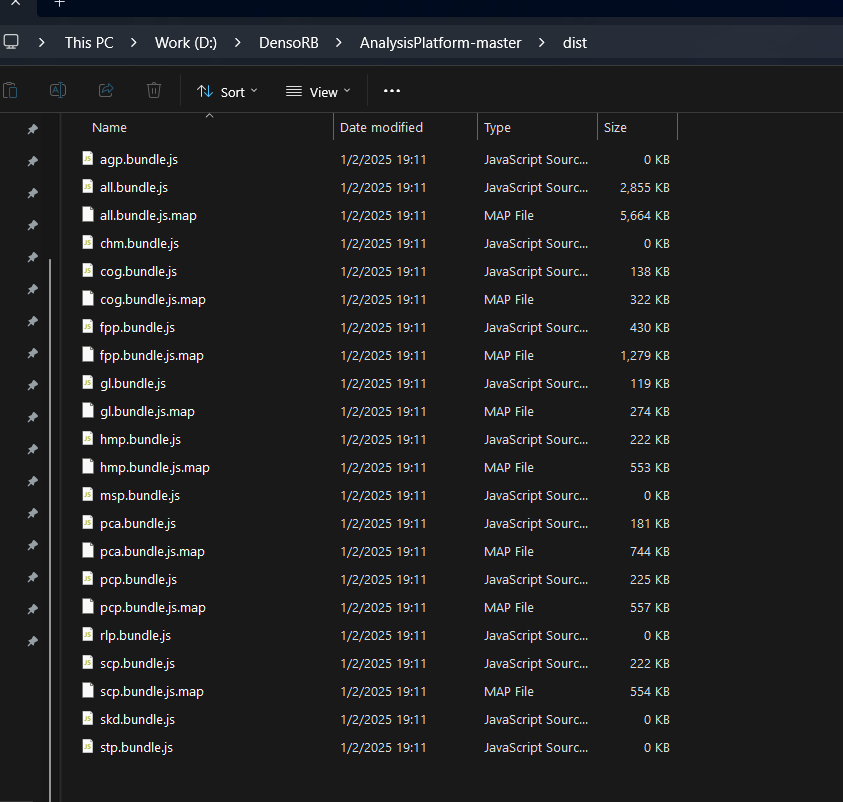
Report webpack



Webpack config file:

const path = require("path");

const fs = require("fs");

const webpack = require("webpack");

const MiniCssExtractPlugin = require("mini-css-extract-plugin");

const TerserPlugin = require("terser-webpack-plugin");

// Đọc file JSON để lấy thông tin assets

const assets = JSON.parse(

  fs.readFileSync("./ap/common/assets/assets.json", "utf-8")

);

// Base path để dẫn đến thư mục chứa các file JS và CSS

const BASE\_PATH = path.resolve(\_\_dirname, "ap/static");

// Hàm tạo các entry từ file assets.json

const createEntries = (assets) => {

  const entries = {};

  Object.keys(assets).forEach((page) => {

    if (assets[page].js) {

      entries[page] = assets[page].js.map((file) => path.join(BASE\_PATH, file));

    }

  });

  return entries;

};

module.exports = {

  mode: "development", // Chế độ phát triển, dễ dàng debug

  // Sử dụng hàm createEntries để lấy entry từ assets.json

  entry: createEntries(assets),

  // Đường dẫn và tên file output sau khi biên dịch

  output: {

    path: path.resolve(\_\_dirname, "dist"), // Thư mục chứa các file xuất ra

    filename: "[name].bundle.js", // Tên file JS dựa trên entry

  },

  // Cấu hình resolver để xử lý các module và alias

  resolve: {

    alias: {

      jquery: path.resolve(\_\_dirname, "ap/static/common/js/jquery.js"),

      moment: path.resolve(\_\_dirname, "ap/static/common/js/moment.js"),

      "@": path.resolve(\_\_dirname, "ap/static/common/js"),

    },

  },

  module: {

    rules: [

      // Xử lý file JavaScript với Babel

      {

        test: /\.js$/,

        exclude: /node\_modules/,

        use: {

          loader: "babel-loader",

          options: {

            presets: ["@babel/preset-env"], // Hỗ trợ ES6+

          },

        },

      },

      // Xử lý các file CSS (nếu có)

      {

        test: /\.css$/,

        use: [

          MiniCssExtractPlugin.loader, // Tách CSS ra file riêng biệt

          "css-loader", // Xử lý CSS

        ],

      },

    ],

  },

  plugins: [

    // Cung cấp jQuery toàn cục mà không cần phải import mỗi lần

    new webpack.ProvidePlugin({

      $: "jquery",

      jQuery: "jquery",

      "window.jQuery": "jquery",

    }),

    // Tách các tệp CSS ra ngoài và sử dụng hash cho tên file

    new MiniCssExtractPlugin({

      filename: "styles.[contenthash].css", // Tên file CSS với hash

    }),

  ],

  optimization: {

    minimize: true, // Bật tính năng giảm dung lượng

    minimizer: [

      new TerserPlugin({

        terserOptions: {

          format: {

            comments: false, // Xóa comment trong file

          },

          compress: {

            drop\_console: true, // Loại bỏ console.log

          },

        },

        extractComments: false, // Không xuất file comment riêng

      }),

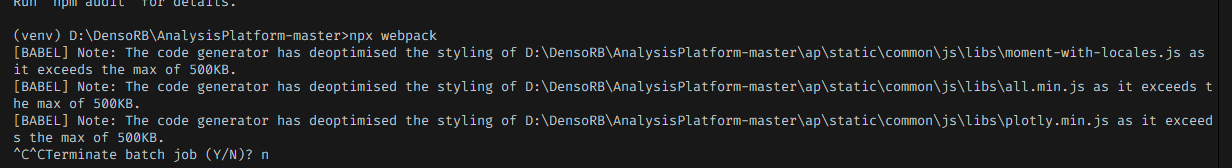
    ],

  },

  // Tạo source map cho dễ dàng debug trong môi trường phát triển

  devtool: "source-map",

};



Các các để có thể vừa chạy được flask vừa chạy được webpack:

1. Concurrently
2. Npm-run-all

"scripts": { "serve:flask": "python app.py", "serve:webpack": "npx webpack --watch", "start": "npm-run-all --parallel serve:\*" }

1. Honcho (python):

Create procfile:

flask: python app.py webpack: npx webpack --watch

Honcho start

