const path = require('path');

const webpack = require('webpack');

const ExtractTextPlugin = require('extract-text-webpack-plugin');

module.exports = (env) => {

const extractCSS = new ExtractTextPlugin('vendor.css');

const isDevBuild = !(env && env.prod);

return [{

stats: { modules: false },

resolve: {

extensions: [ '.js' ]

},

module: {

rules: [

{ test: /\.(png|woff|woff2|eot|ttf|svg)(\?|$)/, use: 'url-loader?limit=100000' },

{ test: /\.css(\?|$)/, use: extractCSS.extract([ isDevBuild ? 'css-loader' : 'css-loader?minimize' ]) }

]

},

entry: {

vendor: ['bootstrap', 'bootstrap/dist/css/bootstrap.css', 'event-source-polyfill', 'isomorphic-fetch', 'react', 'react-dom', 'react-router-dom', 'jquery'],

},

output: {

path: path.join(\_\_dirname, 'wwwroot', 'dist'),

publicPath: 'dist/',

filename: '[name].js',

library: '[name]\_[hash]',

},

plugins: [

extractCSS,

new webpack.ProvidePlugin({ $: 'jquery', jQuery: 'jquery' }), // Maps these identifiers to the jQuery package (because Bootstrap expects it to be a global variable)

new webpack.DllPlugin({

path: path.join(\_\_dirname, 'wwwroot', 'dist', '[name]-manifest.json'),

name: '[name]\_[hash]'

}),

new webpack.DefinePlugin({

'process.env.NODE\_ENV': isDevBuild ? '"development"' : '"production"'

})

].concat(isDevBuild ? [] : [

new webpack.optimize.UglifyJsPlugin()

])

}];

};