

Power BI Embedding with SharePoint Framework Web Parts



Code and Slides for this Webinar

<https://github.com/CriticalPathTraining/PBIESPFX>

The screenshot shows the GitHub repository page for **CriticalPathTraining / PBIESPFX**. The repository description is "Demo of Power BI Embedding with SharePoint Framework Web Parts". The page includes navigation tabs for Code, Issues (0), Pull requests (0), Projects (0), Wiki, Insights, and Settings. On the right, there are buttons for Unwatch (3), Star (0), and Fork (0). Below the repository name, there are buttons for "Branch: master", "New pull request", "Create new file", "Upload files", "Find file", and a green "Clone or download" button. A commit by TedPattison is shown with the message "updates" and the latest commit hash "2f01551" from an hour ago. The file list on the left includes "projects", "LICENSE", "Power BI Embedding with SharePoint Framework Web Parts.pdf", "Power BI Embedding with SharePoint Framework Web Parts.pptx", and "README.md". A dropdown menu is open for the "projects" folder, showing the following subfolders: "PowerBiDaySpa", "cpt-powerbi-spfx-webparts", and "powerbi-embed-react-demo".





Critical Path Training

<https://www.CriticalPathTraining.com>

- PBI365: Power BI Certification Bootcamp – 3 Days
 - For people who have used Power BI Desktop for 6 months or more
- PBD365: Power BI Developer Bootcamp – 4 Days
 - For professional developers working with the Power BI platform
- DDPAF: Deep Dive into Power Apps and Flow – 2 Days
 - For people just getting started with Power Apps and Flow



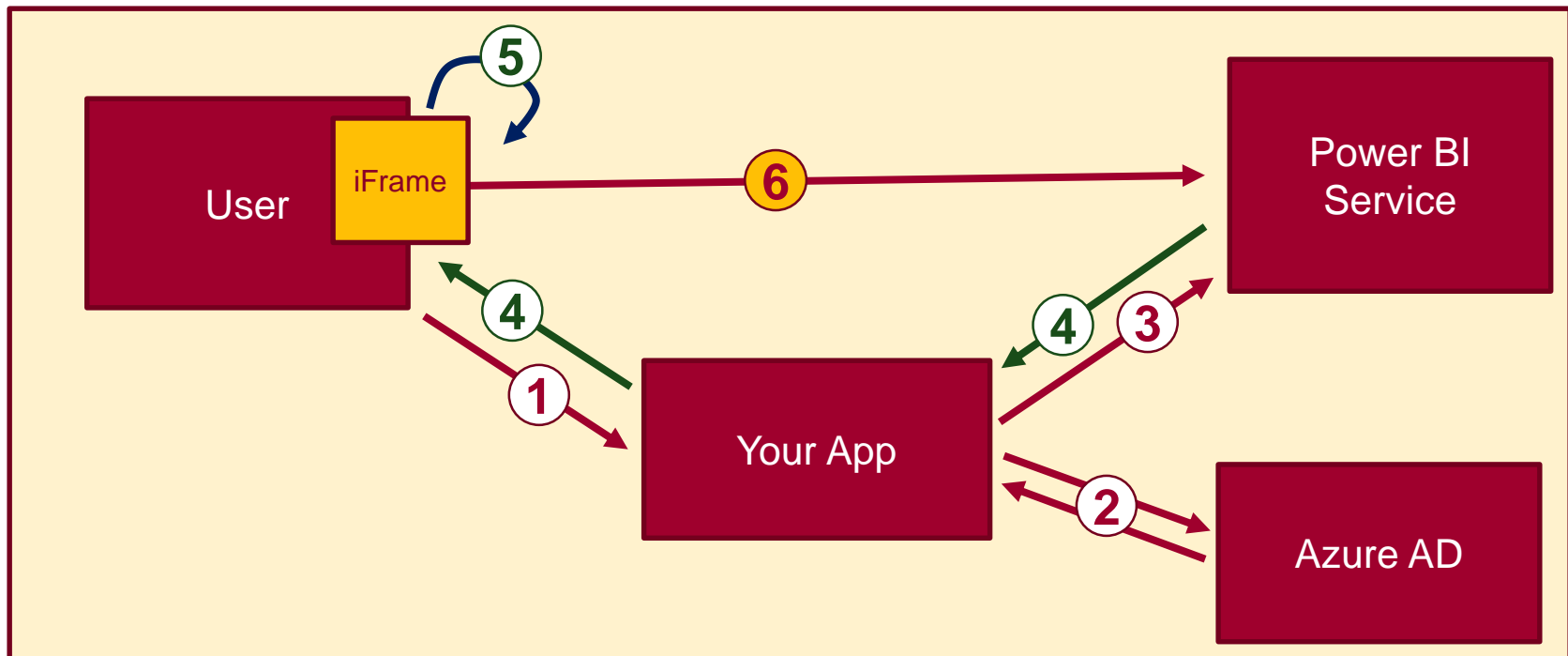
Agenda

- Power BI Embedding Fundamentals
 - Authentication with Azure Active Directory
 - Programming with Power BI Service API
 - Embedding with Power BI JavaScript API
 - SharePoint Framework Web Part



