

1. Building your own component in Dash

MacOS - brew install node OR <https://phoenixnap.com/kb/install-npm-mac>

Windows: <https://phoenixnap.com/kb/install-node-js-npm-on-windows>

Linux - RHEL/CentOS/Ubuntu: <https://www.tecmint.com/install-nodejs-npm-in-centos-ubuntu/>

2. Check for node and npm version

node --version OR nodejs --version

npm --version

3. Install python libraries

pip install cookiecutter

pip install virtualenv

4. Create a React project using the template here:

cookiecutter <https://github.com/plotly/dash-component-boilerplate.git>

```
You've downloaded /Users/kritikaversha/.cookiecutters/dash-component-boilerplate be
[yes]: yes
project_name [my dash component]: example_component
project_shortname [example_component]: customgrid
component_name [Customgrid]: Customgrid
jl_prefix []:
r_prefix []:
author_name [Enter your first and last name (For package.json)]: kritika versha
author_email [Enter your email (For package.json)]: kmversha@umich.edu
github_org []: https://github.com/KritikaVersha
description [Project Description]: custom react component with dash
Select use_async:
1 5-100
```

5. If using virtualenv, install the dependencies:

npm install

virtualenv venv

venv\Scripts\activate

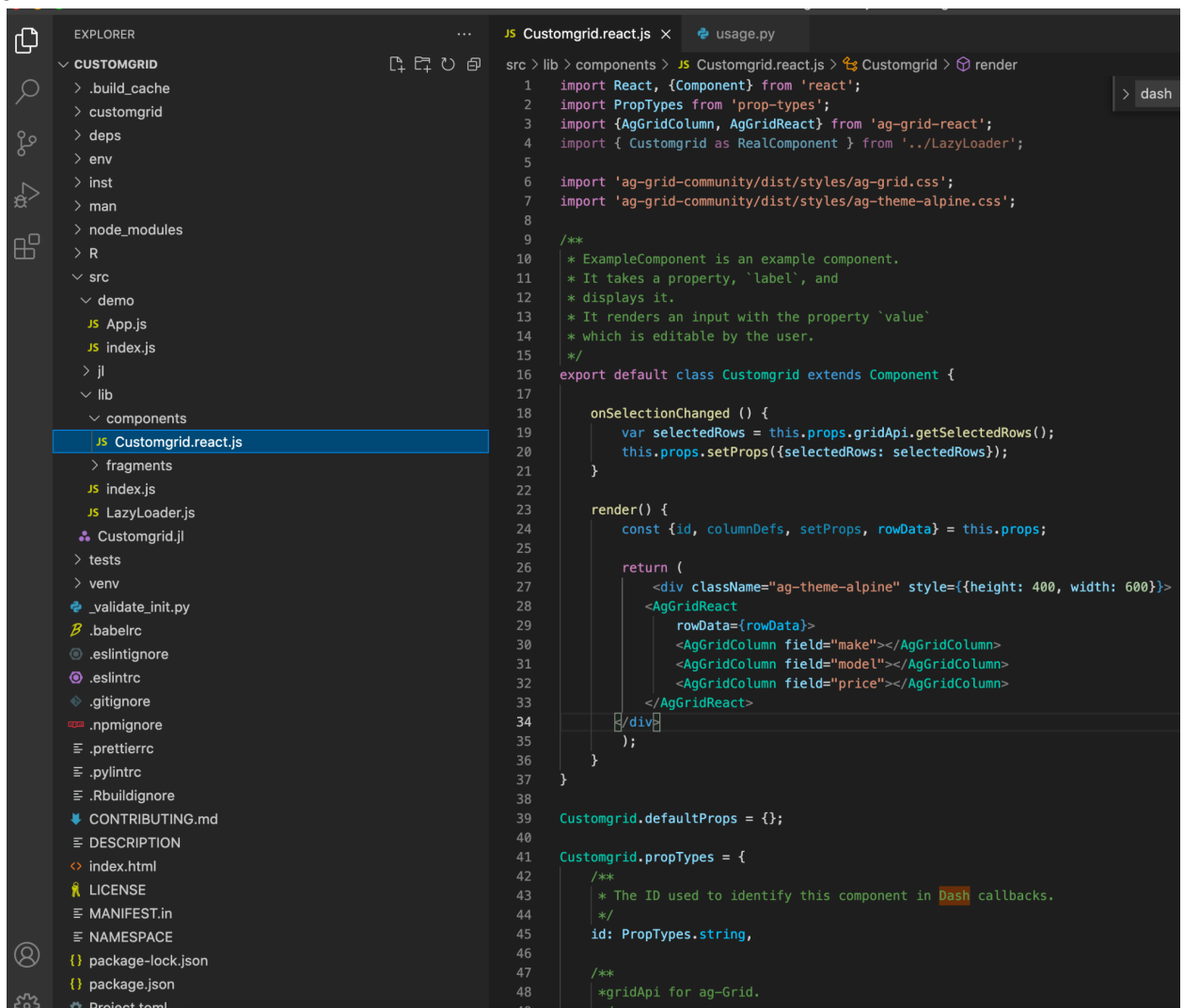
pip install -r requirements.txt

pip install -r tests/requirements.txt

6. Add the ag-Grid NPM package to your folder:

npm install --save ag-grid-community ag-grid-react ag-grid

7. Open your preferred IDE and check the project/workspace to see the component folder. You should be able to see a react.js file. Open the file and update it with your component code



```
src > lib > components > JS Customgrid.react.js > Customgrid > render
1  import React, {Component} from 'react';
2  import PropTypes from 'prop-types';
3  import {AgGridColumn, AgGridReact} from 'ag-grid-react';
4  import { Customgrid as RealComponent } from '../LazyLoader';
5
6  import 'ag-grid-community/dist/styles/ag-grid.css';
7  import 'ag-grid-community/dist/styles/ag-theme-alpine.css';
8
9  /**
10   * ExampleComponent is an example component.
11   * It takes a property, 'label', and
12   * displays it.
13   * It renders an input with the property 'value'
14   * which is editable by the user.
15   */
