Implementing an API Specification in Anypoint Code Builder

Prerequisites

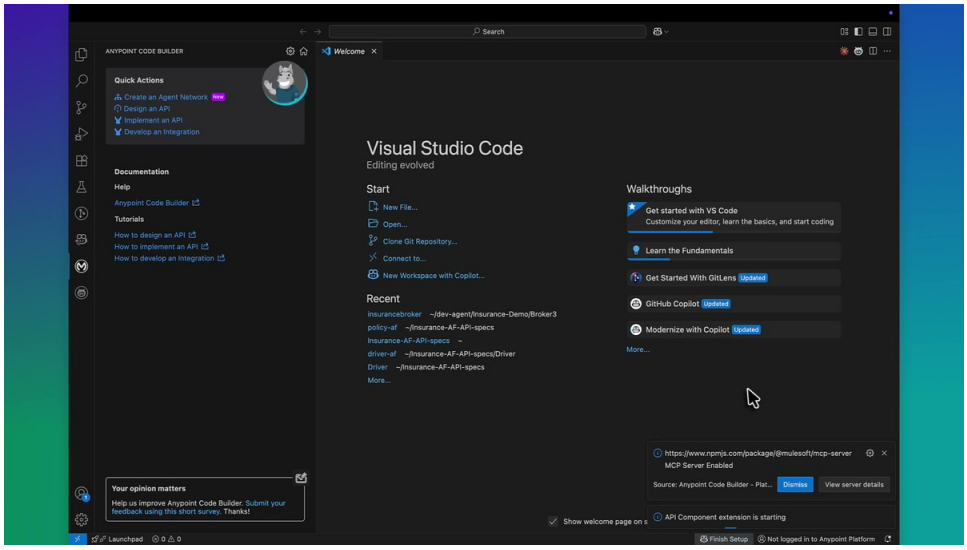
• Installed Visual Studio Code with Anypoint Code Builder Extension.

• An existing Anypoint account.

• Basic knowledge of MuleSoft integration workflows.

Steps

Step 1: Open Visual Studio Code

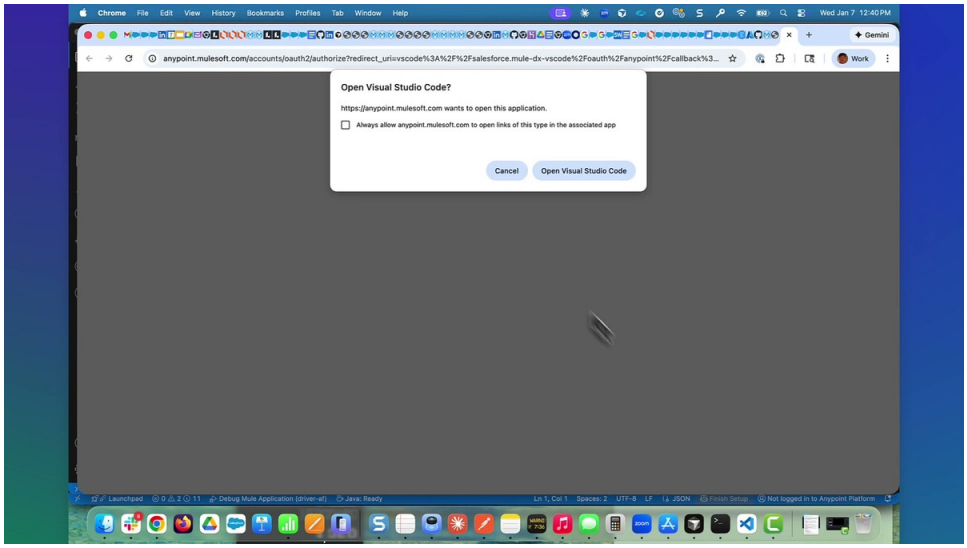


Action: Open Visual Studio Code with Anypoint Code Builder installed.

Location: Start screen in Visual Studio Code.

Details: Ensure the Anypoint Code Builder extension is active.

