fetchData.js

```
// Define the base API URLĐconst apiBaseURL = 'http://localhost:5000/api/
getData';Đ/ Function to fetch data from a specific tableĐfunction
fetchTableData(tableName) {Đ   const outputElement =
document.getElementById('display-content');Đ   // Show a loading message
while fetching dataĐ   outputElement.innerHTML = 'Fetching data, please
wait...';Đ   // Make the API callĐ   fetch(`${apiBaseURL}?
tableName=${tableName}`)Đ       .then((response) => {Đ           if (!
response.ok) {Đ               throw new Error(`HTTP error! Status:
${response.status}`);Đ           }Đ           return response.json();Đ
        })Đ       .then((data) => {Đ           if (data.success && data.data)
{Đ               // Create a table dynamicallyĐ               const table =
document.createElement('table');Đ               table.border = 1;Đ
                // Create table headerĐ               const headerRow =
document.createElement('tr');Đ
Object.keys(data.data[0]).forEach((key) => {Đ                   const
headerCell = document.createElement('th');Đ
headerCell.textContent = key;Đ
headerRow.appendChild(headerCell);Đ               });Đ
table.appendChild(headerRow);Đ               // Populate table rowsĐ
                data.data.forEach((row) => {Đ                   const dataRow
= document.createElement('tr');Đ
Object.values(row).forEach((value) => {Đ                       const cell =
document.createElement('td');Đ                       cell.textContent = value;Đ
                        dataRow.appendChild(cell);Đ                   });Đ
                    table.appendChild(dataRow);Đ               });Đ
                // Replace the output content with the tableĐ
outputElement.innerHTML = '';Đ               outputElement.appendChild(table);Đ
            } else {Đ               outputElement.innerHTML = 'No data found';Đ
            }Đ       })Đ       .catch((error) => {Đ
console.error('Error fetching data:', error);Đ
outputElement.innerHTML = `Error fetching data: ${error.message}`;Đ       });Đ}Đ
```

login.js

```
export default function handler(req, res) {Đ   if (req.method === 'POST') {Đ
      const { username, password } = req.body;Đ Đ     // Example logicĐ
if (username === 'admin' && password === 'password') {Đ
res.status(200).json({ message: 'Login successful' });Đ     } else {Đ
res.status(401).json({ message: 'Invalid credentials' });Đ     }Đ   } else {Đ
      res.status(405).json({ message: 'Method Not Allowed' });Đ   }Đ }Đ
```

server.js

```
const express = require('express');Đconst bodyParser = require('body-parser');Đ
const mssql = require('mssql');Đconst cors = require('cors'); // Import CORSĐ
const app = express();Đconst port = 5000;Đ// Middleware for parsing JSON and
handling CORSĐapp.use(cors());  // Allow cross-origin requests from any originĐ
app.use(bodyParser.json());Đ// SQL Server ConfigurationĐconst dbConfig = {Đ
user: 'sa', // Replace with your database usernameĐ   password:
'@Durga123', // Replace with your database passwordĐ   server:
'BV_DURGA_RAO', // Replace with your server address if differentĐ   database:
'UserLoginDB', // Replace with your database nameĐ   options: {Đ
encrypt: true, // For secure connectionsĐ       trustServerCertificate:
true, // For self-signed certificatesĐ   },Đ};Đ// Login APIĐapp.post('/login',
async (req, res) => {Đ   const { username, password } = req.body;Đ   if (!
username || !password) {Đ       return res.status(400).json({ message:
'Username and password are required.' });Đ   }Đ   try {Đ       // Connect to
the SQL ServerĐ       const pool = await mssql.connect(dbConfig);Đ       //
Query the databaseĐ       const result = await poolĐ           .request()Đ
            .input('Username', mssql.VarChar, username)Đ
            .input('PasswordHash', mssql.VarChar, password)Đ
            .query(`Đ               SELECT * FROM Users Đ               WHERE
Username = @Username Đ               AND PasswordHash = @PasswordHashĐ
            `);Đ       // Check login resultĐ       if
(result.recordset.length > 0) {Đ           res.status(200).json({ success:
true, message: 'Login successful.' });Đ       } else {Đ
res.status(401).json({ success: false, message: 'Invalid username or
password.' });Đ       }Đ   } catch (error) {Đ       console.error('Database
error:', error);Đ       res.status(500).json({ success: false, message:
'Internal server error.' });Đ   }Đ});Đapp.get('/api/getData', async (req, res)
=> {Đ   const { tableName } = req.query;Đ   console.log('Request received for
table:', tableName); // Log the table nameĐ   if (!tableName) {Đ       return
res.status(400).json({ success: false, message: 'Table name is required.' });Đ
    }Đ   try {Đ       const pool = await mssql.connect(dbConfig);Đ
console.log('Database connected');Đ       // Get the current time and
calculate the date range for the past 24 hoursĐ       const now = new Date();Đ
        const fromDate = new Date(now);Đ
fromDate.setHours(now.getHours() - 24); // 24 hours ago, same time as currentĐ
        // Format dates to SQL-friendly format (YYYY-MM-DD HH:mm:ss)Đ
const formattedFromDate = fromDate.toISOString().replace('T', ' ').slice(0,
19); // YYYY-MM-DD HH:mm:ssĐ       const formattedToDate =
now.toISOString().replace('T', ' ').slice(0, 19); // YYYY-MM-DD HH:mm:ssĐ
        // Define queries for specific tablesĐ       let query;Đ       switch
(tableName) {Đ           case 'ALarmHistory1':Đ               query = `Đ
                    SELECT Đ                       [SNo], Đ
                        [AlarmDescription], Đ
FORMAT([AlarmActivatedDateTime], 'dd MM yyyy HH:mm:ss') AS
AlarmActivatedDateTime, Đ
FORMAT([AlarmDeActivatedDateTime], 'dd MM yyyy HH:mm:ss') AS
AlarmDeActivatedDateTime,Đ                      [AlarmDuration], Đ
                        [Comment], Đ                       [AlarmCommentBy], Đ
                        [Groupname]Đ                     FROM [dbo].
[ALarmHistory1]Đ                 ORDER BY [AlarmActivatedDateTime] DESCĐ
                `;Đ               break;Đ           case
'loginActivity1':       Đ           query = `Đ                   SELECT Đ
                        [SNo],Đ                    [Username],Đ
                        FORMAT([DateTime], 'dd MM yyyy HH:mm:ss') AS DateTimeĐ
                    FROM [dbo].[loginActivity1]Đ                 ORDER BY
[DateTime] DESCĐ               `;Đ               break;Đ           case
'DataOfSensors':Đ               // Make sure to filter DataOfSensors between
formattedFromDate and formattedToDateĐ               query = `Đ
                    SELECT *Đ
```

```
                                                FROM [dbo].[DataOfSensors]Đ
                    WHERE [DateTime] >= '${formattedFromDate}' AND [DateTime]
<= '${formattedToDate}'Đ                   ORDER BY [DateTime] DESCĐ
                `;Đ              break;Đ           case 'LoginLogout':Đ
                query = `SELECT [Username], [Login_Time], [Logout_Time],
[Description], [Group], [DateTime] FROM [dbo].[LoginLogout] ORDER BY
[DateTime] DESC;`;Đ              break;Đ           case 'System_Alarms':Đ
                query = `Đ                       SELECT Đ
[SNo], Đ                       [Date], Đ                       [Time], Đ
                        [Status], Đ                       [SystemAlarms], Đ
                        [Username], Đ                       [GroupName]Đ
                    FROM [dbo].[System_Alarms]Đ                    ORDER BY
[Date] DESC, [Time] DESCĐ              `;Đ             break;Đ
default:Đ              query = `Đ                  SELECT * FROM [dbo].
[${tableName}]Đ              `;Đ              break;Đ       }Đ       const
result = await pool.request().query(query);Đ       if
(result.recordset.length > 0) {Đ          res.status(200).json({ success:
true, data: result.recordset });Đ       } else {Đ
res.status(404).json({ success: false, message: 'No data found.' });Đ       }Đ
    } catch (error) {Đ       console.error('Database error:', error.message
|| error);Đ       res.status(500).json({ success: false, message: `Internal
server error: ${error.message || error}` });Đ   }Đ});Đ// Start the serverĐ
app.listen(port, () => {Đ   console.log(`Server is running on http://
localhost:${port}`);Đ});Đ
```

vscode-chartjs\out\chart.preview.js

```
'use strict';Đvar __awaiter = (this && this.__awaiter) || function (thisArg,
_arguments, P, generator) {Đ    function adopt(value) { return value
instanceof P ? value : new P(function (resolve) { resolve(value); }); }Đ
return new (P || (P = Promise))(function (resolve, reject) {Đ        function
fulfilled(value) { try { step(generator.next(value)); } catch (e)
{ reject(e); } }Đ        function rejected(value) { try
{ step(generator["throw"](value)); } catch (e) { reject(e); } }Đ
function step(result) { result.done ? resolve(result.value) :
adopt(result.value).then(fulfilled, rejected); }Đ        step((generator =
generator.apply(thisArg, _arguments || [])).next());Đ    });Đ};Đ
Object.defineProperty(exports, "__esModule", { value: true });Đconst vscode_1
= require("vscode");Đconst path = require("path");Đconst json5 =
require("json5");Đconst config = require("./config");Đconst logger_1 =
require("./logger");Đconst preview_manager_1 = require("./preview.manager");Đ/**Đ
 * Chart preview web panel serializer for restoring previews on vscode reload.Đ
 */Đclass ChartPreviewSerializer {Đ    /**Đ     * Creates new webview serializer.Đ
     * @param viewType Web view type.Đ     * @param extensionPath Extension
path for loading scripts, examples and data.Đ     * @param template Webview
preview html template.Đ     */Đ    constructor(viewType, extensionPath,
template) {Đ        this.viewType = viewType;Đ        this.extensionPath =
extensionPath;Đ        this.template = template;Đ        this._logger = new
logger_1.Logger(`${this.viewType}.serializer:`, config.logLevel);Đ    }Đ    /**Đ
     * Restores webview panel on vscode reload for chart and data previews.Đ
     * @param webviewPanel Webview panel to restore.Đ     * @param state Saved
web view panel state.Đ     */Đ    deserializeWebviewPanel(webviewPanel, state) {Đ
        return __awaiter(this, void 0, void 0, function* () {Đ
this._logger.logMessage(logger_1.LogLevel.Debug, 'deserializeWeviewPanel():
url:', state.uri.toString());Đ            const viewColumn =
(webviewPanel.viewColumn) ? webviewPanel.viewColumn : vscode_1.ViewColumn.One;Đ
            preview_manager_1.previewManager.add(new
ChartPreview(this.viewType, this.extensionPath,
vscode_1.Uri.parse(state.uri), viewColumn, this.template, webviewPanel));Đ
        });Đ    }Đ}Đexports.ChartPreviewSerializer = ChartPreviewSerializer;Đ/**Đ
* Main chart preview webview implementation for this vscode extension.Đ*/Đ
class ChartPreview {Đ    /**Đ     * Creates new Chart preview.Đ     * @param
viewType Preview webview type, i.e. chart.preview or chart.samples.Đ     *
@param extensionPath Extension path for loading webview scripts, etc.Đ     *
@param uri Chart spec json doc uri to preview.Đ     * @param viewColumn vscode
IDE view column to display chart preview in.Đ     * @param template Webview
html template reference.Đ     * @param panel Optional webview panel reference
for restore on vscode IDE reload.Đ     */Đ    constructor(viewType,
extensionPath, uri, viewColumn, template, panel) {Đ        this._disposables =
[];Đ        this._html = '';Đ        // save ext path, document uri, and create
prview uriĐ        this._extensionPath = extensionPath;Đ        this._uri = uri;Đ
        this._fileName = path.basename(uri.fsPath);Đ        this._previewUri =
this._uri.with({ scheme: 'chart' });Đ        this._logger = new
logger_1.Logger(`${viewType}:`, config.logLevel);Đ        // create preview
panel titleĐ        switch (viewType) {Đ            case 'chart.preview':Đ
                this._title = this._fileName;Đ                break;Đ
            case 'chart.samples':Đ                this._title = 'Chart
Samples';Đ                break;Đ            default: // chart.helpĐ
                this._title = 'Charts Help';Đ                break;Đ        }Đ
        // create html template for the webview with scripts path replacedĐ
        const scriptsPath = vscode_1.Uri.file(path.join(this._extensionPath,
'./node_modules/chart.js/dist'))Đ            .with({ scheme: 'vscode-
resource' }).toString(true);Đ        if (template) {Đ            this._html =
template.content.replace(/\{scripts\}/g, scriptsPath);Đ        }Đ        //
initialize webview panelĐ        this._panel = this.initWebview(viewType,
viewColumn, panel);Đ
```

```
                        this.configure();Ð   } // end of constructor()Ð   /
**Ð    * Initializes chart preview webview panel.Ð    * @param viewType
Preview webview type, i.e. chart.preview or chart.samples view.Ð    * @param
viewColumn vscode IDE view column to display preview in.Ð    * @param
viewPanel Optional web view panel to initialize.Ð    */Ð
initWebview(viewType, viewColumn, viewPanel) {Ð       if (!viewPanel) {Ð
            // create new webview panelÐ           viewPanel =
vscode_1.window.createWebviewPanel(viewType, this._title, viewColumn,
this.getWebviewOptions());Ð           viewPanel.iconPath =
vscode_1.Uri.file(path.join(this._extensionPath, './images/chart.svg'));Ð
            this._panel = viewPanel;Ð       }Ð       else {Ð
this._panel = viewPanel;Ð       }Ð       // dispose preview panel Ð
viewPanel.onDidDispose(() => {Ð           this.dispose();Ð       }, null,
this._disposables);Ð       // TODO: handle view state changes laterÐ
viewPanel.onDidChangeViewState((viewStateEvent) => {Ð           let active =
viewStateEvent.webviewPanel.visible;Ð       }, null, this._disposables);Ð
        // process web view messagesÐ
this.webview.onDidReceiveMessage(message => {Ð           switch
(message.command) {Ð               case 'refresh':Ð
this.refresh();Ð                   break;Ð               case 'openFile':Ð

vscode_1.workspace.openTextDocument(this._uri).then(document => {Ð
                        vscode_1.window.showTextDocument(document,
vscode_1.ViewColumn.One);Ð                   });Ð                   break;Ð
                case 'showHelp':Ð                   const helpUri =
vscode_1.Uri.parse('https://github.com/RandomFractals/vscode-chartjs#usage');Ð
                    vscode_1.commands.executeCommand('vscode.open', helpUri);Ð
                    break;Ð           }Ð       }, null, this._disposables);Ð
        return viewPanel;Ð   } // end of initWebview()Ð   /**Ð    * Creates
webview options with local resource roots, etcÐ    * for chart preview
webview display.Ð    */Ð   getWebviewOptions() {Ð       return {Ð
enableScripts: true,Ð           enableCommandUris: true,Ð
retainContextWhenHidden: true,Ð           localResourceRoots:
this.getLocalResourceRoots()Ð       };Ð   }Ð   /**Ð    * Creates local
resource roots for loading scripts in chart preview webview.Ð    */Ð
getLocalResourceRoots() {Ð       const localResourceRoots = [];Ð       const
workspaceFolder = vscode_1.workspace.getWorkspaceFolder(this.uri);Ð       if
(workspaceFolder) {Ð           localResourceRoots.push(workspaceFolder.uri);Ð
        }Ð       else if (!this.uri.scheme || this.uri.scheme === 'file') {Ð

localResourceRoots.push(vscode_1.Uri.file(path.dirname(this.uri.fsPath)));Ð
        }Ð       // add chart preview js scriptsÐ
localResourceRoots.push(vscode_1.Uri.file(path.join(this._extensionPath, './
node_modules/chart.js/dist')));Ð
this._logger.logMessage(logger_1.LogLevel.Debug, 'getLocalResourceRoots():',
localResourceRoots);Ð       return localResourceRoots;Ð   }Ð   /**Ð    *
Configures webview html for preview.Ð    */Ð   configure() {Ð
this.webview.html = this.html;Ð       // NOTE: let webview fire refresh
messageÐ       // when chart preview DOM content is initializedÐ       //
see: this.refresh();Ð   }Ð   /**Ð    * Reload chart preview on chart json doc
save changes or vscode IDE reload.Ð    */Ð   refresh() {Ð       // reveal
corresponding chart preview panelÐ
this._panel.reveal(this._panel.viewColumn, true); // preserve focusÐ       //
open chart json config text documentÐ
vscode_1.workspace.openTextDocument(this.uri).then(document => {Ð
this._logger.logMessage(logger_1.LogLevel.Debug, 'refresh(): file:',
this._fileName);Ð           const chartSpec = document.getText();Ð
try {Ð               const chartConfig = json5.parse(chartSpec);Ð
                this.webview.postMessage({Ð                   command:
'refresh',Ð
```

```
                              fileName: this._fileName,Ð
uri: this._uri.toString(),Ð                    config: chartConfig,Ð
                })；Ð          }Ð           catch (error) {Ð
this._logger.logMessage(logger_1.LogLevel.Error, 'refresh():', error.message);Ð
                this.webview.postMessage({ error: error });Ð           }Ð
        });Ð   }Ð   /**Ð    * Disposes this preview resources.Ð    */Ð
dispose() {Ð       preview_manager_1.previewManager.remove(this);Ð
this._panel.dispose();Ð       while (this._disposables.length) {Ð
const item = this._disposables.pop();Ð           if (item) {Ð
item.dispose();Ð           }Ð       }Ð   }Ð   /**Ð    * Gets preview panel
visibility status.Ð    */Ð   get visible() {Ð       return
this._panel.visible;Ð   }Ð   /**Ð    * Gets the underlying webview instance
for this preview.Ð    */Ð   get webview() {Ð       return this._panel.webview;Ð
    }Ð   /**Ð    * Gets the source chart spec json doc uri for this preview.Ð
     */Ð   get uri() {Ð       return this._uri;Ð   }Ð   /**Ð    * Gets the
preview uri to load on commands triggers or vscode IDE reload.Ð    */Ð   get
previewUri() {Ð       return this._previewUri;Ð   }Ð   /**Ð    * Gets the
html content to load for this preview.Ð    */Ð   get html() {Ð       return
this._html;Ð   }Ð}Ðexports.ChartPreview = ChartPreview;Ð//#
sourceMappingURL=chart.preview.js.map
```

