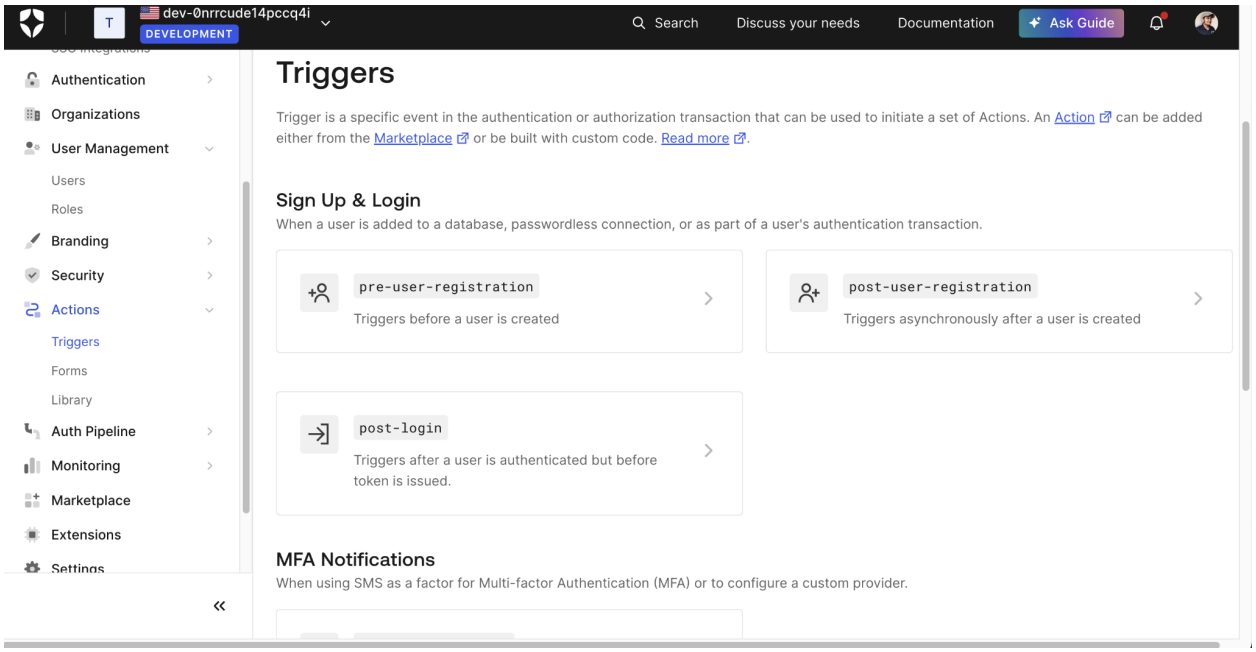


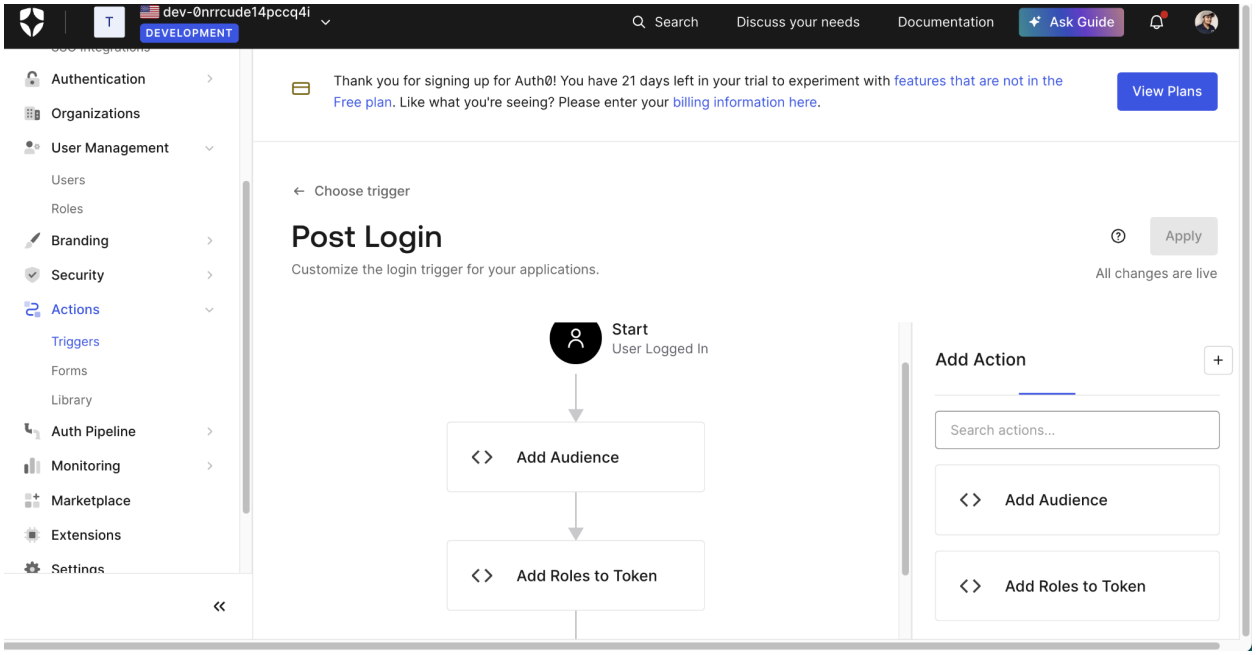

```
exports.onExecutePostLogin = async (event, api) => {
 const namespace = "https://dailycodebuffer.com/"; // Replace with your
custom namespace
 api.accessToken.setCustomClaim(namespace + "audiences", [
   "http://customer/api",
   "http://admin/api"
 ]);
};
```

Add the above code in the Library and click on deploy to deploy your library.

Add Triggers

- The two libraries created now to be added as part of the Post Login flow trigger.
- This trigger will be executed when the user logs in to the system and adds the above Roles and Audience information to the token generated.
- From the application, decode the JWT token and check for the Roles and Audience information, and authorize the user to allow the different APIs as per permission.
- Go to Actions → Triggers to add a trigger
- Click on post-login flow to add trigger
- Select both triggers and add to the flow and click on save.





- Go to Custom Action and add both actions in the flow.
- Drag and drop both the actions to the Post Login Flow
- Click on Apply to apply the settings