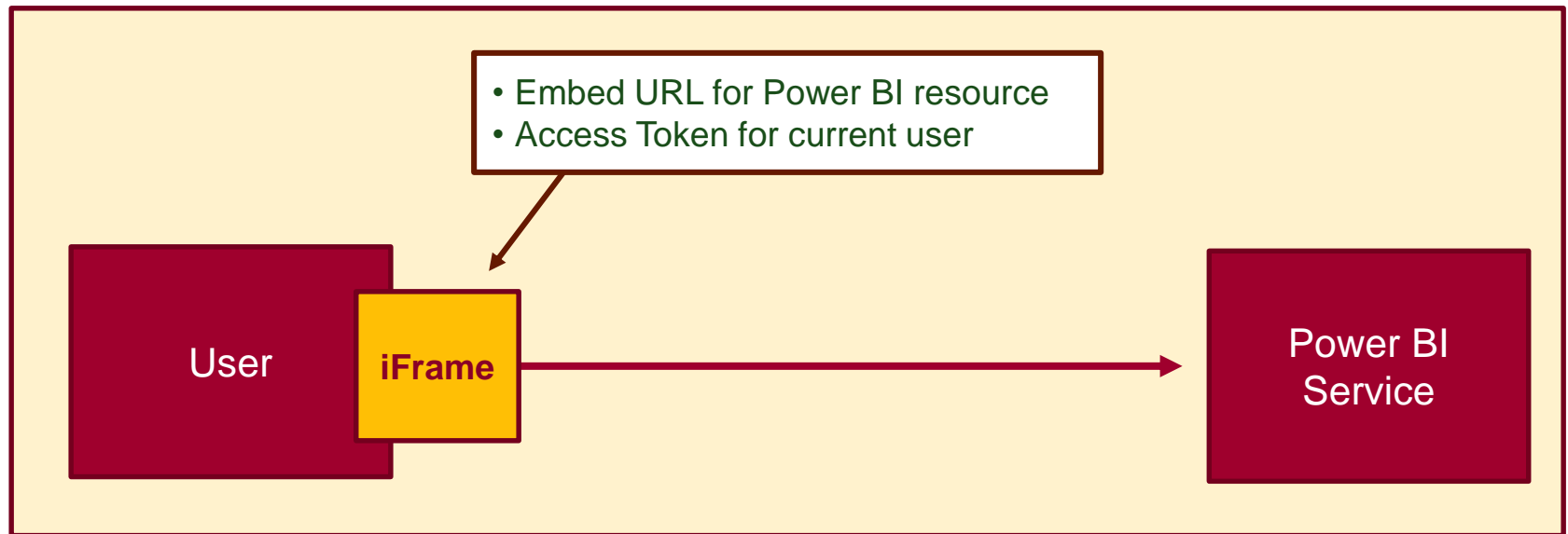
Power BI Embedding – The Big Picture

1. User launches your app using a browser
2. App authenticates with Azure Active Directory and obtains access token
3. App uses access token to call to Power BI Service API
4. App retrieves data for embedded resource and passes it to browser.
5. Client-side code uses Power BI JavaScript API to create embedded resource
6. Embedded resource session created between browser and Power BI service



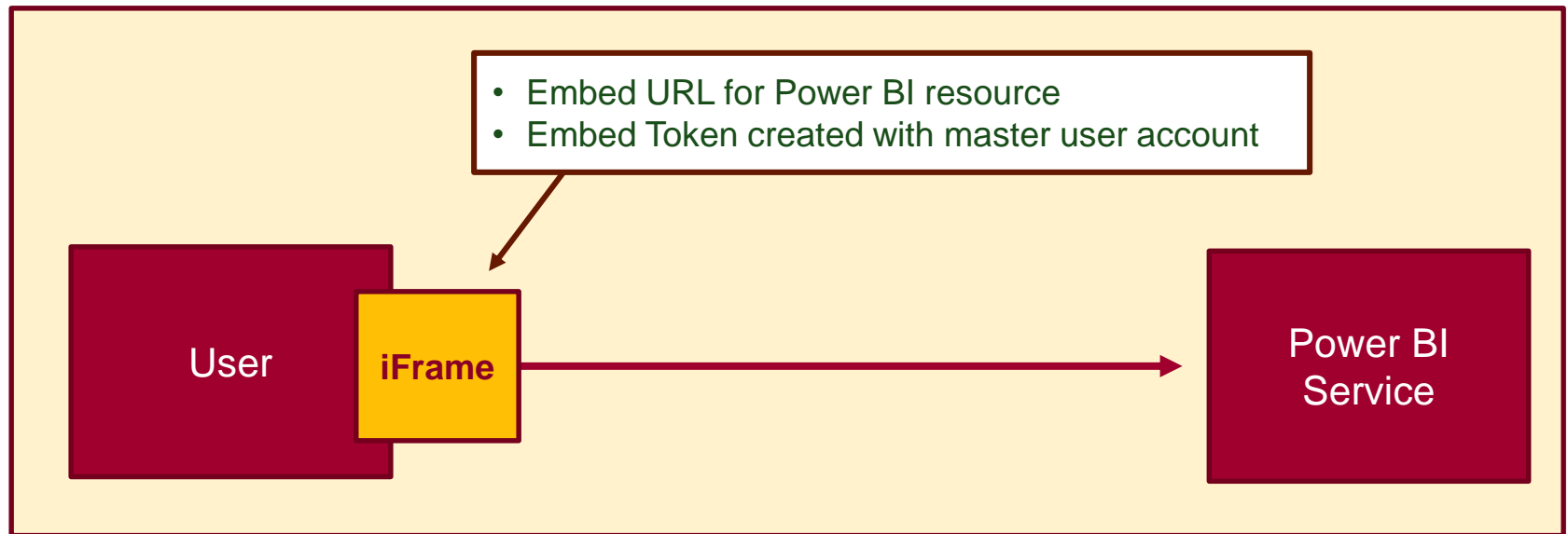
First Party Embedding

- App authenticates current user with Azure AD
 - Your code accesses Power BI Service as current user
 - Embedding requires Azure AD access token for user
 - User requires Azure AD account and Power BI license
 - Your code has access to whatever user has access to



Third Party Embedding

- App authenticates using Master User Account
 - Your code accesses Power BI Service as master user
 - Embedding uses embed token instead of access token
 - Users don't need AAD accounts and Power BI licenses
 - Your code has access to whatever master has access to



First Party vs Third Party Embedding

- What scenarios use first party embedding?
 - Organizations where users have Power BI licenses
 - Embedding Power BI reports in SharePoint and Teams
 - Development should go beyond out-of-box experience
- What scenarios use third party embedding?
 - Scenarios where users don't have Power BI licenses
 - Applications which have custom identity providers
 - Applications which use identity provider other than AAD



Embeddable Resources

1. Reports
2. Dashboards
3. Dashboard Tiles
4. Q & A Experience
5. Visual *

* really just a trick you do when embedding a report





DEMO

The Daily Reporter Pro Sample App

Creating an Azure AD Application

The image shows a sequence of three overlapping screenshots from the Microsoft Azure portal, illustrating the process of creating and configuring an Azure AD application.

Top Screenshot: Create Application
The "Create" dialog box is open in the "premium demo tenant - App registrations" section. The "Name" field is filled with "My Native App" and has a green checkmark. The "Application type" is set to "Native". The "Redirect URI" is "https://localhost". A "Create" button is at the bottom.

Middle Screenshot: My Native App Overview
The "My Native App" overview page is shown. It includes a "Settings" button, a "Manifest" button, and a "Delete" button. The "Essentials" section displays the following information:

Property	Value
Display name	My Native App
Application ID	7802d697-c0d5-480f-8f30-5039226f02a7
Object ID	e9ee95cc-0e31-4705-a6ac-b2b10053df4b
Application type	Native
Home page	Managed application in local directory My Native App

An "All settings" button is located at the bottom right of the Essentials section.