vscode-chartjs\out\config.js

```
"use strict";ĐObject.defineProperty(exports, "__esModule", { value: true });Đ
const logger_1 = require("./logger");Đ// log level setting for prod. vs. dev
run of this ext.Đexports.logLevel = logger_1.LogLevel.Info; // change
to .Debug for ext. dev debugĐexports.supportedDataFiles = /.*\.(chart.json5)/;Đ
//# sourceMappingURL=config.js.map
```

vscode-chartjs\out\extension.js

```
"use strict";Đar __awaiter = (this && this.__awaiter) || function (thisArg,
_arguments, P, generator) {Đ   function adopt(value) { return value
instanceof P ? value : new P(function (resolve) { resolve(value); }); }Đ
return new (P || (P = Promise))(function (resolve, reject) {Đ       function
fulfilled(value) { try { step(generator.next(value)); } catch (e)
{ reject(e); } }Đ       function rejected(value) { try
{ step(generator["throw"](value)); } catch (e) { reject(e); } }Đ
function step(result) { result.done ? resolve(result.value) :
adopt(result.value).then(fulfilled, rejected); }Đ       step((generator =
generator.apply(thisArg, _arguments || [])).next());Đ   });Đ;Đ
Object.defineProperty(exports, "__esModule", { value: true });Đonst vscode_1
= require("vscode");Đonst fs = require("fs");Đonst path = require("path");Đ
const config = require("./config");Đonst logger_1 = require("./logger");Đonst
chart_preview_1 = require("./chart.preview");Đonst preview_manager_1 =
require("./preview.manager");Đonst template_manager_1 = require("./
template.manager");Đ/ supported chart config json file extensionsĐonst
CHART_FILE_EXTENSIONS = [Đ   '.chart.json5'Đ;Đonst logger = new
logger_1.Logger('chart.preview:', config.logLevel);Đ**Đ* Activates this
extension per rules set in package.json.Đ* @param context vscode extension
context.Đ* @see https://code.visualstudio.com/api/references/activation-
events for more info.Đ*/Đunction activate(context) {Đ   const extensionPath =
context.extensionPath;Đ   // logger.logMessage(LogLevel.Info, 'activate():
activating from extPath:', context.extensionPath);Đ   // initialize charts
preview webview panel templatesĐ   const templateManager = new
template_manager_1.TemplateManager(context.asAbsolutePath('templates'));Đ
const chartPreviewTemplate =
templateManager.getTemplate('chart.preview.html');Đ   const
chartSamplesTemplate = templateManager.getTemplate('chart.samples.html');Đ
    // register chart preview serializer for restore on vscode restartĐ
vscode_1.window.registerWebviewPanelSerializer('chart.preview', new
chart_preview_1.ChartPreviewSerializer('chart.preview', extensionPath,
chartPreviewTemplate));Đ   // register chart samples serializer for restore
on vscode restartĐ
vscode_1.window.registerWebviewPanelSerializer('chart.samples', new
chart_preview_1.ChartPreviewSerializer('chart.samples', extensionPath,
chartSamplesTemplate));Đ   // add Chart: Samples commandĐ   const
chartListCommand = vscode_1.commands.registerCommand('chart.samples', () =>
showChartList(context.asAbsolutePath('samples'), 'chart.json5'));Đ
context.subscriptions.push(chartListCommand);Đ   // add Chart: Preview commandĐ
    const chartWebview = createChartPreviewCommand('chart.preview',
extensionPath, chartPreviewTemplate);Đ
context.subscriptions.push(chartWebview);Đ   // refresh associated preview on
chart config file saveĐ   vscode_1.workspace.onDidSaveTextDocument((document)
=> {Đ       if (isChartConfigFile(document)) {Đ           const uri =
document.uri.with({ scheme: 'vega' });Đ           const preview =
preview_manager_1.previewManager.find(uri);Đ           if (preview) {Đ
                preview.refresh();Đ           }Đ       }Đ   });Đ   // reset
associated preview on chart config file changeĐ
vscode_1.workspace.onDidChangeTextDocument((changeEvent) => {Đ       if
(isChartConfigFile(changeEvent.document)) {Đ           const uri =
changeEvent.document.uri.with({ scheme: 'chart' });Đ           const preview
= preview_manager_1.previewManager.find(uri);Đ           if (preview &&
changeEvent.contentChanges.length > 0) {Đ              // TODO: add refresh
interval before enabling thisĐ               // preview.refresh();Đ
            }Đ       }Đ   });Đ   // reset all previews on config changeĐ
vscode_1.workspace.onDidChangeConfiguration(() => {Đ
preview_manager_1.previewManager.configure();Đ   });Đ
logger.logMessage(logger_1.LogLevel.Info, 'activate(): activated! extPath:',
context.extensionPath);Đ
```

```
                    } // end of activate()Ðexports.activate = activate;Ð/**Ð
* Deactivates this vscode extension to free up resources.Ð*/Ðfunction
deactivate() {Ð   // TODO: add extension cleanup code, if neededÐ}Ð
exports.deactivate = deactivate;Ð/**Ð* Creates chart preview command.Ð* @param
viewType Preview command type.Ð* @param extensionPath Extension path for
loading scripts, examples and data.Ð* @param viewTemplate Preview html
template.Ð*/Ðfunction createChartPreviewCommand(viewType, extensionPath,
viewTemplate) {Ð   const chartWebview =
vscode_1.commands.registerCommand(viewType, (uri) => {Ð       let resource =
uri;Ð       let viewColumn = getViewColumn();Ð       if (!(resource
instanceof vscode_1.Uri)) {Ð           if (vscode_1.window.activeTextEditor) {Ð
                resource = vscode_1.window.activeTextEditor.document.uri;Ð
            }Ð           else {Ð
vscode_1.window.showInformationMessage('Open a chart config json5 file to
Preview.');Ð               return;Ð           }Ð       }Ð       const preview
= new chart_preview_1.ChartPreview(viewType, extensionPath, resource,
viewColumn, viewTemplate);Ð
preview_manager_1.previewManager.add(preview);Ð       return preview.webview;Ð
    });Ð   return chartWebview;Ð}Ð/**Ð* Gets 2nd panel view column if chart
json config document is open.Ð*/Ðfunction getViewColumn() {Ð   let viewColumn
= vscode_1.ViewColumn.One;Ð   const activeEditor =
vscode_1.window.activeTextEditor;Ð   if (activeEditor &&
activeEditor.viewColumn) {Ð       viewColumn = activeEditor.viewColumn + 1;Ð
    }Ð   return viewColumn;Ð}Ð/**Ð* Checks if the vscode text document is a
chart config json file.Ð* @param document The vscode text document to check.Ð
*/Ðfunction isChartConfigFile(document) {Ð   const fileName =
path.basename(document.uri.fsPath).replace('.json5', ''); // strip out .json5
extÐ   const fileExt = fileName.substr(fileName.lastIndexOf('.'));Ð
logger.logMessage(logger_1.LogLevel.Debug, 'isChartConfigFile(): document:',
document);Ð   logger.logMessage(logger_1.LogLevel.Debug,
'isChartConfigFile(): file:', fileName);Ð   return
CHART_FILE_EXTENSIONS.findIndex(chartFileExt => chartFileExt === fileExt) >=
0;Ð}Ð/**Ð* Displays chart samples list to preview.Ð* @param examplesPath
Samples file path.Ð* @param examplesExtension Samples extension: chart.json5
for now.Ð*/Ðfunction showChartList(examplesPath, examplesExtension) {Ð
return __awaiter(this, void 0, void 0, function* () {Ð       const fileNames
= fs.readdirSync(examplesPath).filter(f => f.endsWith(examplesExtension));Ð
        const fileItems = [];Ð       fileNames.forEach(fileName =>
fileItems.push({ label: `Ø=ÜÊ ${fileName}` }));Ð   const selectedExample =
yield vscode_1.window.showQuickPick(fileItems, { canPickMany: false });Ð
        if (selectedExample) {Ð           const exampleFileName =
selectedExample.label.replace('Ø=ÜÊ ', '');Ð       const exampleFileUri =
vscode_1.Uri.file(path.join(examplesPath, exampleFileName));Ð
vscode_1.workspace.openTextDocument(exampleFileUri).then(document => {Ð
                vscode_1.window.showTextDocument(document,
vscode_1.ViewColumn.One);Ð           });Ð       }Ð   });Ð}Ð//#
sourceMappingURL=extension.js.map
```