Step 2: Enable Anypoint Platform Login

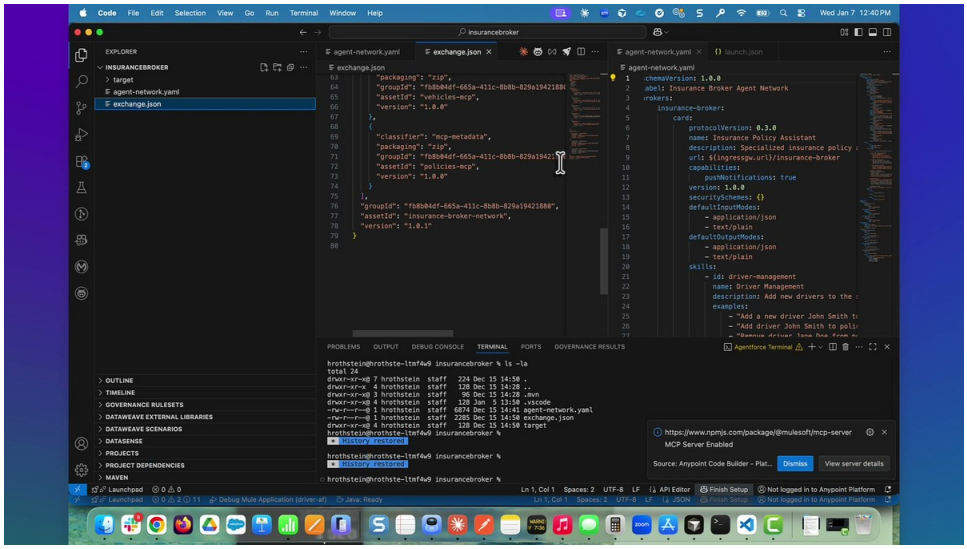


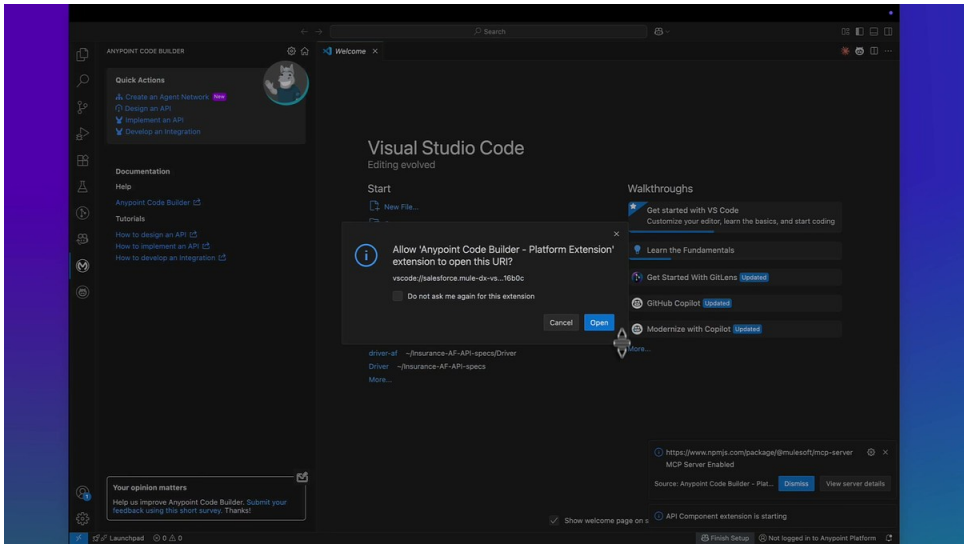
Action: Click **Allow** when prompted to enable Anypoint Platform extension login.

Location: Popup at the center of the screen.

Details: This step enables you to sign in using your Anypoint account credentials.