Bottom Screenshot: Settings Page
The "Settings" page for the application is shown. It features a "Filter settings" search bar. The settings are organized into sections:

- GENERAL**
 - Properties
 - Redirect URIs
 - Owners
- API ACCESS**
 - Required permissions
 - Keys



Application Permissions

- Applications can be granted permissions to other applications
 - Application permissions are app-only permissions
 - Delegated permissions are (app + user) permissions
 - Delegated permissions requires 1-time consent from user

Required permissions

[+ Add](#) [Grant Permissions](#)

API	APPLICATION PERMI...	DELEGATED PERMIS...
Windows Azure Active Directory	6	
Power BI Service	0	

DELEGATED PERMISSIONS

Permission	Requires Admin
<input checked="" type="checkbox"/> Add data to a user's dataset (preview)	No
<input checked="" type="checkbox"/> View all Dashboards (preview)	No
<input checked="" type="checkbox"/> View all Datasets	No
<input checked="" type="checkbox"/> Read and Write all Datasets	No
<input checked="" type="checkbox"/> View content properties (preview)	No
<input checked="" type="checkbox"/> Create content (preview)	No
<input checked="" type="checkbox"/> View all Reports (preview)	No
<input checked="" type="checkbox"/> View all Groups	No
<input checked="" type="checkbox"/> View users Groups	No
<input checked="" type="checkbox"/> Read and Write all Reports	No



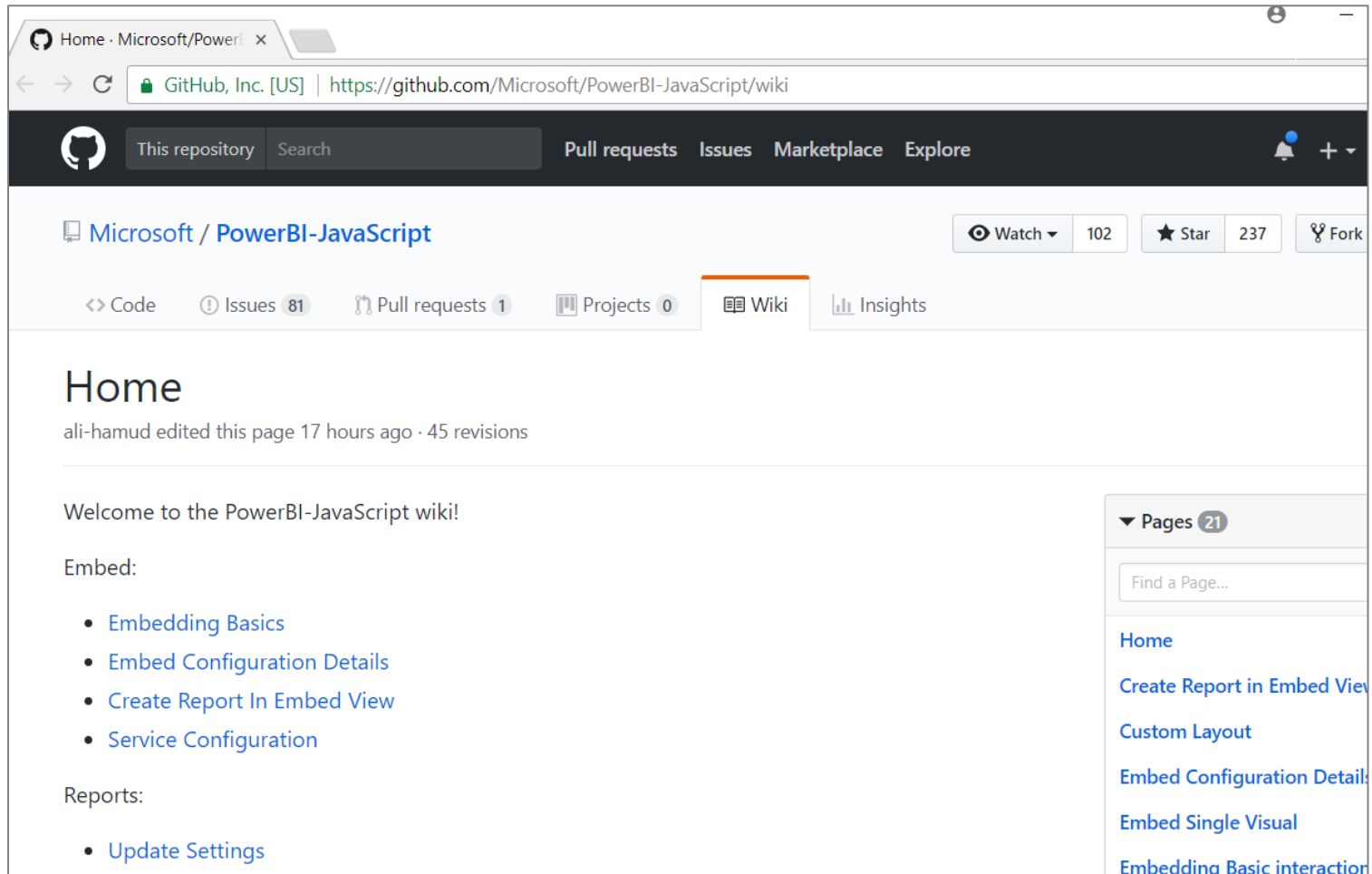


DEMO

PowerBiDaySpa Sample App

Power BI JavaScript API (PBIJS)

- <https://github.com/Microsoft/PowerBI-JavaScript/wiki>



The screenshot shows a web browser displaying the GitHub repository page for Microsoft/PowerBI-JavaScript. The browser's address bar shows the URL <https://github.com/Microsoft/PowerBI-JavaScript/wiki>. The repository page has a dark header with the GitHub logo, a search bar, and navigation links for Pull requests, Issues, Marketplace, and Explore. Below the header, the repository name "Microsoft / PowerBI-JavaScript" is displayed, along with statistics: 102 Watchers, 237 Stars, and a Fork button. The "Wiki" tab is selected, showing a "Home" page. The page content includes a welcome message, a list of links for embedding and configuration, and a sidebar with a list of 21 pages.

Home · Microsoft/PowerBI-JavaScript

GitHub, Inc. [US] | <https://github.com/Microsoft/PowerBI-JavaScript/wiki>

This repository Search Pull requests Issues Marketplace Explore

Microsoft / PowerBI-JavaScript Watch 102 Star 237 Fork

Code Issues 81 Pull requests 1 Projects 0 Wiki Insights

Home

ali-hamud edited this page 17 hours ago · 45 revisions

Welcome to the PowerBI-JavaScript wiki!