vscode-chartjs\out\logger.js

```
"use strict";ÐObject.defineProperty(exports, "__esModule", { value: true });Ð//
supported log levelsÐvar LogLevel;Ð(function (LogLevel) {Ð
LogLevel[LogLevel["Debug"] = 0] = "Debug";Ð   LogLevel[LogLevel["Warn"] = 1]
= "Warn";Ð   LogLevel[LogLevel["Info"] = 2] = "Info";Ð
LogLevel[LogLevel["Error"] = 3] = "Error";Ð})(LogLevel = exports.LogLevel ||
(exports.LogLevel = {}));Ðclass Logger {Ð   /**Ð    * Creates new logger
instance.Ð    * @param category Logger category, usually the source class
name.Ð    * @param logLevel Log level to use or supress logging.Ð    */Ð
constructor(category, logLevel = LogLevel.Debug) {Ð       this.category =
category;Ð       this.logLevel = logLevel;Ð   }Ð   /**Ð    * Logs new message.Ð
     * @param logLevel Log message level.Ð    * @param message Log message.Ð
     * @param params Log message params, if any.Ð    */Ð
logMessage(logLevel, message, params = '') {Ð       if (logLevel >=
this.logLevel) {Ð           if (params) {Ð               this.log(logLevel,
message, params);Ð           }Ð           else {Ð
this.log(logLevel, message);Ð           }Ð       }Ð   }Ð   /**Ð    * Logs new
debug message.Ð    * @param message Debug log message.Ð    * @param params
Debug log message params, if any.Ð    */Ð   debug(message, params = '') {Ð
        if (this.logLevel <= LogLevel.Debug) {Ð           if (typeof params
=== 'object') {Ð               params = JSON.stringify(params, null, 2);Ð
            }Ð           this.log(LogLevel.Debug, message, params);Ð       }Ð
    }Ð   /**Ð    * Logs new error message.Ð    * @param message Error log
message.Ð    * @param params Error log message params, if any.Ð    */Ð
error(message, params = '') {Ð       if (typeof params === 'object') {Ð
            params = JSON.stringify(params, null, 2);Ð       }Ð
this.log(LogLevel.Error, message, params);Ð   }Ð   /**Ð    * Logs new message
to console based on the specified log level.Ð    * @param logLevel Log
message level.Ð    * @param message Log message.Ð    * @param params Log
message params, if any.Ð    */Ð   log(logLevel, message, params = '') {Ð
        switch (logLevel) {Ð           case LogLevel.Warn:Ð
console.warn(this.category + message, params);Ð               break;Ð
            case LogLevel.Info:Ð               console.info(this.category +
message, params);Ð               break;Ð           case LogLevel.Error:Ð
                console.error(this.category + message, params);Ð
                break;Ð           default: // debugÐ
console.log(this.category + message, params);Ð               break;Ð       }Ð
    }Ð}Ðexports.Logger = Logger;Ð//# sourceMappingURL=logger.js.map
```

vscode-chartjs\out\preview.manager.js

```
'use strict';Đbject.defineProperty(exports, "__esModule", { value: true });Đ
class PreviewManager {Đ   constructor() {Đ       // tracked previews for
config/restore updatesĐ       this._previews = [];Đ   }Đ   /**Đ    * Creates
preview manager singleton instance.Đ    */Đ   static get Instance() {Đ
return this._instance || (this._instance = new this());Đ   }Đ   /**Đ    *
Adds new preview instance for config/restore tracking.Đ    * @param preview
preview instance to add.Đ    */Đ   add(preview) {Đ
this._previews.push(preview);Đ   }Đ   /**Đ    * Removes preview instance from
previews tracking collection.Đ    * @param preview preview instance to remove.Đ
     */Đ   remove(preview) {Đ       let found =
this._previews.indexOf(preview);Đ       if (found >= 0) {Đ
this._previews.splice(found, 1);Đ       }Đ   }Đ   /**Đ    * Returns matching
preview for the specified uri.Đ    * @param uri preview uri.Đ    */Đ
find(uri) {Đ       return this._previews.find(p => p.previewUri.toString()
=== uri.toString());Đ   }Đ   /**Đ    * Returns active preview instance.Đ    */Đ
    active() {Đ       return this._previews.find(p => p.visible);Đ   }Đ   /**Đ
     * Reloads open previews on extension config changes.Đ    */Đ
configure() {Đ       this._previews.forEach(p => p.configure());Đ   }ĐĐ
exports.PreviewManager = PreviewManager;Đ/ export preview manager singletonĐ
exports.previewManager = PreviewManager.Instance;Đ/#
sourceMappingURL=preview.manager.js.map
```

vscode-chartjs\out\template.manager.js

```
"use strict";Đbject.defineProperty(exports, "__esModule", { value: true });Đ
const fs = require("fs");Đonst path = require("path");Đonst config =
require("./config");Đonst logger_1 = require("./logger");Đ**Đ* Template type
for loading file templates and template file content.Đ*/Đlass Template {Đ   /
**Đ    * Creates new templateĐ    * @param name Template name.Đ    * @param
content Template file content.Đ    */Đ   constructor(name = '', content = '')
{Đ       this.name = name;Đ       this.content = content;Đ   }Đ   /**Đ    *
Injects template content params by replacing {} tokens with regex.Đ    *
@param params Template key/value pair params to inject.Đ    */Đ
replace(params) {Đ       let templateContent = this.content;Đ
Object.keys(params).map(key => {Đ           templateContent =
templateContent.replace(RegExp(`{${key}}`, 'g'), params[key]);Đ       });Đ
        return templateContent;Đ   }ĐĐxports.Template = Template;Đ**Đ*
Template manager implementation for html and json files.Đ*/Đlass
TemplateManager {Đ   /**Đ    * Creates new template manager and loads
templatesĐ    * from the specified template folder.Đ    * @param
templateFolder Template folder to inspect.Đ    */Đ
constructor(templateFolder) {Đ       this.templateFolder = templateFolder;Đ
        this.logger = new logger_1.Logger('template.manager:',
config.logLevel);Đ       this.templates = this.loadTemplates();Đ   }Đ   /**Đ
     * Loads .html and .json templates from the specified template folder.Đ
     * @param templateFolder Template folder to inspect.Đ    */Đ
loadTemplates() {Đ       this.logger.debug('loadTemplates(): loading file
templates... \n templateFolder:', this.templateFolder);Đ       const
fileNames = fs.readdirSync(this.templateFolder)Đ           .filter(fileName
=> fileName.endsWith('.html') || fileName.endsWith('.json'));Đ       const
templates = [];Đ       // TODO: change this to read file async ???Đ
fileNames.forEach(fileName => templates.push(new Template(fileName,
fs.readFileSync(path.join(this.templateFolder, fileName), 'utf8')) // file
contentĐ       ));Đ       this.logger.debug('loadTemplates(): templates:',
fileNames);Đ       return templates;Đ   }Đ   /**Đ    * Gets file template
with the specified name.Đ    * @param name Template name to find.Đ    */Đ
getTemplate(name) {Đ       return this.templates.find(template =>
template.name === name);Đ   }ĐĐxports.TemplateManager = TemplateManager;Đ/#
sourceMappingURL=template.manager.js.map
```

vscode-chartjs\out\test\runTest.js

```
"use strict";Ðvar __awaiter = (this && this.__awaiter) || function (thisArg,
_arguments, P, generator) {Ð   function adopt(value) { return value
instanceof P ? value : new P(function (resolve) { resolve(value); }); }Ð
return new (P || (P = Promise))(function (resolve, reject) {Ð       function
fulfilled(value) { try { step(generator.next(value)); } catch (e)
{ reject(e); } }Ð       function rejected(value) { try
{ step(generator["throw"](value)); } catch (e) { reject(e); } }Ð
function step(result) { result.done ? resolve(result.value) :
adopt(result.value).then(fulfilled, rejected); }Ð       step((generator =
generator.apply(thisArg, _arguments || [])).next());Ð   });Ð};Ð
Object.defineProperty(exports, "__esModule", { value: true });Ðconst path =
require("path");Ðconst vscode_test_1 = require("vscode-test");Ðfunction main() {Ð
    return __awaiter(this, void 0, void 0, function* () {Ð       try {Ð
            // The folder containing the Extension Manifest package.jsonÐ
            // Passed to `--extensionDevelopmentPath`Ð           const
extensionDevelopmentPath = path.resolve(__dirname, '../../');Ð           //
The path to test runnerÐ           // Passed to --extensionTestsPathÐ
            const extensionTestsPath = path.resolve(__dirname, './suite/
index');Ð           // Download VS Code, unzip it and run the integration testÐ
            yield vscode_test_1.runTests({ extensionDevelopmentPath,
extensionTestsPath });Ð       }Ð       catch (err) {Ð
console.error('Failed to run tests');Ð           process.exit(1);Ð       }Ð
    });Ð}Ðmain();Ð//# sourceMappingURL=runTest.js.map
```

vscode-chartjs\out\test\suite\extension.test.js

```
"use strict";Đbject.defineProperty(exports, "__esModule", { value: true });Đ
const assert = require("assert");Đ/ You can import and use all API from the
'vscode' moduleĐ/ as well as import your extension to test itĐonst vscode =
require("vscode");Đ/ import * as myExtension from '../extension';Đ
suite('Extension Test Suite', () => {Đ
vscode.window.showInformationMessage('Start all tests.');Đ   test('Sample
test', () => {Đ       assert.equal(-1, [1, 2, 3].indexOf(5));Đ
assert.equal(-1, [1, 2, 3].indexOf(0));Đ   });Đ});Đ/#
sourceMappingURL=extension.test.js.map
```

vscode-chartjs\out\test\suite\index.js

```
"use strict";ĐObject.defineProperty(exports, "__esModule", { value: true });Đ
const path = require("path");Đconst Mocha = require("mocha");Đconst glob =
require("glob");Đfunction run() {Đ    // Create the mocha testĐ    const mocha =
new Mocha({Đ        ui: 'tdd',Đ    });Đ    mocha.useColors(true);Đ    const
testsRoot = path.resolve(__dirname, '..');Đ    return new Promise((c, e) => {Đ
        glob('**/**.test.js', { cwd: testsRoot }, (err, files) => {Đ
            if (err) {Đ                return e(err);Đ            }Đ
            // Add files to the test suiteĐ            files.forEach(f =>
mocha.addFile(path.resolve(testsRoot, f)));Đ            try {Đ
                // Run the mocha testĐ                mocha.run(failures => {Đ
                    if (failures > 0) {Đ                        e(new
Error(`${failures} tests failed.`));Đ                    }Đ
else {Đ                        c();Đ                    }Đ                });Đ
            }Đ            catch (err) {Đ                e(err);Đ            }Đ
        });Đ    });Đ}Đexports.run = run;Đ//# sourceMappingURL=index.js.map
```