16  export default class Customgrid extends Component {
17
18      onSelectionChanged () {
19          var selectedRows = this.props.gridApi.getSelectedRows();
20          this.props.setProps({selectedRows: selectedRows});
21      }
22
23      render() {
24          const {id, columnDefs, setProps, rowData} = this.props;
25
26          return (
27              <div className="ag-theme-alpine" style={{height: 400, width: 600}}>
28                  <AgGridReact
29                      rowData={rowData}>
30                      <AgGridColumn field="make"></AgGridColumn>
31                      <AgGridColumn field="model"></AgGridColumn>
32                      <AgGridColumn field="price"></AgGridColumn>
33                  </AgGridReact>
34              </div>
35          );
36      }
37  }
38
39  Customgrid.defaultProps = {};
40
41  Customgrid.propTypes = {
42      /**
43       * The ID used to identify this component in Dash callbacks.
44       */
45      id: PropTypes.string,
46
47      /**
48       * gridApi for ag-Grid.
49       */
50  }
```

8. Copy the attached code into this file to add ag-grid table component

```
import React, {Component} from 'react';
import PropTypes from 'prop-types';
import {AgGridColumn, AgGridReact} from 'ag-grid-react';
import { Customgrid as RealComponent } from '../LazyLoader';

import 'ag-grid-community/dist/styles/ag-grid.css';
import 'ag-grid-community/dist/styles/ag-theme-alpine.css';

/**
 * ExampleComponent is an example component.
 * It takes a property, `label`, and
 * displays it.
 * It renders an input with the property `value`
 * which is editable by the user.
 */
export default class Customgrid extends Component {

  onSelectionChanged () {
    var selectedRows = this.props.gridApi.getSelectedRows();
    this.props.setProps({selectedRows: selectedRows});
  }

  render() {
    const {id, columnDefs, setProps, rowData} = this.props;

    return (
      <div className="ag-theme-alpine" style={{height: 400, width: 600}}>
        <AgGridReact
          rowData={rowData}>
          <AgGridColumn field="make"></AgGridColumn>
          <AgGridColumn field="model"></AgGridColumn>
          <AgGridColumn field="price"></AgGridColumn>
        </AgGridReact>
      </div>
    );
  }
}

Customgrid.defaultProps = {};

Customgrid.propTypes = {
```

