plotly

Star 23,447

Dash Python > ***Authentication***

> Plotly Studio: Transform any dataset into an interactive data application in minutes with AI. **Sign up for early access now.**

Authentication

This chapter covers two forms of authentication maintained by Plotly:

- `dash-enterprise-auth`, the authentication and authorization layer built-in to Plotly's commercial product, **Dash Enterprise**.
- `dash-auth`, a simple **basic auth** implementation.

Dash Enterprise Auth

Dash Enterprise provides an **authentication middleware** that is configured by your administrator. This authentication middleware connects to your organization's identity provider, allows your end users to log in with SSO, verifies if the user has permission to view the app, and then passes along user information like their username or group.

> Dash Enterprise can be installed on the cloud services of **AWS**, **Azure**, or **Google**.
>
> **Find out if your company is using Dash Enterprise**.

The `dash-enterprise-auth` package provides an API to access the username of the viewer of your Dash app. Use this username to implement conditional logic depending on who is logged in or use it in your API or database calls (row level security).

Dash Enterprise automatically implements app authorization if your **Dash app's viewer access level** is set to *Restricted* or *Authenticated*, but not if it is set to *Unauthenticated*.

Using `dash-enterprise-auth` in an Existing Dash App

If you have previously deployed your Dash app to Dash Enterprise, add `dash-enterprise-libraries` (Dash Enterprise >= 5.2) or `dash-enterprise-auth` (Dash Enterprise < 5.2) to your `requirements.txt` file to get started.

`dash-enterprise-auth` allows you to get information about your app viewer with the `get_username` and `get_user_data` methods, as well as add a logout button with the `create_logout_button` method.

For more information about developing your app with `dash-enterprise-auth`, see **Dash Enterprise Auth**.

Dash Enterprise Auth Example

```
from dash import Dash, dcc, html, Input, Output, callback
import dash_enterprise_auth as auth

external_stylesheets = ['https://codepen.io/chriddyp/pen/bWLwgP.css']

app = Dash(__name__, external_stylesheets=external_stylesheets)

server = app.server  # Expose the server variable for deployments

# Standard Dash app code below
app.layout = html.Div(className='container', children=[
```

```
    html.Div([
        html.H2('Sample App', id='header-title', className='ten columns'),
        html.Div(auth.create_logout_button(), className='two columns', style={'marginTop': 30}
    ]),
    html.Div(id='dummy-input', style={'display': 'none'}),

    html.Div([
        html.Div(
            className='four columns',
            children=[
                dcc.Dropdown(['LA', 'NYC', 'MTL'], 'LA', id='dropdown')
        ]),
        html.Div(
            className='eight columns',
            children=[
                dcc.Graph(id='graph')
            ])
    ])
])

@callback(Output('header-title','children'), Input('dummy-input', 'children'))
def update_title(_):

    # print user data to the logs
    print(auth.get_user_data())

    # update header with username
    return 'Hello {}'.format(auth.get_username())

@callback(Output('graph', 'figure'),
              Input('dropdown', 'value'))
def update_graph(value):
    return {
        'data': [{
            'x': [1, 2, 3, 4, 5, 6],
            'y': [3, 1, 2, 3, 5, 6]
        }],
        'layout': {
            'title': value,
            'margin': {
```

Basic Auth

The `dash-auth` package provides a **HTTP Basic Auth**.

As a Dash developer, you hardcode a set of usernames and passwords in your code and send those usernames and passwords to your viewers. There are a few limitations to HTTP Basic Auth:

- Users can not log out of applications
- You are responsible for sending the usernames and passwords to your viewers over a secure channel
- Your viewers can not create their own account and cannot change their password
- You are responsible for safely storing the username and password pairs in your code.

Basic Auth Example

Installation:

```
pip install dash==3.1.1
pip install dash-auth==2.0.0
```

Example Code:

```
from dash import Dash, dcc, html, Input, Output, callback
import dash_auth

# Keep this out of source code repository - save in a file or a database
VALID_USERNAME_PASSWORD_PAIRS = {
```

```
    'hello': 'world'
}

external_stylesheets = ['https://codepen.io/chriddyp/pen/bWLwgP.css']

app = Dash(__name__, external_stylesheets=external_stylesheets)
auth = dash_auth.BasicAuth(
    app,
    VALID_USERNAME_PASSWORD_PAIRS
)

app.layout = html.Div([
    html.H1('Welcome to the app'),
    html.H3('You are successfully authorized'),
    dcc.Dropdown(['A', 'B'], 'A', id='dropdown'),
    dcc.Graph(id='graph')
], className='container')

@callback(
    Output('graph', 'figure'),
    Input('dropdown', 'value'))
def update_graph(dropdown_value):
    return {
        'layout': {
            'title': 'Graph of {}'.format(dropdown_value),
            'margin': {
                'l': 20,
                'b': 20,
                'r': 10,
                't': 60
            }
        },
        'data': [{'x': [1, 2, 3], 'y': [4, 1, 2]}]
    }

if __name__ == '__main__':
    app.run(debug=True)
```

Dash Python ***> Authentication***