vscode-chartjs\out\vega.preview.js

```
'use strict';Đar __awaiter = (this && this.__awaiter) || function (thisArg,
_arguments, P, generator) {Đ   function adopt(value) { return value
instanceof P ? value : new P(function (resolve) { resolve(value); }); }Đ
return new (P || (P = Promise))(function (resolve, reject) {Đ       function
fulfilled(value) { try { step(generator.next(value)); } catch (e)
{ reject(e); } }Đ       function rejected(value) { try
{ step(generator["throw"](value)); } catch (e) { reject(e); } }Đ
function step(result) { result.done ? resolve(result.value) :
adopt(result.value).then(fulfilled, rejected); }Đ       step((generator =
generator.apply(thisArg, _arguments || [])).next());Đ   });Đ;Đ
Object.defineProperty(exports, "__esModule", { value: true });Đonst vscode_1
= require("vscode");Đonst fs = require("fs");Đonst path = require("path");Đ
const config = require("./config");Đonst logger_1 = require("./logger");Đonst
preview_manager_1 = require("./preview.manager");Đ**Đ* Vega preview web panel
serializer for restoring previews on vscode reload.Đ*/Đlass
VegaPreviewSerializer {Đ   /**Đ    * Creates new webview serializer.Đ    *
@param viewType Web view type.Đ    * @param extensionPath Extension path for
loading scripts, examples and data.Đ    * @param template Webview preview
html template.Đ    */Đ   constructor(viewType, extensionPath, template) {Đ
        this.viewType = viewType;Đ       this.extensionPath = extensionPath;Đ
        this.template = template;Đ       this._logger = new
logger_1.Logger(`${this.viewType}.serializer:`, config.logLevel);Đ   }Đ   /**Đ
     * Restores webview panel on vscode reload for vega and data previews.Đ
     * @param webviewPanel Webview panel to restore.Đ    * @param state Saved
web view panel state.Đ    */Đ   deserializeWebviewPanel(webviewPanel, state) {Đ
        return __awaiter(this, void 0, void 0, function* () {Đ
this._logger.logMessage(logger_1.LogLevel.Debug, 'deserializeWeviewPanel():
url:', state.uri.toString());Đ
preview_manager_1.previewManager.add(new VegaPreview(this.viewType,
this.extensionPath, vscode_1.Uri.parse(state.uri), webviewPanel.viewColumn,
this.template, webviewPanel));Đ       });Đ   }ĐĐxports.VegaPreviewSerializer
= VegaPreviewSerializer;Đ**Đ* Main vega preview webview implementation for
this vscode extension.Đ*/Đlass VegaPreview {Đ   /**Đ    * Creates new Vega
preview.Đ    * @param viewType Preview webview type, i.e. vega.preview or
vega.data.preview.Đ    * @param extensionPath Extension path for loading
webview scripts, etc.Đ    * @param uri Vega spec json doc uri to preview.Đ
* @param viewColumn vscode IDE view column to display vega preview in.Đ    *
@param template Webview html template reference.Đ    * @param panel Optional
webview panel reference for restore on vscode IDE reload.Đ    */Đ
constructor(viewType, extensionPath, uri, viewColumn, template, panel) {Đ
        this._disposables = [];Đ       // save ext path, document uri, and
create prview uriĐ       this._extensionPath = extensionPath;Đ
this._uri = uri;Đ       this._fileName = path.basename(uri.fsPath);Đ
this._previewUri = this._uri.with({ scheme: 'vega' });Đ       this._logger =
new logger_1.Logger(`${viewType}:`, config.logLevel);Đ       // create
preview panel titleĐ       switch (viewType) {Đ           case 'vega.preview':Đ
                this._title = this._fileName;Đ               break;Đ
            case 'vega.visual.vocabulary':Đ               this._title =
'Visual Vocabulary';Đ               break;Đ           default: // vega.helpĐ
                this._title = 'Vega Help';Đ               break;Đ       }Đ
        // create html template for the webview with scripts path replacedĐ
        const scriptsPath = vscode_1.Uri.file(path.join(this._extensionPath,
'scripts'))Đ           .with({ scheme: 'vscode-resource' }).toString(true);Đ
        this._html = template.content.replace(/\{scripts\}/g, scriptsPath);Đ
        // initialize webview panelĐ       this._panel = panel;Đ
this.initWebview(viewType, viewColumn);Đ       this.configure();Đ   } // end
of constructor()Đ   /**Đ    * Initializes vega preview webview panel.Đ    *
@param viewType Preview webview type, i.e. vega.preview or vega.data.preview.Đ
     * @param viewColumn vscode IDE view column to display preview in.Đ
```

```
                                                                         */Đ
    initWebview(viewType, viewColumn) {Đ       if (!this._panel) {Đ
            // create new webview panelĐ           this._panel =
vscode_1.window.createWebviewPanel(viewType, this._title, viewColumn,
this.getWebviewOptions());Đ           let panelIconPath;Đ           switch
(viewType) {Đ               case 'vega.preview':Đ
panelIconPath = './images/vega-viewer.svg';Đ                    break;Đ
                case 'vega.visual.vocabulary':Đ
panelIconPath = './images/visual-vocabulary.svg';Đ                    break;Đ
                default: // vega.help, etc.Đ                    panelIconPath
= './images/vega-viewer.svg';Đ                   break;Đ           }Đ
            this._panel.iconPath =
vscode_1.Uri.file(path.join(this._extensionPath, panelIconPath));Đ       }Đ
        // dispose preview panel Đ       this._panel.onDidDispose(() => {Đ
            this.dispose();Đ       }, null, this._disposables);Đ       //
TODO: handle view state changes laterĐ
this._panel.onDidChangeViewState((viewStateEvent) => {Đ           let active
= viewStateEvent.webviewPanel.visible;Đ       }, null, this._disposables);Đ
        // process web view messagesĐ
this.webview.onDidReceiveMessage(message => {Đ           switch
(message.command) {Đ               case 'refresh':Đ
this.refresh();Đ               break;Đ               case 'exportSvg':Đ
                    this.exportSvg(message.svg);Đ                    break;Đ
                case 'exportPng':Đ
this.exportPng(message.imageData);Đ                    break;Đ
case 'openFile':Đ
vscode_1.workspace.openTextDocument(this._uri).then(document => {Đ
                        vscode_1.window.showTextDocument(document,
vscode_1.ViewColumn.One);Đ                  });Đ                    break;Đ
                case 'showData':Đ
this.showData(message.dataUri);Đ                    break;Đ               case
'showHelp':Đ                    const helpUri = vscode_1.Uri.parse('https://
github.com/RandomFractals/vscode-vega-viewer#usage');Đ
vscode_1.commands.executeCommand('vscode.open', helpUri);Đ
break;Đ           }Đ       }, null, this._disposables);Đ   } // end of
initWebview()Đ   /**Đ    * Creates webview options with local resource roots,
etcĐ    * for vega preview webview display.Đ    */Đ   getWebviewOptions() {Đ
        return {Đ           enableScripts: true,Đ
enableCommandUris: true,Đ           retainContextWhenHidden: true,Đ
localResourceRoots: this.getLocalResourceRoots()Đ       };Đ   }Đ   /**Đ    *
Creates local resource roots for loading scripts in vega preview webview.Đ
*/Đ   getLocalResourceRoots() {Đ       const localResourceRoots = [];Đ
const workspaceFolder = vscode_1.workspace.getWorkspaceFolder(this.uri);Đ
        if (workspaceFolder) {Đ
localResourceRoots.push(workspaceFolder.uri);Đ       }Đ       else if (!
this.uri.scheme || this.uri.scheme === 'file') {Đ
localResourceRoots.push(vscode_1.Uri.file(path.dirname(this.uri.fsPath)));Đ
        }Đ       // add vega preview js scriptsĐ
localResourceRoots.push(vscode_1.Uri.file(path.join(this._extensionPath,
'scripts')));Đ       this._logger.logMessage(logger_1.LogLevel.Debug,
'getLocalResourceRoots():', localResourceRoots);Đ       return
localResourceRoots;Đ   }Đ   /**Đ    * Configures webview html for preview.Đ
     */Đ   configure() {Đ       this.webview.html = this.html;Đ       //
NOTE: let webview fire refresh messageĐ       // when vega preview DOM
content is initializedĐ       // see: this.refresh();Đ   }Đ   /**Đ    *
Launches referenced vega spec csv or json data preview.Đ    * @param dataUrl
The url of the data file to load.Đ    */Đ   showData(dataUrl) {Đ       let
dataUri;Đ       if (dataUrl.startsWith('http://') ||
dataUrl.startsWith('https://')) {Đ           dataUri =
vscode_1.Uri.parse(dataUrl);Đ
```

```
                                    }Đ       else {Đ           // join with
vega spec file path for reletive data file loadingĐ           dataUri =
vscode_1.Uri.file(path.join(path.dirname(this._uri.fsPath), dataUrl));Đ
        }Đ       this._logger.logMessage(logger_1.LogLevel.Info, `showData():
${this.dataPreviewCommand}`, dataUri.toString(true));Đ
vscode_1.commands.executeCommand(this.dataPreviewCommand, dataUri);Đ   }Đ   /
**Đ    * Reload vega preview on vega spec json doc save changes or vscode IDE
reload.Đ    */Đ   refresh() {Đ       // reveal corresponding Vega preview
panelĐ       this._panel.reveal(this._panel.viewColumn, true); // preserve
focusĐ       // open Vega json spec text documentĐ
vscode_1.workspace.openTextDocument(this.uri).then(document => {Đ
this._logger.logMessage(logger_1.LogLevel.Debug, 'refresh(): file:',
this._fileName);Đ           const vegaSpec = document.getText();Đ
try {Đ               const spec = JSON.parse(vegaSpec);Đ               const
data = this.getData(spec);Đ               this.webview.postMessage({Đ
                    command: 'refresh',Đ                   fileName:
this._fileName,Đ                   uri: this._uri.toString(),Đ
                    spec: vegaSpec,Đ                   data: dataĐ
                });Đ           }Đ           catch (error) {Đ
this._logger.logMessage(logger_1.LogLevel.Error, 'refresh():', error.message);Đ
                this.webview.postMessage({ error: error });Đ           }Đ
        });Đ   }Đ   /**Đ    * Extracts data urls and loads local data files
to pass to vega preview webview.Đ    * @param spec Vega json doc spec root or
nested data references to extract.Đ    */Đ   getData(spec) {Đ       const
dataFiles = {};Đ       // get top level data urlsĐ       let dataUrls =
this.getDataUrls(spec);Đ       // add nested spec data urls for view
compositions (facets, repeats, etc.)Đ       dataUrls =
dataUrls.concat(this.getDataUrls(spec['spec']));Đ
this._logger.logMessage(logger_1.LogLevel.Debug, 'getData(): dataUrls:',
dataUrls);Đ       // get all local files dataĐ       dataUrls.forEach(dataUrl
=> {Đ           if (dataUrl.startsWith('http://') ||
dataUrl.startsWith('https://')) {Đ               // add remote data source
referenceĐ               dataFiles[dataUrl] = dataUrl;Đ           }Đ
            else {Đ               // get local file dataĐ               const
fileData = this.getFileData(dataUrl);Đ               if (fileData) {Đ
                    dataFiles[dataUrl] = fileData;Đ               }Đ
                this._logger.logMessage(logger_1.LogLevel.Debug, 'getData():
localDataUrl:', dataUrl);Đ           }Đ       });Đ       return dataFiles;Đ
    }Đ   /**Đ    * Recursively extracts data urls from the specified vega
json doc specĐ    * or knowwn nested data elements for loading local data
content.Đ    * @param spec Vega json doc spec root or nested data references
to extract.Đ    */Đ   getDataUrls(spec) {Đ       let dataUrls = [];Đ       if
(spec === undefined) {Đ           return dataUrls; // base caseĐ       }Đ
        const data = spec['data'];Đ       const transforms =
spec['transform'];Đ       let layers = [];Đ       layers =
layers.concat(spec['layer']);Đ       layers = layers.concat(spec['concat']);Đ
        layers = layers.concat(spec['hconcat']);Đ       layers =
layers.concat(spec['vconcat']);Đ       if (data !== undefined) {Đ
            // get top level data referencesĐ           if
(Array.isArray(data)) {Đ               data.filter(d => d['url'] !==
undefined).forEach(d => {Đ                   dataUrls.push(d['url']);Đ
                });Đ           }Đ           else if (data['url'] !==
undefined) {Đ               dataUrls.push(data['url']);Đ           }Đ       }Đ
        if (layers !== undefined && Array.isArray(layers)) {Đ           //
get layers data referencesĐ           layers.forEach(layer => {Đ
                dataUrls = dataUrls.concat(this.getDataUrls(layer));Đ
            });Đ       }Đ       if (transforms !== undefined) {Đ           //
get transform data referencesĐ           transforms.forEach(transformData => {Đ
                dataUrls =
dataUrls.concat(this.getDataUrls(transformData['from']));Đ
```

```
                                                                        });Đ
        }Đ        return dataUrls;Đ    }Đ    /**Đ     * Loads actual local data
file content.Đ     * @param filePath Local data file path.Đ     * TODO: change
this to async laterĐ     */Đ    getFileData(filePath) {Đ        let data = null;Đ
        const dataFilePath = path.join(path.dirname(this._uri.fsPath),
filePath);Đ        if (fs.existsSync(dataFilePath)) {Đ            data =
fs.readFileSync(dataFilePath, 'utf8');Đ        }Đ        else {Đ
this._logger.logMessage(logger_1.LogLevel.Error, 'getFileData():',
`${filePath} doesn't exist`);Đ        }Đ        return data;Đ    }Đ    /**Đ     *
Displays Save SVG dialog and saves it for export SVG feature from preview
panel.Đ     * @param svg Svg document export to save.Đ     */Đ    exportSvg(svg)
{Đ        return __awaiter(this, void 0, void 0, function* () {Đ
const svgFilePath = this._uri.fsPath.replace('.json', '');Đ           const
svgFileUri = yield vscode_1.window.showSaveDialog({Đ
defaultUri: vscode_1.Uri.parse(svgFilePath).with({ scheme: 'file' }),Đ
                filters: { 'SVG': ['svg'] }Đ            });Đ            if
(svgFileUri) {Đ               fs.writeFile(svgFileUri.fsPath, svg, (error) =>
{Đ                  if (error) {Đ                     const errorMessage =
`Failed to save file: ${svgFileUri.fsPath}`;Đ
this._logger.logMessage(logger_1.LogLevel.Error, 'exportSvg():',
errorMessage);Đ
vscode_1.window.showErrorMessage(errorMessage);Đ                   }Đ
                });Đ            }Đ
this.webview.postMessage({ command: 'showMessage', message: '' });Đ        });Đ
    }Đ    /**Đ     * Displays Save PNG dialog and saves it for export PNG
feature from preview panel.Đ     * @param imageData Image data to save in png
format.Đ     */Đ    exportPng(imageData) {Đ        return __awaiter(this, void
0, void 0, function* () {Đ           const base64 =
imageData.replace('data:image/png;base64,', '');Đ           const pngFilePath
= this._uri.fsPath.replace('.json', '');Đ           const pngFileUri = yield
vscode_1.window.showSaveDialog({Đ               defaultUri:
vscode_1.Uri.parse(pngFilePath).with({ scheme: 'file' }),Đ
filters: { 'PNG': ['png'] }Đ            });Đ           if (pngFileUri) {Đ
                fs.writeFile(pngFileUri.fsPath, base64, 'base64', (error) => {Đ
                    if (error) {Đ                    const errorMessage =
`Failed to save file: ${pngFileUri.fsPath}`;Đ
this._logger.logMessage(logger_1.LogLevel.Error, 'exportPng():',
errorMessage);Đ
vscode_1.window.showErrorMessage(errorMessage);Đ                   }Đ
                });Đ            }Đ
this.webview.postMessage({ command: 'showMessage', message: '' });Đ        });Đ
    }Đ    /**Đ     * Disposes this preview resources.Đ     */Đ    dispose() {Đ
        preview_manager_1.previewManager.remove(this);Đ
this._panel.dispose();Đ        while (this._disposables.length) {Đ
const item = this._disposables.pop();Đ           if (item) {Đ
item.dispose();Đ           }Đ        }Đ    }Đ    /**Đ     * Gets preview panel
visibility status.Đ     */Đ    get visible() {Đ        return
this._panel.visible;Đ    }Đ    /**Đ     * Gets the underlying webview instance
for this preview.Đ     */Đ    get webview() {Đ        return this._panel.webview;Đ
    }Đ    /**Đ     * Gets the source vega spec json doc uri for this preview.Đ
     */Đ    get uri() {Đ        return this._uri;Đ    }Đ    /**Đ     * Gets the
preview uri to load on commands triggers or vscode IDE reload.Đ     */Đ    get
previewUri() {Đ        return this._previewUri;Đ    }Đ    /**Đ     * Gets the
html content to load for this preview.Đ     */Đ    get html() {Đ        return
this._html;Đ    }Đ    /**Đ     * Gets vega data preview command setting.Đ     */Đ
    get dataPreviewCommand() {Đ        return
vscode_1.workspace.getConfiguration('vega.viewer').get('dataPreviewCommand');Đ
    }ĐĐexports.VegaPreview = VegaPreview;Đ//#
sourceMappingURL=vega.preview.js.map
```