Embed:

- [Embedding Basics](#)
- [Embed Configuration Details](#)
- [Create Report In Embed View](#)
- [Service Configuration](#)

Reports:

- [Update Settings](#)

Pages 21

Find a Page...

- [Home](#)
- [Create Report in Embed View](#)
- [Custom Layout](#)
- [Embed Configuration Details](#)
- [Embed Single Visual](#)
- [Embedding Basic interaction](#)

Hello World with Power BI Embedding

- PBIJS library provides **powerbi** as top-level service object
 - You create configuration and then call **powerbi.embed** to embed a report
 - You must pass access token as part of the configuration

```
// data required for embedding Power BI report
var embedReportId = "f10c9de9-a325-4a43-af9f-0cf35cca2ab7";
var embedUrl = "https://app.powerbi.com/reportEmbed?reportId=f10c9de9-a325-4a43-af9f";
var accessToken = "H4sIAAAAAAEACWwtw6sCBZE_-wlrIR3K02A9x66gQzvVwe0_76tmbySW6pbdF7-Y";

// Get models object to access enums for embed configuration
var models = window['powerbi-client'].models;

var config = {
  type: 'report',
  id: embedReportId,
  embedUrl: embedUrl,
  accessToken: accessToken,
  tokenType: models.TokenType.Embed,
};

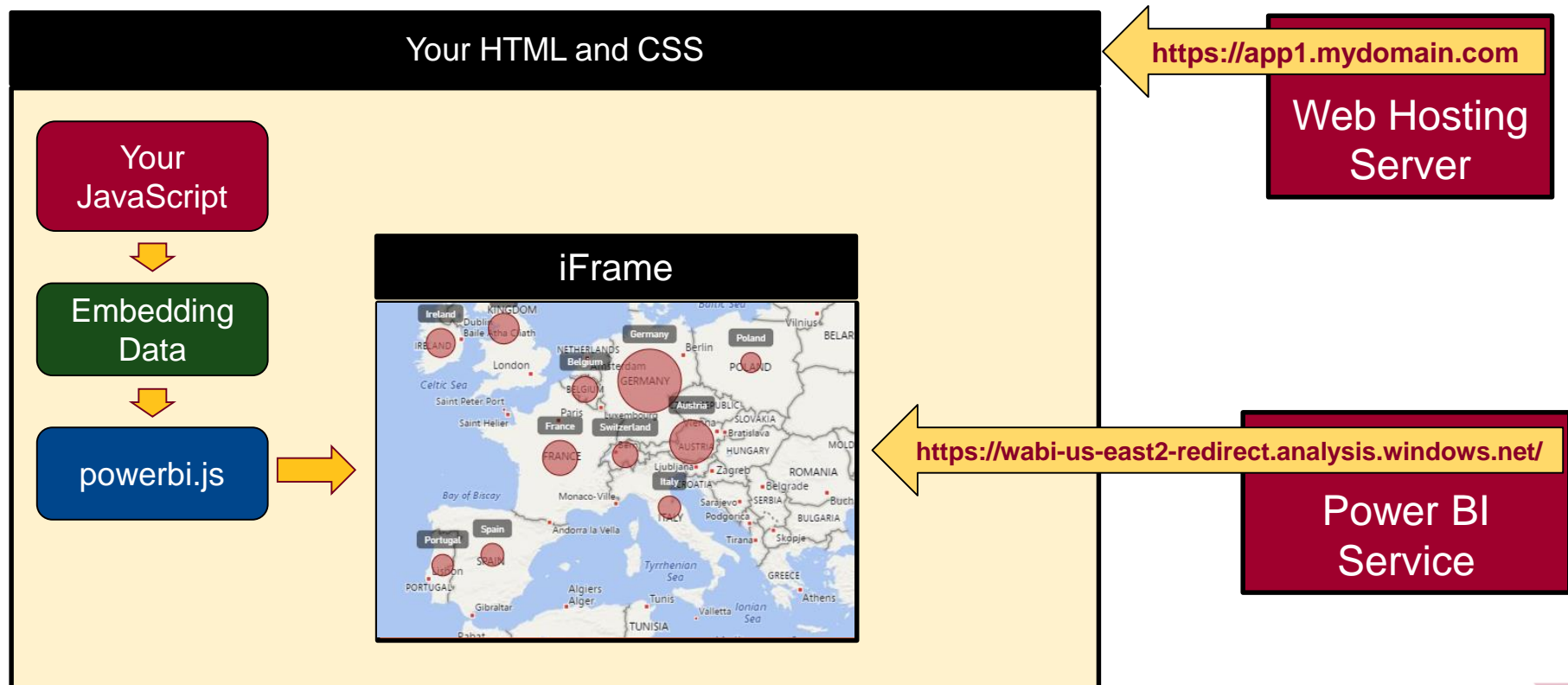
// Get a reference to the embedded report HTML element
var reportContainer = document.getElementById('embedContainer');

// Embed the report and display it within the div container.
var report = powerbi.embed(reportContainer, config);
```