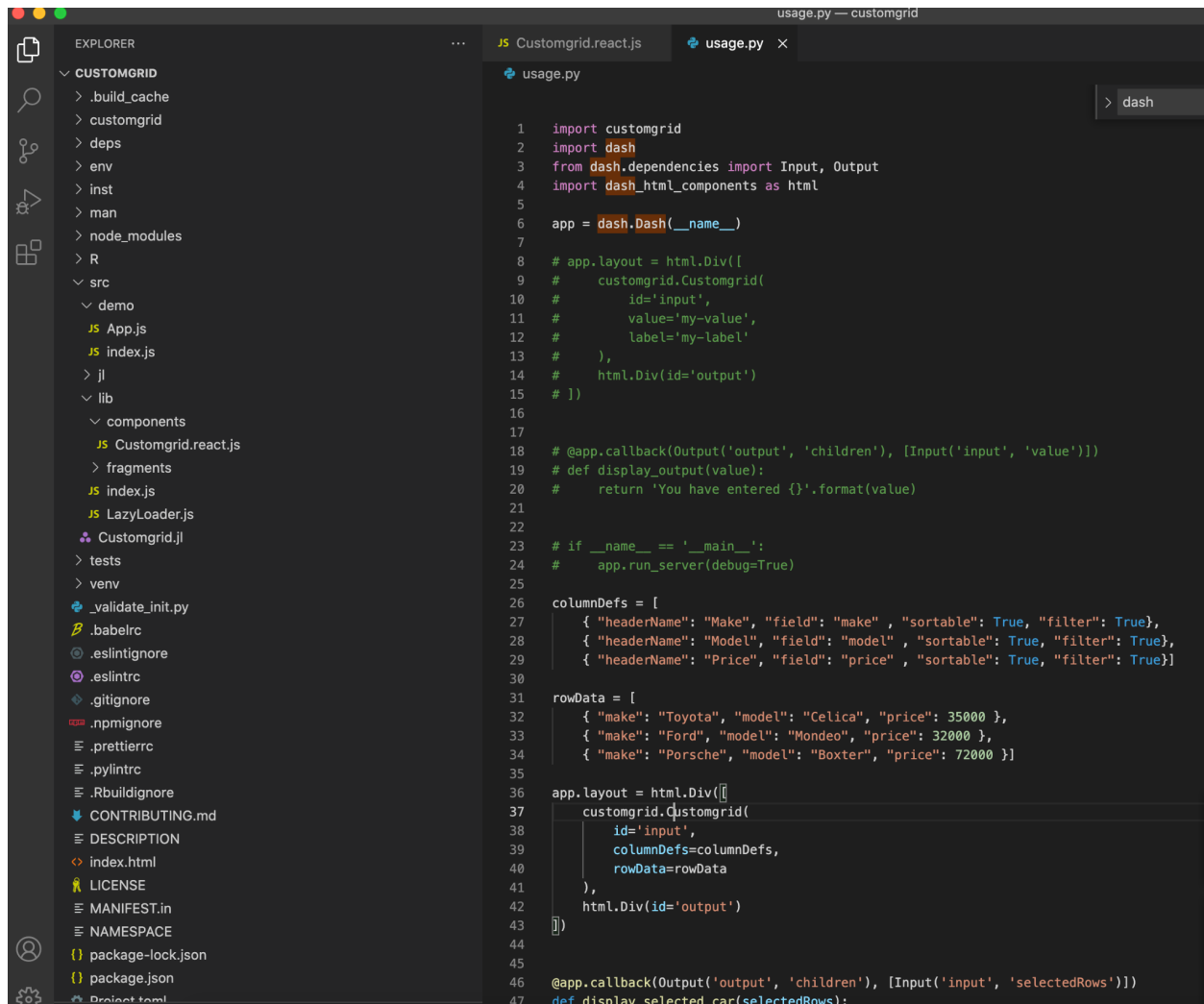
```

/**
 * The ID used to identify this component in Dash callbacks.
 */
id: PropTypes.string,

/**
 *gridApi for ag-Grid.
 */
gridApi: PropTypes.object,
/**
 *columnDefs for ag-Grid.
 */
columnDefs: PropTypes.array,
/**
 * rowData for ag-Grid.
 */
rowData: PropTypes.array,
/**
 * selectedRows for ag-Grid.
 */
selectedRows: PropTypes.array,
/**
 * Dash-assigned callback that should be called to report property changes
 * to Dash, to make them available for callbacks.
 */
setProps: PropTypes.func
};

// export const defaultProps = Customgrid.defaultProps;
// export const propTypes = Customgrid.propTypes;