angular\src\index.html

```
<!DOCTYPE html><html lang="en">  <head>    <meta charset="utf-8" />
<title>Angular</title>    <base href="/" />    <meta name="viewport"
content="width=device-width, initial-scale=1" />    <link rel="icon"
type="image/x-icon" href="favicon.ico" />  </head>  <body>    <app-root></app-
root>  </body></html>
```

index.html

```
<!DOCTYPE html>Ðhtml lang="en">Ðhead>Ð   <meta charset="UTF-8">Ð   <meta
name="viewport" content="width=device-width, initial-scale=1.0">Ð
<title>RIAC 2.1 - Login</title>Ð   <style>Ð       body {Ð
background-color: #008000;Ð           background-image:
url('./4893694.webp'); /* Background image */Ð           background-size:
cover;Ð           background-repeat: no-repeat;Ð           font-family:
Arial, sans-serif;Ð           margin: 0;Ð           padding: 0;Ð
position: fixed;Ð           width: 100%;Ð           height: 100%;Ð
display: flex;Ð           justify-content: center;Ð           align-items:
center;Ð       }Ð       .top-left-image {Ð           position: absolute;Ð
            top: 20px;Ð           left: 20px;Ð           width: 120px;Ð
            height: auto;Ð           background-color: rgba(8, 8, 8, 0.5); /*
Light smoke white */Ð           padding: 5px;Ð           border-radius: 10px;Ð
        }Ð       .login-container {Ð           display: flex;Ð
align-items: center;Ð           background: rgba(255, 255, 255, 0.1); /*
Transparent background */Ð           border-radius: 10px;Ð           box-
shadow: 0 4px 10px rgba(0, 0, 0, 5.0);Ð           padding: 20px 30px;Ð
            gap: 20px; /* Space between the image and form */Ð
padding-right: 90px;Ð       }Ð       .login-icon {Ð           width: 100px;Ð
            height: auto;Ð           position: fixed;Ð       }Ð       .login-
box {Ð           text-align: left;Ð           color: white;Ð
padding-left: 250px;Ð           margin-top: 100px;Ð           position:
sticky;Ð       }ÐÐ       .login-box h3 {Ð           margin-bottom: 20px;Ð
            font-size: 24px;Ð           font-weight: bold;Ð       }Ð
        .login-box input[type="text"],Ð       .login-box
input[type="password"] {Ð           width: 50%;Ð           padding: 10px;Ð
            margin-bottom: 15px;Ð           font-size: 16px;Ð
border: 1px solid #ccc;Ð           border-radius: 5px;Ð           background-
color: rgba(255, 255, 255, 0.9);Ð           color: black;Ð       }Ð
        .login-box button {Ð           width: 20%;Ð           padding: 10px;Ð
            margin-top: 10px;Ð           background: #ffcc00; /* Bright login
button */Ð           color: black;Ð           font-size: 18px;Ð
border: none;Ð           border-radius: 5px;Ð           cursor: pointer;Ð
            transition: background-color 0.3s;Ð       }Ð       .login-box
button:hover {Ð           background: #ff9900; /* Darker hover color */Ð
        }Ð       .footer {Ð           margin-top: 20px;Ð           font-size:
14px;Ð           color: white;Ð           text-align: center;Ð       }Ð
marquee {Ð           color: rgb(251, 255, 0);Ð           font-size: 24px;Ð
            font-weight: bold;Ð           margin-bottom: 20px;Ð       }Ð   </
style>Ð/head>Ðbody>Ð   <div class="top-left-image">Ð       <img
src="RIAC.PNG" alt="RIAC Logo" style="width: 100%; height: auto;">Ð   </div>Ð
    <div>Ð       <center>Ð           <marquee direction="left-right"
scrollamount="4">Ð               Ran Industrial Automation PVT LTD!Ð
            </marquee>Ð       </center>Ð       <div class="login-container">Ð
            <!-- Left side: Image -->Ð  Ð           <!-- Right side: Login
form -->Ð           <div class="login-box">Ð               <h3>Login</h3>Ð
                <form id="loginForm">Ð                   <input type="text"
id="username" placeholder="Login Name" required />Ð               <input
type="password" id="password" placeholder="Password" required /><br>Ð
                    <button type="submit">Login</button>Ð               </
form>Ð               <p id="error" style="color: red;"></p>Ð           </div>Ð
        </div>Ð       <div class="footer">Ð           <p>VERSION: 2.1.1</p>Ð
        </div>Ð   </div>Ð   <script>Ð
document.getElementById('loginForm').addEventListener('submit', async (e) => {Ð
            e.preventDefault();Ð           const username =
document.getElementById('username').value;Ð           const password =
document.getElementById('password').value;Ð           const errorElement =
document.getElementById('error');Ð           errorElement.textContent = '';Ð
            try {Ð
```

```
                                const response = await fetch('http://
localhost:5000/login', {Đ                   method: 'POST',Đ
                    headers: { 'Content-Type': 'application/json' },Đ
                    body: JSON.stringify({ username, password }),Đ
                });Đ               const result = await response.json();Đ
               Đ               if (response.ok) {Đ                   if
(result.success) {Đ                         // Save the username to
localStorage before redirectingĐ
localStorage.setItem("username", username);Đ
window.location.href = 'Sucess.html'; // Redirect to Sucess.htmlĐ
                    } else {Đ                       errorElement.textContent
= result.message || 'Invalid credentials. Please try again.';Đ
                    }Đ               } else {Đ
errorElement.textContent = 'An error occurred. Please try again later.';Đ
                }Đ          } catch (error) {Đ
console.error('Error during login:', error);Đ
errorElement.textContent = 'Network error: Could not reach the server. Please
check your internet connection and try again later.';Đ           }Đ      });Đ
    </script>Đ/body>Đ/html>Đ
```