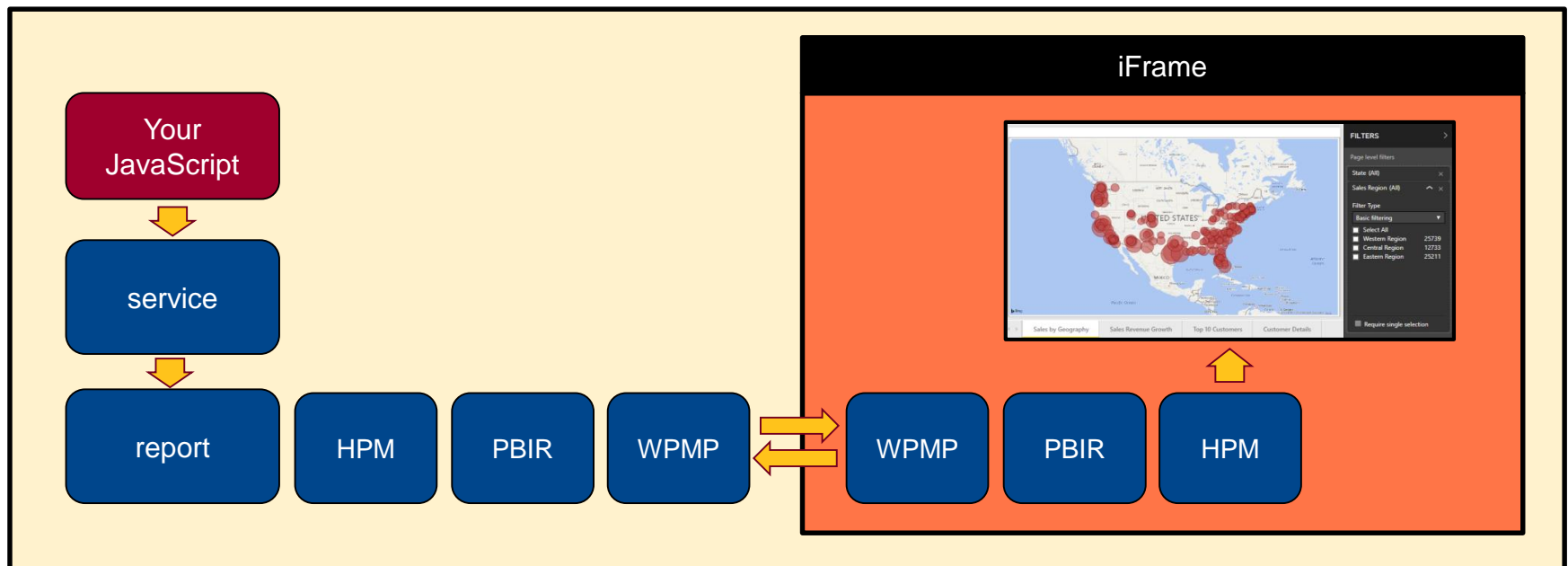
Report Embedding Architecture

- Embedding involves creating an iFrame on the page
 - PBIJS transparently creates iFrame and sets source to Power BI Service
 - ***The iFrame and hosting page originate from different DNS domains***



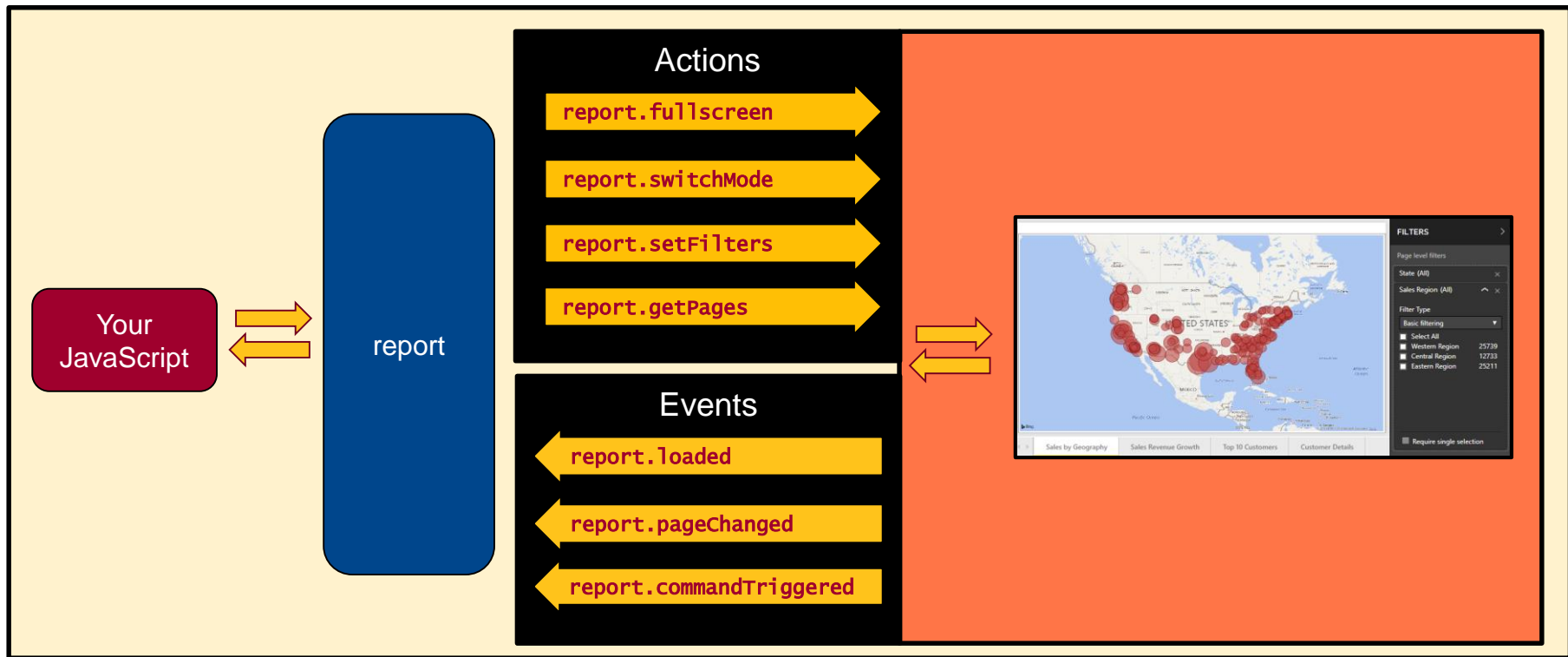
Post Message Communications Flow

- 4 extra libraries used communicate with report in iFrame
 - window-post-message-proxy (WPMP)
 - http-post-message (HPM)
 - powerbi-router (PBIR)
 - powerbi-models (PBIM)



A Promise-based Programming Model

- Design of PBIJS simulates HTTP protocol
 - Creates more intuitive programming model for developers
 - Programming based on asynchronous requests and promises
 - Embedded objects programmed using actions and events



PowerBiEmbeddedScratchpad Sample

<https://github.com/CriticalPathTraining/PowerBiEmbeddedScratchpad>

The screenshot shows the GitHub repository page for **CriticalPathTraining / PowerBiEmbeddedScratchpad**. The repository is described as "A sample application for developer's learning about Power BI Embedding". It has 4 commits, 1 branch, 0 releases, 1 contributor, and is licensed under MIT. The repository is currently on the **master** branch. A table of recent updates is shown below the repository information.