```



9. Build the project

npm run build:js

npm run build:backends

npm run build

10. Open the `usage.py` file in the folder. Update this file to add the `react.js` component and check if your dash application is running and you are able to see the ag-Grid table

```
import React, {Component} from 'react';
import PropTypes from 'prop-types';
import {AgGridColumn, AgGridReact} from 'ag-grid-react';
import { Customgrid as RealComponent } from '../LazyLoader';

import 'ag-grid-community/dist/styles/ag-grid.css';
import 'ag-grid-community/dist/styles/ag-theme-alpine.css';

/**
 * ExampleComponent is an example component.
 * It takes a property, `label`, and
 * displays it.
 * It renders an input with the property `value`
 * which is editable by the user.
 */
export default class Customgrid extends Component {

  onSelectionChanged () {
    var selectedRows = this.props.gridApi.getSelectedRows();
    this.props.setProps({selectedRows: selectedRows});
  }

  render() {
    const {id, columnDefs, setProps, rowData} = this.props;

    return (
      <div className="ag-theme-alpine" style={{height: 400, width: 600}}>
        <AgGridReact
          rowData={rowData}>
          <AgGridColumn field="make"></AgGridColumn>
          <AgGridColumn field="model"></AgGridColumn>
          <AgGridColumn field="price"></AgGridColumn>
        </AgGridReact>
      </div>
    );
  }
}
```

```

Customgrid.defaultProps = {};

Customgrid.propTypes = {
  /**
   * The ID used to identify this component in Dash callbacks.
   */
  id: PropTypes.string,

  /**
   *gridApi for ag-Grid.
   */
  gridApi: PropTypes.object,
  /**
   *columnDefs for ag-Grid.
   */
  columnDefs: PropTypes.array,
  /**
   * rowData for ag-Grid.
   */
  rowData: PropTypes.array,
  /**
   * selectedRows for ag-Grid.
   */
  selectedRows: PropTypes.array,
  /**
   * Dash-assigned callback that should be called to report property changes
   * to Dash, to make them available for callbacks.
   */
  setProps: PropTypes.func
};

// export const defaultProps = Customgrid.defaultProps;
// export const propTypes = Customgrid.propTypes;

```

11. Run the command `python usage.py` file

←

→

↺

🏠

📄 127.0.0.1:8050

Make	Model	Price
Toyota	Celica	35000
Ford	Mondeo	32000
Porsche	Boxter	72000

12. Export the component as python package (Optional required for today's training)

```
python setup.py sdist
```

```
pip install customgrid-0.0.1.tar.gz
```