sucess.html

```
<!DOCTYPE html>Đhtml lang="en">Đhead>Đ   <meta charset="UTF-8">Đ   <meta
name="viewport" content="width=device-width, initial-scale=1.0">Đ
<title>RIAC 2.1.1</title>Đ   <style>Đ       body {Đ           margin: 0;Đ
            font-family: Arial, sans-serif;Đ       }Đ       .container {Đ
            display: flex;Đ           height: 100vh;Đ       }Đ       .sidebar
{Đ           width: 250px;Đ           background-color: #008000;Đ
color: white;Đ           padding: 15px;Đ           display: flex;Đ
flex-direction: column;Đ           gap: 10px;Đ       }Đ       .sidebar h2 {Đ
            margin: 0;Đ           font-size: 20px;Đ           padding-bottom:
10px;Đ           border-bottom: 2px solid white;Đ       }Đ       .sidebar
button {Đ           background: none;Đ           color: white;Đ
border: none;Đ           text-align: left;Đ           cursor: pointer;Đ
            font-size: 16px;Đ           padding: 8px 0;Đ           border-
bottom: 1px solid white;Đ       }Đ       .sidebar button:hover {Đ
text-decoration: underline;Đ       }Đ       .sidebar .sub-buttons {Đ
            display: none;Đ           padding-left: 20px;Đ           gap: 5px;Đ
            flex-direction: column;Đ       }Đ       .sidebar .sub-buttons
button {Đ           font-size: 14px;Đ           padding-left: 0;  /* Ensures
buttons align vertically */Đ       }Đ       .sidebar button.active + .sub-
buttons {Đ           display: flex;Đ       }Đ       .content {Đ
flex-grow: 1;Đ           padding: 20px;Đ           background-color: rgb(218,
215, 228);Đ           position: relative;Đ           background-image:
url('RIAC.png'); /* Set the image */Đ           background-position: center;Đ
            background-repeat: no-repeat;Đ           background-size: cover;Đ
        }Đ     Đ       .content h1 {Đ           font-size: 24px;Đ
color: #333;Đ       }Đ       .content .company-info {Đ           color: #555;Đ
            font-size: 16px;Đ           margin-top: 20px;Đ       }Đ       /*
Positioning for logout and print buttons horizontally */Đ      /* Positioning
for logout and print buttons to the right side */Đ      .top-buttons-
container {Đ   display: flex;Đ   justify-content: space-between; /* Spread
out the buttons */Đ   align-items: center; /* Align items vertically centered
*/Đ   background-color: rgb(16, 67, 233); /* Green background for top bar */Đ
    color:white;Đ   padding: 10px 20px;Đ   margin-bottom: 20px;}ĐĐtop-buttons-
container div {Đ   display: flex;Đ   gap: 15px; /* Add space between
username, datetime, and status */}ĐĐlogout-button {Đ   padding: 10px 20px;Đ
background-color: #f44336;Đ   color: white;Đ   font-size: 16px;Đ   border:
none;Đ   cursor: pointer;Đ   border-radius: 5px;}ĐĐlogout-button:hover {Đ
background-color: #d32f2f;}ĐĐprint-button {Đ   background-color:yellow;Đ
color:black;Đ   border: none;Đ   cursor: pointer;}ĐĐprint-button:hover {Đ
background-color:white;Đ   color:black;}ĐĐwatermark {Đ   position: absolute;Đ
    bottom: 10px;Đ   right: 10px;Đ   font-size: 12px;Đ   color: rgba(206, 21,
21, 0.2);}ĐĐ       /* Styling for tables */Đ       table {Đ           width:
100%;Đ           margin-top: 20px;Đ           border-collapse: collapse;Đ
            background-color: #333; /* Dark background */Đ           color:
white; /* Bright text */Đ       }Đ       th, td {Đ           padding: 8px;Đ
            text-align: left;Đ           border: 1px solid #ddd;Đ       }Đ
        th {Đ           background-color: #444;Đ       }Đ   </style>Đ/head>Đ
<body>Đ   <div class="container">Đ       <!-- Sidebar -->Đ       <div
class="sidebar">Đ           <h2>RIAC 2.1.1</h2>Đ           <button
onclick="showContent('home')">Home</button>Đ           <button
onclick="showContent('configuration')">Configuration</button>Đ
<button onclick="showContent('data-log')">Data Log Analysis</button>Đ
            <button class="audit-trails-btn"
onclick="toggleSubMenu(this)">Audit Trails</button>Đ           <div
class="sub-buttons">Đ               <button onclick="showContent('alarm-
audit')">Alarm Audit Trail</button>Đ               <button
onclick="showContent('equipment-audit')">Equipment Audit Trail</button>Đ
                <button onclick="showContent('user-audit')">User Audit Trail</
button>Đ
```

```
                    <button onclick="showContent('Trend-audit')">Trend
Data Report</button>Đ               <button onclick="showContent('email-
audit')"> Email Audit Trail</button>Đ               <button
onclick="showContent('sms-audit')">SMS Audit Trail</button>Đ
<button onclick="showContent('review-approved')">Review and Approved</button>Đ
            </div>Đ           <button onclick="showContent('archive-
analysis')">Archive Analysis</button>Đ           <button
onclick="showContent('company-info')">Company Information</button>Đ       </
div>Đ       <!-- Main Content -->Đ       <div class="content">Đ
<!-- Top Buttons Container -->Đ           <div class="top-buttons-container">Đ
             Đ               <!-- Username, Date, and Status will appear here
-->Đ              <div>Đ                   <div class="Communication:-"
id="status">Communication:- Online</div>Đ                  <div
id="username"></div>Đ                   <div id="datetime"></div>Đ
                  Đ              </div>Đ               <button class="logout-
button" onclick="logout()">Logout</button>Đ   Đ           </div>Đ
<div id="display-content">Đ               <p></p>Đ           </div>Đ       </
div>Đ   </div>Đ   <script>Đ       // Function to update date and time
continuouslyĐ       function updateDatetime() {Đ           const now = new
Date();Đ           const datetimeStr = now.toLocaleString();Đ
document.getElementById("datetime").textContent = `DateTime: ${datetimeStr}`;Đ
        }Đ       // Function to handle logoutĐ       function logout() {Đ
            // Remove the username from localStorage and redirect to login
pageĐ           localStorage.removeItem("username");Đ
window.location.href = "index.html";Đ       }Đ       // Retrieve the username
from localStorage and display itĐ       window.onload = function() {Đ
            const username = localStorage.getItem("username");Đ           if
(username) {Đ               document.getElementById("username").textContent =
`User:- ${username}`;Đ           } else {Đ               window.location.href
= "index.html"; // If not logged in, redirect to login pageĐ           }Đ
            // Update the date and time every secondĐ
setInterval(updateDatetime, 1000);Đ           updateDatetime(); // Call once
initially to show the time immediatelyĐ       };Đ   </script>Đ   <script
src="fetchdata.js"></script>Đ   <script>Đ       function
toggleSubMenu(button) {Đ           const subMenu = button.nextElementSibling;Đ
            subMenu.style.display = subMenu.style.display === 'none' ||
subMenu.style.display === '' ? 'flex' : 'none';Đ       }Đ       function
showContent(contentId) {Đ   const displayContent =
document.getElementById('display-content');Đ   const content =
document.querySelector('.content');Đ   // Clear existing contentĐ
displayContent.innerHTML = '';Đ   // Reset background imageĐ
content.style.backgroundImage = '';Đ   // Hide the print button initiallyĐ
hidePrintButton();Đ   switch (contentId) {Đ       case 'Trend-audit':Đ
            // Display a container for the trend chartĐ
displayContent.innerHTML = `Đ               <h1>Trend Data Report</h1>Đ
                <div id="chart-container" style="height: 400px; margin-top:
20px;">Đ                   <canvas id="sensorChart"></canvas>Đ
</div>`;Đ           // Fetch and render sensor data in a graph formatĐ
            fetchAndRenderSensorData();Đ
showPrintButton('TrendData');Đ           break;Đ       case 'user-audit':Đ
            fetchTableData('LoginActivity1');Đ
displayContent.innerHTML = `<h1>USER AUDIT TRAIL</h1>`;Đ
showPrintButton('Audit');Đ           break;Đ       case 'equipment-audit':Đ
            fetchTableData('System_Alarms');Đ
displayContent.innerHTML = `<h1>EQUIPMENT AUDIT TRAIL</h1>`;Đ
showPrintButton('Equipment-Data');Đ           break;Đ       case 'alarm-
audit':Đ           fetchTableData('ALarmHistory1');Đ
displayContent.innerHTML = `<h1>ALARM AUDIT TRAIL</h1>`;Đ
showPrintButton('Alarm_History');Đ           break;Đ       case 'data-log':Đ
            fetchTableData('LoginLogout');Đ
```

```
displayContent.innerHTML = `<h1>SENSOR AUDIT TRAIL</h1>`;Đ
showPrintButton('Sensors');Đ           break;Đ      case 'company-info':Đ
displayContent.innerHTML = `Đ       <h1>Company Information</h1>Đ       <div
class="company-info">Đ          <p><strong>Company Name:</strong> Ran
Industrial Automation & Controls Private Limited</p>Đ
<p><strong>Address:</strong> Plot No. 25, Road No. 5, Aleap Industrial
Estate, Kukatpally,<br>Đ           Near Pragathi Nagar, Near Aleap Industrial
Estate,<br>Đ           Gajularamaram, Hyderabad-500090, Telangana, India</p>Đ
            <p><strong>Contact Person:</strong> G. Ranjith Kumar (Managing
Director)</p>Đ       </div>Đ   `;Đ   break;Đ       case 'home':Đ           //
displayContent.innerHTML = `<h1>HOME</h1>`;Đ           //
content.style.backgroundImage = "url('Home.png')";Đ          //
content.style.backgroundSize = "cover";Đ         //
content.style.backgroundPosition = "center";Đ          break;Đ       default:Đ
            displayContent.innerHTML = `<h1>${contentId.replace(/-/g, '
').toUpperCase()}</h1>`;Đ   }ĐĐfunction fetchAndRenderSensorData() {Đ   const
now = new Date();Đ   Đ   // Define the start of the day (yesterday) and end
of the day (today)Đ   const startOfDay = new Date(now);Đ
startOfDay.setHours(0, 0, 0, 0);Đ   startOfDay.setDate(now.getDate() - 1); //
Yesterday's midnightĐ   const endOfDay = new Date(now);Đ
endOfDay.setHours(23, 59, 59, 999); // Today's end of the dayĐ   // Format
dates to ISO stringsĐ   const startDate = startOfDay.toISOString();Đ   const
endDate = endOfDay.toISOString();Đ   // Fetch data from the API for the
calculated date rangeĐ   fetch(`http://localhost:5000/api/getData?
tableName=DataOfSensors&startDate=${startDate}&endDate=${endDate}`)Đ
        .then(response => response.json())Đ       .then(data => {Đ
if (data.success && data.data.length > 0) {Đ              // Parse the data
from the API responseĐ              const parsedData = data.data.map(item =>
({Đ                  x: item.DateTime, // Use DateTime exactly as-is from
the databaseĐ                  RTS1: item.RTS1,Đ                  RTS2:
item.RTS2,Đ                  RTS3: item.RTS3,Đ                  RTS4:
item.RTS4,Đ                  RTS5: item.RTS5,Đ                  HSTS:
item.HSTS,Đ                  CSTS: item.CSTS,Đ              }));Đ
                // Prepare datasets for the chartĐ              const
datasets = [Đ                  {Đ                      label: 'RTS1',Đ
                        data: parsedData.map(item => ({ x: item.x, y:
item.RTS1 })),Đ                      borderColor: '#FF5733',Đ
                        backgroundColor: 'rgba(255, 87, 51, 0.2)',Đ
                        fill: true,Đ                  },Đ                  {Đ
                        label: 'RTS2',Đ                      data:
parsedData.map(item => ({ x: item.x, y: item.RTS2 })),Đ
borderColor: '#33FF57',Đ                      backgroundColor: 'rgba(51,
255, 87, 0.2)',Đ                      fill: true,Đ                  },Đ
                    {Đ                      label: 'RTS3',Đ
                        data: parsedData.map(item => ({ x: item.x, y:
item.RTS3 })),Đ                      borderColor: '#3357FF',Đ
                        backgroundColor: 'rgba(51, 87, 255, 0.2)',Đ
                        fill: true,Đ                  },Đ                  {Đ
                        label: 'RTS4',Đ                      data:
parsedData.map(item => ({ x: item.x, y: item.RTS4 })),Đ
borderColor: '#FF33FF',Đ                      backgroundColor: 'rgba(255,
51, 255, 0.2)',Đ                      fill: true,Đ                  },Đ
                    {Đ                      label: 'RTS5',Đ
                        data: parsedData.map(item => ({ x: item.x, y:
item.RTS5 })),Đ                      borderColor: '#FFD700',Đ
                        backgroundColor: 'rgba(255, 215, 0, 0.2)',Đ
                        fill: true,Đ                  },Đ                  {Đ
                        label: 'HSTS',Đ                      data:
parsedData.map(item => ({ x: item.x, y: item.HSTS })),Đ
```

```
borderColor: '#00FF7F',Ð                       backgroundColor: 'rgba(0, 255, 
127, 0.2)',Ð                       fill: true,Ð                   },Ð
                    {Ð                         label: 'CSTS',Ð
                        data: parsedData.map(item => ({ x: item.x, y: 
item.CSTS })),Ð                       borderColor: '#8A2BE2',Ð
                        backgroundColor: 'rgba(138, 43, 226, 0.2)',Ð
                        fill: true,Ð                  },Ð               ];Ð
                // Render the chartÐ               renderChart(datasets);Ð
            } else {Ð              console.error("No data available or 
failed to fetch data");Ð          }Ð       })Ð      .catch(error => 
console.error('Error fetching data:', error));Ð}Ð// Schedule the data update to 
run at midnight every dayÐfunction scheduleMidnightUpdate() {Ð   const now = 
new Date();Ð   const nextMidnight = new Date(now);Ð
nextMidnight.setHours(24, 0, 0, 0); // Set to midnight of the next dayÐ
const timeUntilMidnight = nextMidnight - now;Ð   setTimeout(() => {Ð
fetchAndRenderSensorData(); // Refresh data at midnightÐ
scheduleMidnightUpdate(); // Reschedule for the next dayÐ   }, 
timeUntilMidnight);Ð}ÐscheduleMidnightUpdate(); // Start the midnight update 
processÐ// Chart rendering functionÐfunction adjustTimeZone(datasets) {Ð   // 
If your dataset is in UTC and you want to apply an offset (e.g., UTC +5:30)Ð
    const timeZoneOffset = 5.5 * 60 * 60 * 1000; // 5.5 hours in milliseconds 
(for IST)Ð   Ð   // Adjust dataset times based on offsetÐ
datasets.forEach(dataset => {Ð       dataset.data = 
dataset.data.map(dataPoint => {Ð           // Assuming dataPoint.x is the 
timestamp, adjust by adding the offsetÐ          dataPoint.x = new 
Date(dataPoint.x).getTime() - timeZoneOffset;Ð           return dataPoint;Ð
        });Ð   });Ð   Ð   return datasets;Ð}Ðfunction renderChart(datasets) {Ð
    datasets = adjustTimeZone(datasets);Ð   Ð   const ctx = 
document.getElementById('sensorChart').getContext('2d');Ð   Ð   new 
Chart(ctx, {Ð       type: 'line', // Line chart to represent the trend dataÐ
        data: {Ð           datasets: datasetsÐ       },Ð       options: {Ð
            responsive: true,Ð           maintainAspectRatio: false, // Allow 
resizing without keeping the aspect ratioÐ          backgroundColor: 
'#ffffff', // Set chart background color to whiteÐ          scales: {Ð
                x: {Ð                  type: 'time', // Time-based X-axisÐ
                    time: {Ð                   unit: 
datasets[0].data.length > 1440 ? 'hour' : 'minute', // Hour if >1440 points 
(1 per min for 24 hrs), otherwise minuteÐ
tooltipFormat: 'dd MM yyyy HH:mm:ss', // Tooltip format for date and timeÐ
                        displayFormats: {Ð                           hour: 
'dd MM yyyy HH:mm:ss', // Hour format for timelineÐ
minute: 'dd MM yyyy HH:mm:ss', // Minute with seconds for detailed viewÐ
                        },Ð                    },Ð                 title: {Ð
                        display: true,Ð                     text: 'Time 
(Hours or Minutes)', // Title for X-axisÐ                      color: 
'#000000', Ð                 },Ð                grid: {Ð
                        color: '#000000', // Set grid lines color to blackÐ
                    },Ð                   ticks: {Ð
autoSkip: true, // Automatically skip labels to avoid clutterÐ
                        maxRotation: 45, // Rotate tick labels for better fitÐ
                        minRotation: 30, // Minimum rotation angleÐ
                        source: 'auto', // Adjust spacing automaticallyÐ
                    }Ð               },Ð              y: {Ð
                    title: {Ð                      display: true,Ð
                        text: 'Temperature(°C)', Ð
color: '#000000' // Title for Y-axisÐ                   },Ð
beginAtZero: true, // Always start from zero for consistencyÐ
                    ticks: {Ð                      autoSkip: true, // Skip 
ticks for clarityÐ
```

```
                                },Đ                   grid: {Đ
                        color: '#000000', // Set grid lines color to blackĐ
                    }Đ                }Đ            },Đ           plugins: {Đ
                legend: {Đ                   display: true, // Display the
legend for datasetsĐ              }Đ           },Đ          elements: {Đ
                line: {Đ                  tension: 0.1, // Smoothen lines
slightlyĐ              },Đ                point: {Đ                   radius:
datasets[0].data.length > 1440 ? 0 : 3, // Hide points for large datasetsĐ
                    hoverRadius: 5, // Point radius on hoverĐ
                    hitRadius: 10, // Increase hit radius for easier
interactionĐ              }Đ           }Đ        }Đ   });ĐĐfunction
showPrintButton(contentId) {Đ   // Hide the previous print button if anyĐ
hidePrintButton();Đ   // Create and show the new print buttonĐ   const
printButton = document.createElement('button');Đ
printButton.classList.add('print-button');Đ   printButton.textContent =
`Print ${contentId} Data`;Đ   Đ   // Attach the correct functionality to the
print buttonĐ   if (contentId === 'TrendData') {Đ       printButton.onclick =
() => printChart(); // Print chart logicĐ   } else {Đ
printButton.onclick = () => generatePDF(contentId); // Standard print logic
for tablesĐ   }Đ   document.querySelector('.top-buttons-
container').appendChild(printButton);ĐĐfunction hidePrintButton() {Đ   const
existingPrintButton = document.querySelector('.print-button');Đ   if
(existingPrintButton) {Đ       existingPrintButton.remove();Đ   }ĐĐfunction
printChart() {Đ   const canvas = document.getElementById('sensorChart'); //
Get your chart canvasĐ   if (!canvas) {Đ       alert("Chart canvas not
found!");Đ       return;Đ   }Đ   // Use html2canvas to capture the chart
contentĐ   html2canvas(canvas)Đ       .then((canvasImage) => {Đ
const imgData = canvasImage.toDataURL('image/png'); // Convert canvas to
image data URLĐ           const { jsPDF } = window.jspdf; // Get jsPDF
instanceĐ           const doc = new jsPDF('landscape'); // Create a new PDF
in landscape modeĐ           // Add the chart image to the PDF documentĐ
            doc.addImage(imgData, 'PNG', 10, 10, 180, 160); // Position the
image on the PDFĐ           // Save the PDFĐ
doc.save('TrendDataChart.pdf'); // Save the chart as a PDF fileĐ       })Đ
        .catch((error) => {Đ           console.error("Error generating chart
PDF:", error);Đ           alert("Failed to generate the chart PDF. Please try
again.");Đ       });ĐĐfunction generatePDF(contentId, reportType) {Đ   const
{ jsPDF } = window.jspdf;Đ   const doc = new jsPDF('landscape');Đ   // Page
dimensionsĐ   const pageWidth = doc.internal.pageSize.getWidth();Đ   const
pageHeight = doc.internal.pageSize.getHeight();Đ   // Fetch username and
current date-timeĐ   const username = localStorage.getItem("username") ||
"Unknown User";Đ   const currentDateTime = new Date().toLocaleString();Đ   //
Common Header SectionĐ   function addHeader() {Đ
doc.setFont('Helvetica', 'bold');Đ       doc.setFontSize(12);Đ       const
logoWidth = 20;Đ       const logoHeight = 20;Đ       // Left Image (Logo)Đ
        const leftImageBase64 = ''; // Add base64 or path to your logo imageĐ
        if (leftImageBase64) doc.addImage(leftImageBase64, 'PNG', 10, 10,
logoWidth, logoHeight);Đ       // Center TitleĐ       doc.text('Ran
Industrial Automation Pvt Ltd.', pageWidth / 2, 15, { align: 'center' });Đ
        doc.setFontSize(10);Đ       doc.text('Location: Block 1 | EQUIPMENT
ID: PN-PSGRT-001', pageWidth / 2, 22, { align: 'center' });Đ
doc.text(`Equipment Name: Ante Room | Report ID: ${contentId}`, pageWidth /
2, 29, { align: 'center' });Đ       // Right Image (Logo)Đ       const
rightImageBase64 = ''; // Add base64 or path to your right logo imageĐ
if (rightImageBase64) doc.addImage(rightImageBase64, 'PNG', pageWidth - 30,
10, logoWidth, logoHeight);Đ       // Separator lineĐ
doc.setLineWidth(0.5);Đ       doc.line(10, 35, pageWidth - 10, 35);Đ   }Đ
    // Common Footer SectionĐ   function addFooter(pageNumber, totalPages) {Đ
        doc.setFont('Helvetica', 'normal');Đ       doc.setFontSize(10);Đ
        const footerText = [Đ
```

```
                                        `Printed Date & Time:
${currentDateTime}`,Đ            `Printed By: ${username}`,Đ            `Page
${pageNumber} of ${totalPages}`,Đ       ];Đ       // Align footer contentĐ
        footerText.forEach((text, i) => {Đ           doc.text(text, 15,
pageHeight - (20 - i * 5));Đ       });Đ       // Separator lineĐ
doc.setLineWidth(0.5);Đ       doc.line(10, pageHeight - 25, pageWidth - 10,
pageHeight - 25);Đ   }Đ   // Fetch Table Data DynamicallyĐ   const
tableContent = document.querySelector("#display-content");Đ   const tableData
= [];Đ   tableContent.querySelectorAll("table tr").forEach(row => {Đ
const cols = row.querySelectorAll("td, th");Đ
tableData.push(Array.from(cols).map(col => col.textContent.trim()));Đ   });Đ
    // Add Table ContentĐ   doc.autoTable({Đ       head: [tableData[0]], //
First row as headerĐ       body: tableData.slice(1), // Remaining rows as
table contentĐ       startY: 40, // Start below headerĐ       theme:
'grid', // Uniform grid themeĐ       styles: {Đ           font: 'Helvetica',Đ
            fontSize: 10,Đ           cellPadding: 3,Đ           lineColor:
[0, 0, 0],Đ       },Đ       columnStyles: {Đ           0: { cellWidth: 30 },Đ
            1: { cellWidth: 50 },Đ           2: { cellWidth: 60 },Đ       },Đ
        didDrawPage: (data) => {Đ           const pageNumber =
data.pageNumber;Đ           const totalPages =
doc.internal.getNumberOfPages();Đ           // Add header and footer on each
pageĐ           addHeader();Đ           addFooter(pageNumber, totalPages);Đ
        }Đ   });Đ   // Add Notes and Signatures on the Last PageĐ   const
totalPages = doc.internal.getNumberOfPages();Đ   doc.setPage(totalPages);Đ
const lastY = doc.lastAutoTable.finalY || 50;Đ   doc.setFont('Helvetica',
'normal');Đ   doc.setFontSize(12);Đ   doc.text('Checked By:
____________________', 15, lastY + 30);Đ   doc.text('Reviewed By:
____________________', pageWidth / 2, lastY + 30);Đ   // Save the PDFĐ
doc.save(`${contentId}-${reportType}-Report.pdf`);}ĐĐ       function logout() {Đ
            window.location.href = "index.html";Đ       }Đ   </script>Đscript
src="https://cdn.jsdelivr.net/npm/chart.js"></script>Đscript src="https://
cdn.jsdelivr.net/npm/chartjs-adapter-date-fns@latest"></script>Đscript
src="https://cdnjs.cloudflare.com/ajax/libs/html2canvas/1.4.1/
html2canvas.min.js"></script>Đscript src="https://cdnjs.cloudflare.com/ajax/
libs/jspdf/2.5.1/jspdf.umd.min.js"></script>Đscript src="https://
cdnjs.cloudflare.com/ajax/libs/jspdf-autotable/3.5.21/
jspdf.plugin.autotable.min.js"></script>Đ      Đ/body>Đ/html>Đ
```