Update	Commit	Time
TedPattison Updates	3d2e699	7 days ago
DemoPagesWithEmbedding	Updates	8 days ago
PBIX	Updates	7 days ago
PowerBiEmbeddedScratchpad	Updates	8 days ago





DEMO

The Power BI Embedded Scratchpad App

Create a New SPFX Web Part Project

```
c:\Vegas\PBIESPFX\powerbi-api-demo>yo @microsoft/sharepoint
```

```
--(o)--  
(_ ^U _ )  
/_A_\br/> | ~ |  
-.-'-o'-'--Y-
```

```
Welcome to the  
SharePoint Client-side  
Solution Generator
```

Let's create a new SharePoint solution.

? What is your solution name? powerbi-api-demo

? Which baseline packages do you want to target for your component(s)? SharePoint Online only (latest)

? Where do you want to place the files? Use the current folder

? Do you want to allow the tenant admin the choice of being able to deploy the solution to all sites immediately without running any feature deployment or adding apps in sites? Yes

? Which type of client-side component to create? WebPart

? What is your Web part name? reportlist

? What is your Web part description? A sample web part which calls the Power BI Service API

? Which framework would you like to use? (Use arrow keys)

> No JavaScript framework

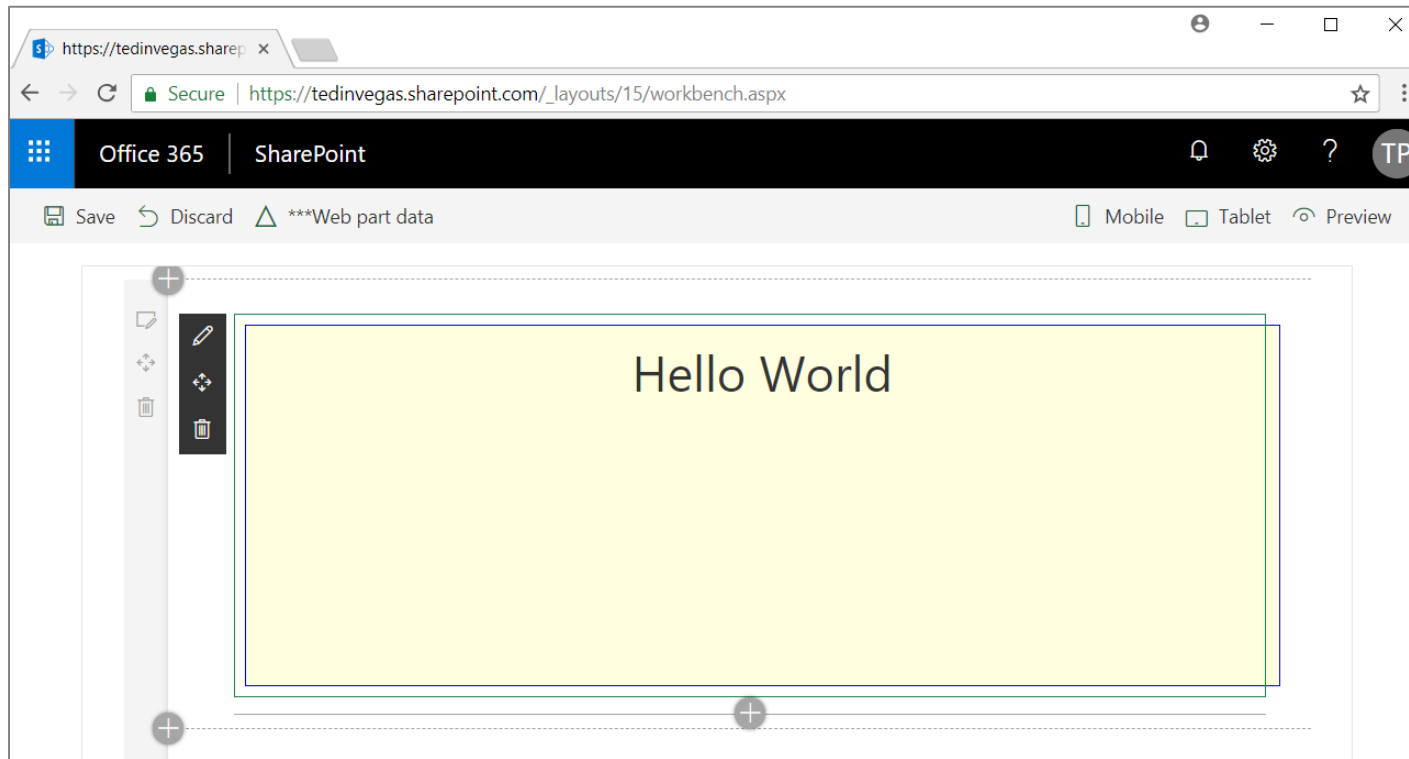
React

Knockout

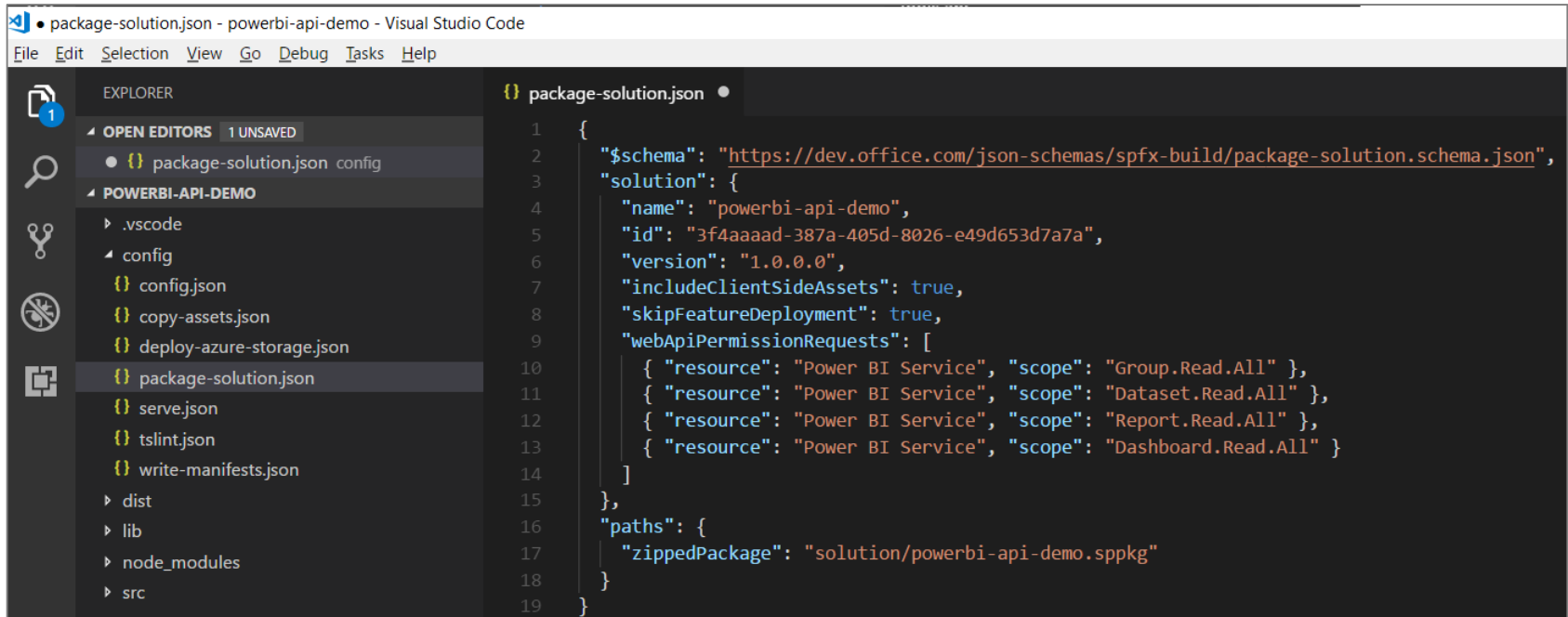


SharePoint Workbench

- You must test/debug in hosted workbench



Configuring Web API Permissions

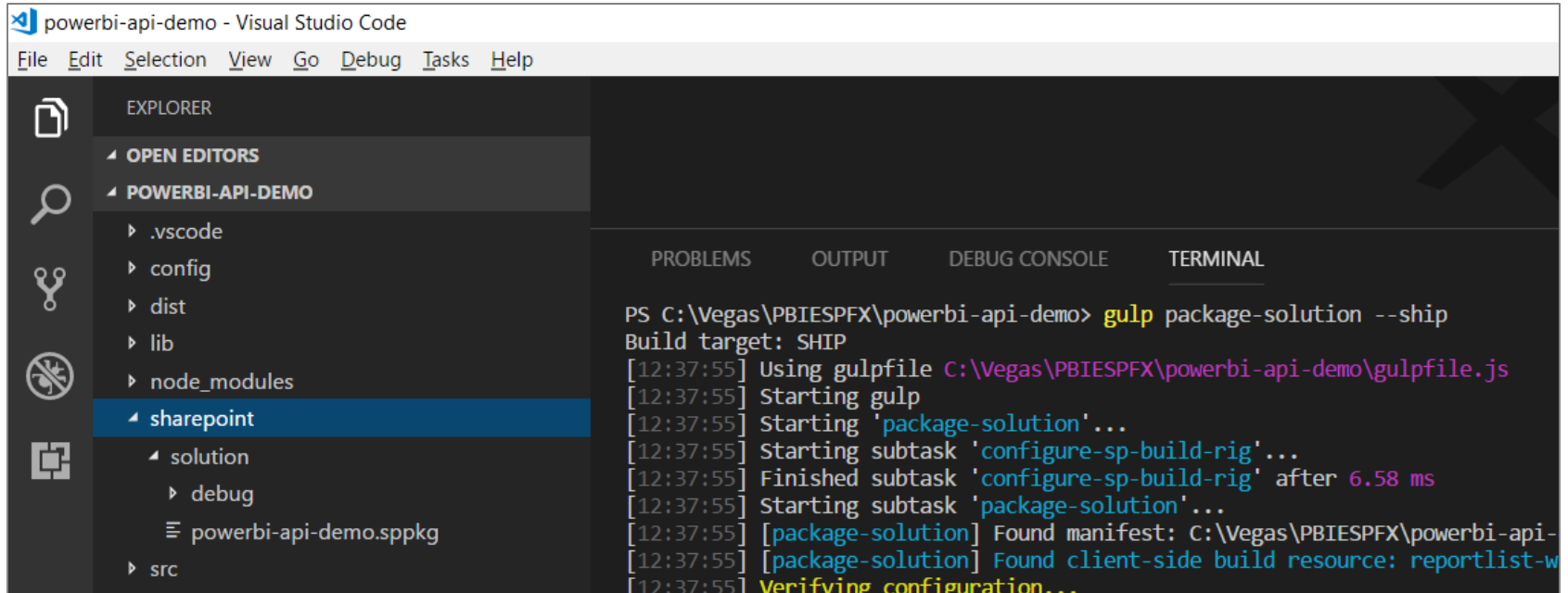


The screenshot shows the Visual Studio Code interface with the file `package-solution.json` open in the editor. The Explorer sidebar on the left shows the project structure for `powerbi-api-demo`, with `package-solution.json` selected under the `config` folder. The editor displays the following JSON configuration:

```
1 {
2   "$schema": "https://dev.office.com/json-schemas/spfx-build/package-solution.schema.json",
3   "solution": {
4     "name": "powerbi-api-demo",
5     "id": "3f4aaaad-387a-405d-8026-e49d653d7a7a",
6     "version": "1.0.0.0",
7     "includeClientSideAssets": true,
8     "skipFeatureDeployment": true,
9     "webApiPermissionRequests": [
10      { "resource": "Power BI Service", "scope": "Group.Read.All" },
11      { "resource": "Power BI Service", "scope": "Dataset.Read.All" },
12      { "resource": "Power BI Service", "scope": "Report.Read.All" },
13      { "resource": "Power BI Service", "scope": "Dashboard.Read.All" }
14    ],
15  },
16  "paths": {
17    "zippedPackage": "solution/powerbi-api-demo.sppkg"
18  }
19 }
```



Packaging Your SPFX Solution



```
powerbi-api-demo - Visual Studio Code
File Edit Selection View Go Debug Tasks Help

EXPLORER
└─ OPEN EDITORS
└─ POWERBI-API-DEMO
    ├── .vscode
    ├── config
    ├── dist
    ├── lib
    ├── node_modules
    └─ sharepoint
        ├── solution
        │   ├── debug
        │   └─ powerbi-api-demo.sppkg
        └─ src

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
PS C:\Vegas\PBIESPFX\powerbi-api-demo> gulp package-solution --ship
Build target: SHIP
[12:37:55] Using gulpfile C:\Vegas\PBIESPFX\powerbi-api-demo\gulpfile.js
[12:37:55] Starting gulp
[12:37:55] Starting 'package-solution'...
[12:37:55] Starting subtask 'configure-sp-build-rig'...
[12:37:55] Finished subtask 'configure-sp-build-rig' after 6.58 ms
[12:37:55] Starting subtask 'package-solution'...
[12:37:55] [package-solution] Found manifest: C:\Vegas\PBIESPFX\powerbi-api-
[12:37:55] [package-solution] Found client-side build resource: reportlist-w
[12:37:55] Verifying configuration...
```