vscode-chartjs\templates\chart.preview.html

```
<!DOCTYPE html>Đhtml lang="en">Đ <head>Đ   <meta charset="utf-8">Đ   <meta
http-equiv="X-UA-Compatible" content="IE=edge">Đ   <meta http-equiv="Content-
Security-Policy" Đ     content="default-src * vscode-resource: https: 'unsafe-
inline' 'unsafe-eval';Đ       script-src vscode-resource: https: 'unsafe-
inline' 'unsafe-eval';Đ       style-src vscode-resource: https: 'unsafe-
inline';Đ       img-src vscode-resource: https:;Đ       connect-src vscode-
resource: https: http:;">Đ   <meta name="viewport" content="width=device-
width, initial-scale=1.0">Đ   <meta name="description" content="Chart.js
Preview">Đ   <base href="https://www.chartjs.org/samples/latest/"
target="_blank" />Đ   <title>Preview</title>Đ   <script src="{scripts}/
Chart.min.js"></script>Đ   <style>Đ     body {Đ       background:#fff;Đ
color: #333;Đ       margin: 0px;Đ       padding: 0px;Đ     }Đ     #message {Đ
        font-family: 'Lucida Sans', 'Lucida Sans Regular', 'Lucida Grande',
'Lucida Sans Unicode', Geneva, Verdana, sans-serif;Đ       color: red;Đ
font-size: 11pt;Đ       text-align: center;Đ       padding-top: 10px;Đ     }Đ
      #chart-canvas {Đ       padding: 5px;Đ     }Đ   </style>Đ </head>Đ <body>Đ
    <div id="message">Loading Chart Preview...</div>Đ   <canvas id="chart-
canvas" width="400" height="400"></canvas>Đ   <script type="text/javascript">Đ
      let vscode, message, chartContext, chartConfig, chart;Đ
document.addEventListener('DOMContentLoaded', event => {Đ       // initialize
page elementsĐ       message = document.getElementById('message');Đ
chartContext = document.getElementById('chart-
canvas').getContext('2d');       Đ       try {Đ         // notify webviewĐ
          vscode = acquireVsCodeApi();Đ         vscode.postMessage({command:
'refresh'});Đ       }Đ       catch (error) {Đ         // ignore: must be
loaded outside of vscode webviewĐ       }Đ     });Đ     // vega spec update
handlerĐ     window.addEventListener('message', event => {Đ       switch
(event.data.command) {Đ         case 'showMessage':Đ
showMessage(event.data.message);Đ           break;Đ         case 'refresh':Đ
            try {Đ             vscode.setState({uri: event.data.uri});Đ
              chartConfig = event.data.config;Đ             chart =
preview(chartConfig);Đ           }Đ           catch (error) {Đ
console.error('chart.preview:', error.message);Đ
showMessage(error.message);Đ           }Đ           break;Đ       }Đ     });Đ
      // chart preview updateĐ     function preview(chartConfig) {Đ
showMessage(''); // 'Loading chart preview...';Đ       try {Đ         chart =
new Chart(chartContext, chartConfig);Đ       }Đ       catch (error) {Đ
          console.error('chart.preview:', error.message);Đ
showMessage(error.message);Đ         chart = null;Đ       };Đ       return
chart;Đ     }Đ     function showHelp() {Đ       vscode.postMessage({command:
'showHelp'});Đ     }Đ     function showMessage(text) {Đ
message.innerText = text;Đ     }Đ   </script>Đ </body>Đ/html>
```

angular\src\styles.css

```
/* You can add global styles to this file, and also import other style files
*/
```

.vercel\project.json

```
{"projectId":"prj_hFk4HtL8Q6O8qbAf6K9EaMUCKmgJ","orgId":"team_SDvvM34rN9qWCmEY
zN1jAqn1"}
```

.vscode\launch.json

```
{Đ   // Use IntelliSense to learn about possible attributes.Đ   // For more
information, visit: https://go.microsoft.com/fwlink/?linkid=830387Đ
"version": "0.2.0",Đ   "configurations": [Đ Đ       {Đ           "name":
"Open index.html in Edge",Đ           "request": "launch",Đ           "type":
"msedge",Đ           "file": "E:\\Html\\Html\\webapp\\index.html",Đ
"webRoot": "${workspaceFolder}"Đ       },Đ      Đ   ]}ĐĐ
```

angular\angular.json

```
{  "$schema": "./node_modules/@angular/cli/lib/config/schema.json",
"version": 1,  "newProjectRoot": "projects",  "projects": {    "angular-
test": {      "projectType": "application",      "schematics": {},
"root": "",      "sourceRoot": "src",      "prefix": "app",      "architect":
{        "build": {          "builder": "@angular-devkit/build-
angular:browser",          "options": {            "outputPath": "dist/
angular-test",            "index": "src/index.html",            "main": "src/
main.ts",            "polyfills": ["zone.js"],            "tsConfig":
"tsconfig.app.json",            "assets": ["src/favicon.ico", "src/assets"],
            "styles": ["src/styles.css"],            "scripts": []          },
          "configurations": {            "production": {
"budgets": [                {                  "type": "initial",
                  "maximumWarning": "500kb",                  "maximumError":
"1mb"                },                {                  "type":
"anyComponentStyle",                  "maximumWarning": "2kb",
                  "maximumError": "4kb"                }              ],
              "outputHashing": "all"            },            "development": {
              "buildOptimizer": false,              "optimization": false,
              "vendorChunk": true,              "extractLicenses": false,
              "sourceMap": true,              "namedChunks": true            }
          },          "defaultConfiguration": "production"        },
"serve": {          "builder": "@angular-devkit/build-angular:dev-server",
          "configurations": {            "production": {
"browserTarget": "angular-test:build:production"            },
"development": {              "browserTarget": "angular-
test:build:development"            }          },
"defaultConfiguration": "development"        },        "extract-i18n": {
          "builder": "@angular-devkit/build-angular:extract-i18n",
"options": {            "browserTarget": "angular-test:build"          }
        },        "test": {          "builder": "@angular-devkit/build-
angular:karma",          "options": {            "polyfills": ["zone.js",
"zone.js/testing"],            "tsConfig": "tsconfig.spec.json",
"assets": ["src/favicon.ico", "src/assets"],            "styles": ["src/
styles.css"],            "scripts": []          }        }      }    }  }}
```

angular\package.json

```
{  "private": true,  "scripts": {    "ng": "ng",    "start": "ng serve",
"build": "ng build",    "watch": "ng build --watch --configuration
development",    "test": "ng test"  },  "dependencies": {    "@angular/
animations": "^15.0.0",    "@angular/common": "^15.0.0",    "@angular/
compiler": "^15.0.0",    "@angular/core": "^15.0.0",    "@angular/forms":
"^15.0.0",    "@angular/platform-browser": "^15.0.0",    "@angular/platform-
browser-dynamic": "^15.0.0",    "@angular/router": "^15.0.0",    "rxjs":
"~7.5.0",    "tslib": "^2.3.0",    "zone.js": "~0.12.0"  },
"devDependencies": {    "@angular-devkit/build-angular": "^15.0.0",
"@angular/cli": "~15.0.0",    "@angular/compiler-cli": "^15.0.0",    "@types/
jasmine": "~4.3.0",    "jasmine-core": "~4.5.0",    "karma": "~6.4.0",
"karma-chrome-launcher": "~3.1.0",    "karma-coverage": "~2.2.0",    "karma-
jasmine": "~5.1.0",    "karma-jasmine-html-reporter": "~2.0.0",
"typescript": "~4.8.2"  }}
```

angular\tsconfig.app.json

```
/* To learn more about this file see: https://angular.io/config/tsconfig. */{
  "extends": "./tsconfig.json",  "compilerOptions": {    "outDir": "./out-tsc/
app",    "types": []  },  "files": ["src/main.ts"],  "include": ["src/**/
*.d.ts"]}
```

angular\tsconfig.json

```
/* To learn more about this file see: https://angular.io/config/tsconfig. */{
  "compileOnSave": false,  "compilerOptions": {    "baseUrl": "./",
"outDir": "./dist/out-tsc",    "forceConsistentCasingInFileNames": true,
"strict": true,    "noImplicitOverride": true,
"noPropertyAccessFromIndexSignature": true,    "noImplicitReturns": true,
"noFallthroughCasesInSwitch": true,    "sourceMap": true,    "declaration":
false,    "downlevelIteration": true,    "experimentalDecorators": true,
"moduleResolution": "node",    "importHelpers": true,    "target": "ES2022",
    "module": "ES2022",    "useDefineForClassFields": false,    "lib":
["ES2022", "dom"]  },  "angularCompilerOptions": {
"enableI18nLegacyMessageIdFormat": false,    "strictInjectionParameters":
true,    "strictInputAccessModifiers": true,    "strictTemplates": true  }}
```

angular\tsconfig.spec.json

```
/* To learn more about this file see: https://angular.io/config/tsconfig. */{
  "extends": "./tsconfig.json",  "compilerOptions": {    "outDir": "./out-tsc/
spec",    "types": ["jasmine"]  },  "include": ["src/**/*.spec.ts", "src/**/
*.d.ts"]}
```

application.json

```
{Ð   "ConnectionStrings": {Ð     "DefaultConnection":
"Server=localhost;Database=UseLoginDB;Trusted_Connection=True;"Ð   },Ð
"Logging": {Ð     "LogLevel": {Ð       "Default": "Information",Ð
"Microsoft": "Warning",Ð       "Microsoft.Hosting.Lifetime": "Information"Ð
      }Ð   },Ð   "AllowedHosts": "*"Ð }Ð
```

launch.json

```
{Đ   "version": "0.2.0",Đ   "configurations": [Đ       {Đ           "name":
"Launch Edge with index.html",Đ           "request": "launch",Đ
"type": "msedge",Đ           "file": "E:/Html/Html/webapp/index.html",Đ
            "webRoot": "E:/Html/Html/webapp"Đ       },Đ       {Đ
"name": "Launch Chrome with index.html",Đ           "request": "launch",Đ
            "type": "chrome",Đ           "file": "E:/Html/Html/webapp/
index.html",Đ           "webRoot": "E:/Html/Html/webapp"Đ       }Đ       Đ   ]Đ
}Đ
```

package.json

```
{  "dependencies": {    "axios": "^1.7.7",    "bcryptjs": "^2.4.3",    "body-
parser": "^1.20.3",    "chartjs-adapter-date-fns": "^3.0.0",    "chartjs-
adapter-luxon": "^1.3.1",    "chartjs-adapter-moment": "^1.0.1",    "cors":
"^2.8.5",    "dayjs": "^1.11.13",    "dotenv": "^16.4.5",    "express":
"^4.21.1",    "jspdf": "^2.5.2",    "jspdf-autotable": "^3.8.4",    "luxon":
"^3.5.0",    "moment": "^2.30.1",    "mssql": "^11.0.1"  }}
```

vercel.json

```
{Ð "version": 2,Ð "builds": [Ð   {Ð     "src": "server.js",Ð     "use":
"@vercel/node"Ð   }Ð ],Ð "routes": [Ð   {Ð     "src": "/(.*)",Ð     "dest": "/
server.js"Ð   }Ð ]Ð}Ð
```

vscode-chartjs\.vscode\extensions.json

```
{Ð̃òò 6VR ‡GG ¢òövòæÖ–7&÷6ögBæ6öÒögvÆ–æ²óôÆ–æ´–CÓƒ#sƒCḿòò f÷" F†R Fö7VÖVçF F–öâ
about the extensions.json formatÐ̃'&V6öÖÖVæF F–öç2#¢ ½"ms-vscode.vscode-
typescript-tslint-plugin"Ð̃Ý
```

vscode-chartjs\.vscode\launch.json

```
// A launch configuration that compiles the extension and then opens it
inside a new windowÐ/ Use IntelliSense to learn about possible attributes.Ð/
Hover to view descriptions of existing attributes.Ð/ For more information,
visit: https://go.microsoft.com/fwlink/?linkid=830387Ð{Ð'fW'6–öâ#¢ # ã"ã "Í
’&6öæf–wW& F–öç2#¢ ½™™{Ð&æ ÖR#¢ %'Vâ W‡FVç6–öâ™Í'G— R#¢ &W‡FVç6–öä†÷7B"Í
™’''&W VW7B#¢ &Æ Væ6,™Í''VçF–ÖTW†V7WF &ÆR#¢ "G¶W†V5  F‡Ò™Í& &w2#¢™½"--
extensionDevelopmentPath=${workspaceFolder}"Ð™•Ò™Í&÷WDf–ÆW2#¢ ½
™™"${workspaceFolder}/out/**/*.js™"•ÐÍ' &TÆ Væ6…F 6²#¢ &ç Ó¢ v F6™,}™Ð{Ð&æ ÖR#¢
"Extension Tests",Ð™’'G— R#¢ &W‡FVç6–öä†÷7B™Í'&W VW7B#¢ &Æ Væ6,"Í
™’''VçF–ÖTW†V7WF &ÆR#¢ "G¶W†V5  F‡Ò™Í& &w2#¢™½"--
extensionDevelopmentPath=${workspaceFolder}",Ð™™"--
extensionTestsPath=${workspaceFolder}/out/test/suite/index"Ð™•Ò™Í&÷WDf–ÆW2#¢ ½
™™"${workspaceFolder}/out/test/**/*.js™"•ÐÍ' &TÆ Væ6…F 6²#¢ &ç Ó¢ v F6™,}•ÐÐ
```

vscode-chartjs\.vscode\settings.json

```
// Place your settings in this file to overwrite default and user settings.Ð{Ð
    "files.exclude": {Ð       "out": false // set this to true to hide the
"out" folder with the compiled JS filesÐ   },Ð   "search.exclude": {Ð
"out": true // set this to false to include "out" folder in search resultsÐ
    },Ð   // Turn off tsc task auto detection since we have the necessary
tasks as npm scriptsÐ   "typescript.tsc.autoDetect": "off"Ð}
```

vscode-chartjs\.vscode\tasks.json

```
// See https://go.microsoft.com/fwlink/?LinkId=733558Ð/ for the documentation
about the tasks.json formatÐÐ'fW'6–öâ#¢ #"ã ã "Í'F 6·2#¢ ½{Ð'G— R#¢ &ç Ò"Í
™’'67&— B#¢ 'v F6,™Í' &ö&ÆVÖÖ F6†W"#¢ "GG62×v F6,™Í&—4& 6¶w&÷VæB#¢ G'VRÍ
™’' &W6VçF F–öâ#¢™½"reveal": "never"—ÐÍ&w&÷W #¢™½"kind": "build"™,ÐisDefault":
trueÐ—ÿÐÐ
```

vscode-chartjs\package.json

```
{Đ "name": "vscode-chartjs",Đ "displayName": "Chart.js Preview",Đ
"description": "Chart.js Preview Ø=ÜÊ extension"vĐrsion": "1.3.0",Đ "icon":
"images/chart.png",Đ "publisher": "RandomFractalsInc",Đ "author": "Taras
Novak a.k.a. dataPixy devTools maker :)",Đ "contributors": [Đ   "Taras Novak"Đ
  ],Đ "license": "Apache-2.0",Đ "readme": "README.md",Đ "repository": {Đ
"type": "git",Đ   "url": "https://github.com/RandomFractals/vscode-chartjs"Đ
  },Đ "bugs": "https://github.com/RandomFractals/vscode-chartjs/issues",Đ
"homepage": "https://github.com/RandomFractals/vscode-chartjs/README.md",Đ
"keywords": [Đ   "chart.js",Đ   "chart",Đ   "viewer",Đ   "preview",Đ
"charts",Đ   "dataViz",Đ   "dev tools",Đ   "chart samples",Đ   "json5 chart
config",Đ   "chart examples"Đ ],Đ "galleryBanner": {Đ   "color": "#333",Đ
"theme": "dark"Đ },Đ "engines": {Đ   "vscode": "^1.39.0"Đ },Đ "categories": [Đ
    "Programming Languages"Đ ],Đ "activationEvents": [Đ
"onWebviewPanel:chart.preview",Đ   "onWebviewPanel:chart.samples",Đ
"onLanguage:json5"Đ ],Đ "main": "./out/extension.js",Đ "contributes": {Đ
"languages": [Đ     {Đ       "id": "json5",Đ       "extensions": [Đ
".json5"Đ       ],Đ       "aliases": [Đ         "JSON5"Đ       ]Đ     }Đ   ],Đ
    "commands": [Đ     {Đ       "command": "chart.preview",Đ       "title":
"Preview Chart",Đ       "category": "Chart",Đ       "icon": {Đ
"light": "./images/chart.svg",Đ         "dark": "./images/chart.svg"Đ       }Đ
      },Đ     {Đ       "command": "chart.samples",Đ       "title": "Chart
Samples",Đ       "category": "Chart"Đ     }Đ   ],Đ   "keybindings": [Đ     {Đ
        "command": "chart.preview",Đ       "key": "shift+alt+c"Đ     },Đ     {Đ
        "command": "chart.samples",Đ       "key": "alt+s"Đ     }Đ   ],Đ
"menus": {Đ     "explorer/context": [Đ       {Đ         "command":
"chart.preview",Đ         "when": "resourceFilename =~ /.*\\.(chart.json5)/",Đ
          "group": "navigation"Đ       }Đ     ],Đ     "editor/title": [Đ
        {Đ         "command": "chart.preview",Đ         "when":
"resourceFilename =~ /.*\\.(chart.json5)/",Đ         "group": "navigation"Đ
        }Đ     ],Đ     "editor/title/context": [Đ       {Đ         "command":
"chart.preview",Đ         "when": "resourceFilename =~ /.*\\.(chart.json5)/",Đ
          "group": "navigation"Đ       }Đ     ]Đ   }Đ },Đ "scripts": {Đ
"vscode:prepublish": "npm run compile",Đ   "compile": "tsc -p ./",Đ
"watch": "tsc -watch -p ./",Đ   "pretest": "npm run compile",Đ   "test":
"node ./out/test/runTest.js"Đ },Đ "devDependencies": {Đ   "@types/json5":
"0.0.30",Đ   "@types/glob": "^7.1.1",Đ   "@types/mocha": "^5.2.7",Đ   "@types/
node": "^12.12.7",Đ   "@types/vscode": "^1.39.0",Đ   "glob": "^7.1.6",Đ
"mocha": "^6.2.2",Đ   "typescript": "^3.7.2",Đ   "tslint": "^5.20.1",Đ
"vscode-test": "^1.2.3"Đ },Đ "dependencies": {Đ   "chart.js": "^2.9.2",Đ
"json5": "^2.1.1"Đ }ĐĐ
```

vscode-chartjs\tsconfig.json

```
{Ð&6ö× –ÆW$÷ F–öç2#¢ ½"types": ["node"],™Ðmodule": "commonjs",™Ðtarget": "es6",Ð
™"outDir": "out",™Ðlib": [™Ð&W3b],™ÐsourceMap": true,™ÐrootDir": "src",™Ðstrict":
true   /* enable all strict type-checking options */Ð/* Additional Checks */™Ð/
"noImplicitReturns": true, /* Report error when not all code paths in
function return a value. */Ð// "noFallthroughCasesInSwitch": true, /* Report
errors for fallthrough cases in switch statement. */Ð// "noUnusedParameters":
true,  /* Report errors on unused parameters. */ÐÒÍ&W†6ÇVFR#¢ ½"node_modules",Ð
™".vscode-test"ÑÐ
```

vscode-chartjs\tslint.json

```
{Ð''VÆW2#¢ ™½"no-string-throw": true,™Ðno-unused-expression": true,™Ðno-duplicate-
variable": true,Ð™"curly": true,™Ðclass-name": true,™Ðsemicolon": ™[ÐG'VR™Í& Çv —2™],Ð
™"triple-equals": true—ÐÍ&FVf VÇE6WfW&—G’#¢ 'v &æ–ær}Ð
```

Table of Contents

fetchData.js
..... [object Object]

login.js
..... [object Object]

server.js
..... [object Object]

vscode-chartjs\out\chart.preview.js
..... [object Object]

vscode-chartjs\out\config.js
..... [object Object]

vscode-chartjs\out\extension.js
..... [object Object]

vscode-chartjs\out\logger.js
..... [object Object]

vscode-chartjs\out\preview.manager.js
..... [object Object]

vscode-chartjs\out\template.manager.js
..... [object Object]

vscode-chartjs\out\test\runTest.js
..... [object Object]

vscode-chartjs\out\test\suite\extension.test.js
..... [object Object]

vscode-chartjs\out\test\suite\index.js
..... [object Object]

vscode-chartjs\out\vega.preview.js
..... [object Object]

angular\src\index.html
..... [object Object]

index.html
..... [object Object]

sucess.html
..... [object Object]

vscode-chartjs\templates\chart.preview.html
..... [object Object]

angular\src\styles.css
..... [object Object]

.vercel\project.json
..... [object Object]

.vscode\launch.json
..... [object Object]

angular\angular.json
..... [object Object]

angular\package.json
..... [object Object]

angular\tsconfig.app.json
..... [object Object]

angular\tsconfig.json
..... [object Object]

angular\tsconfig.spec.json
..... [object Object]

application.json
..... [object Object]

launch.json
..... [object Object]

package.json
..... [object Object]

vercel.json
..... [object Object]

vscode-chartjs\.vscode\extensions.json
..... [object Object]

vscode-chartjs\.vscode\launch.json
..... [object Object]

vscode-chartjs\.vscode\settings.json
..... [object Object]

vscode-chartjs\.vscode\tasks.json
..... [object Object]

vscode-chartjs\package.json
..... [object Object]

vscode-chartjs\tsconfig.json
..... [object Object]

vscode-chartjs\tslint.json
..... [object Object]