Deploy the Web Part to the App Gallery

The screenshot displays the SharePoint 'Apps for SharePoint' page. At the top, there's a 'Home' link and the title 'Apps for SharePoint' with an information icon. Below this are action buttons: 'New', 'Upload', 'Sync', 'Share', and a 'More' dropdown. A navigation bar includes 'All Apps', 'Featured Apps', 'Unavailable Apps', and a search box labeled 'Find a file'. A table with columns 'Title', 'Name', and 'App Version' is partially visible. Overlaid on this is a modal dialog titled 'Do you trust powerbi-api-demo?'. The dialog contains a warning about full trust client-side code, a list of domains (SharePoint Online), a checked checkbox for 'Make this solution available to all sites in the organization', and instructions to go to the Service Principal Permissions Management Page. At the bottom of the dialog are 'Deploy' and 'Cancel' buttons. In the background, a table with columns 'Name', 'Version', and 'Deployed' is partially visible, showing entries for 'powerbi-api-demo'.

Home

Apps for SharePoint ⓘ

New Upload Sync Share More ▾

All Apps Featured Apps Unavailable Apps ... Find a file 🔍

✓	📄	Title	Name	App Version
---	---	-------	------	-------------

Do you trust powerbi-api-demo?

The client-side solution you are about to deploy contains full trust client side code. The components in the solution can, and usually do, run in full trust, and no resource usage restrictions are placed on them.

This client side solution will get content from the following domains:

SharePoint Online

☒ Make this solution available to all sites in the organization

Please go to the Service Principal Permissions Management Page to approve pending permissions.

Deploy Cancel

	Name	Version	Deployed
	powerbi-api-demo		





Granting Web API Permissions

API management

Control the third-party APIs that can be called by apps and custom script.

^ API name	Permission	Access
^ Pending approval (0)		
^ Approved (4)		
Power BI Service	Group.Read.All	Approved
Power BI Service	Group.Read.All Dataset.Read.All	Approved
Power BI Service	Group.Read.All Dataset.Read.All Report.	Approved
Power BI Service	Group.Read.All Dataset.Read.All Report.	Approved

[admin center](#)  [feedback](#) 



Calling the Power BI Service API

```
import { AadHttpClient, HttpClientResponse } from '@microsoft/sp-http';

export default class ReportlistWebPart extends BaseClientSideWebPart<any> {

  private powerbiApiResourceId = "https://analysis.windows.net/powerbi/api";
  private pbiClient: AadHttpClient;

  protected onInit(): Promise<void> {
    this.pbiClient = new AadHttpClient(this.context.serviceScope, this.powerbiApiResourceId);
    return Promise.resolve();
  }

  public render(): void {
    var urlReports: string = "https://api.powerbi.com/v1.0/myorg/reports/";
    this.pbiClient.get(urlReports, AadHttpClient.configurations.v1)
      .then((res: HttpClientResponse): Promise<any> => {
        return res.json();
      })
      .then((reports: any): void => {
        this.domElement.innerHTML =
          `<h2>Power BI Reports</h2>
          <ul>
            ${ reports.value.map(r => `<li><a href='${r.webUrl}' target='_blank' >${r.name}</a></li>`).join("") }
          </ul>`;
      });
  }
}
```

Power BI Reports

- [Northwind Retro](#)
- [Wingtip Sales Analysis](#)



npm install powerbi-client --save

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
PS C:\Vegas\PBIESPFX\embed-report-demo> npm install powerbi-client --save
npm WARN optional SKIPPING OPTIONAL DEPENDENCY: fsevents@1.2.4 (node_modules\fsevents)
npm WARN notsup SKIPPING OPTIONAL DEPENDENCY: Unsupported platform for fsevents@1.2.4
)
+ powerbi-client@2.5.1
added 9 packages in 22.762s
```

- postcss-value-parser
- postcss-zindex
- powerbi-client
- powerbi-models
- powerbi-router
- prelude-ls

```
{ tsconfig.json x
1 {
2   "compilerOptions": {
3     "target": "es5",
4     "forceConsistentCasingInFileNames": true,
5     "module": "commonjs",
6     "jsx": "react",
7     "declaration": true,
8     "sourceMap": true,
9     "experimentalDecorators": true,
10    "skipLibCheck": true,
11    "typeRoots": [
12      "./node_modules/@types",
13      "./node_modules/@microsoft",
14      "./node_modules/powerbi-client",
15      "./node_modules/powerbi-models"
16    ],
17    "types": [
18      "es6-promise",
19      "webpack-env"
20    ],
21    "lib": [
22      "es5",
23      "dom",
24      "es2015.collection"
25    ]
26  }
27 }
28 }
```



Embedding Code

```
public render(): void {
    let hostDiv: JQuery = $(this.domElement);
    let height: string = this.properties.reportHeight + "px";
    hostDiv.empty().css({ "margin": "0", "padding": "0", "height": height });

    var reqHeaders: HeadersInit = new Headers();
    reqHeaders.append("Accept", "*");
    this.pbiclient.get(this.reportUrl, AadHttpClient.configurations.v1, { headers: reqHeaders })
        .then((res: HttpClientResponse): Promise<any> => {
            return res.json();
        })
        .then((report: any): void => {
            console.log("begin embed...");
            var embedReportId: string = report.id;
            var embedUrl: string = report.embedUrl;
            var accessToken: string = window.sessionStorage["adal.access.token.keyhttps://analysis.windows.net/powerbi/api"];
            // Get models object to access enums for embed configuration
            var models = pbimodels;

            var config: any = {
                type: 'report',
                id: embedReportId,
                embedUrl: embedUrl,
                accessToken: accessToken,
                tokenType: models.TokenType.Aad,
                permissions: models.Permissions.All,
                viewMode: models.ViewMode.View,
                settings: {
                    filterPaneEnabled: false,
                    navContentPaneEnabled: this.properties.showPageTabs,
                }
            };
            window.powerbi.reset(this.domElement);
            window.powerbi.embed(this.domElement, config);
        });
}
```



Summary

- ✓ Power BI Embedding Fundamentals
- ✓ Authentication with Azure Active Directory
- ✓ Programming with Power BI Service API
- ✓ Embedding with Power BI JavaScript API
- ✓ SharePoint Framework Web Part