Step 3: Confirm External Website Access



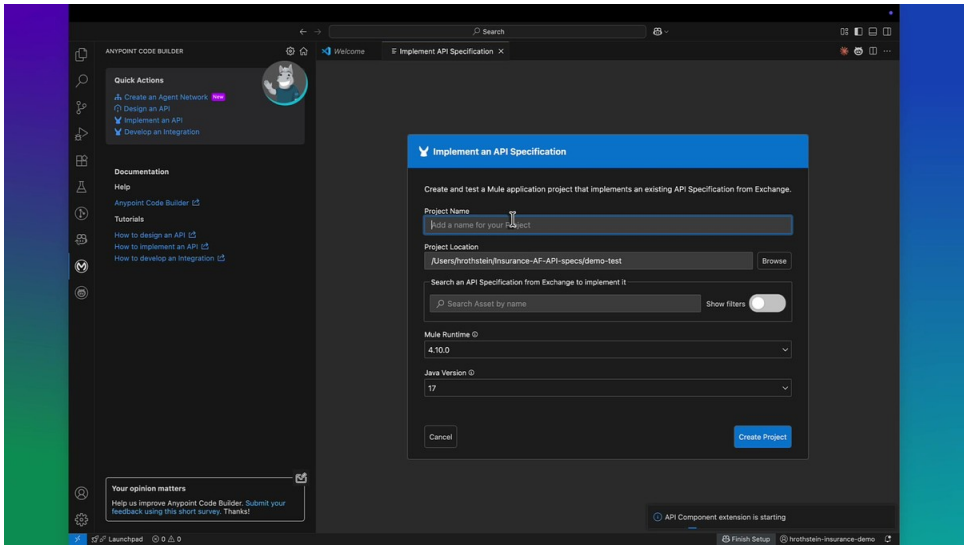


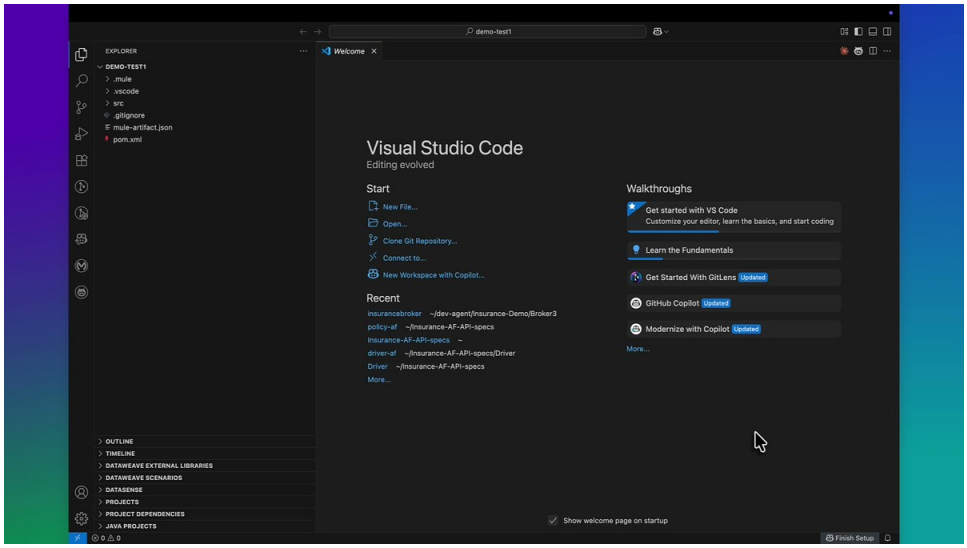
Action: Click **Open** to grant access to an external website for authentication.

Location: Browser application popup.

Details: Ensure you allow Visual Studio Code to open authentication prompts.

Step 4: Complete Platform Login



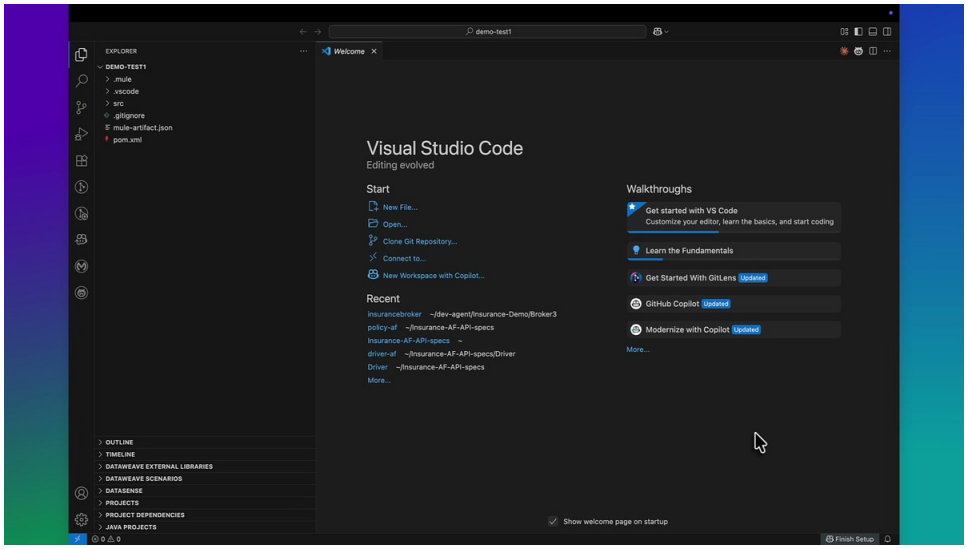


Action: Log in and return to Visual Studio Code after authentication.

Location: Welcome screen in Visual Studio Code.

Details: Confirm successful authentication via the notification bar displaying your login status.

Step 5: Navigate to Quick Actions

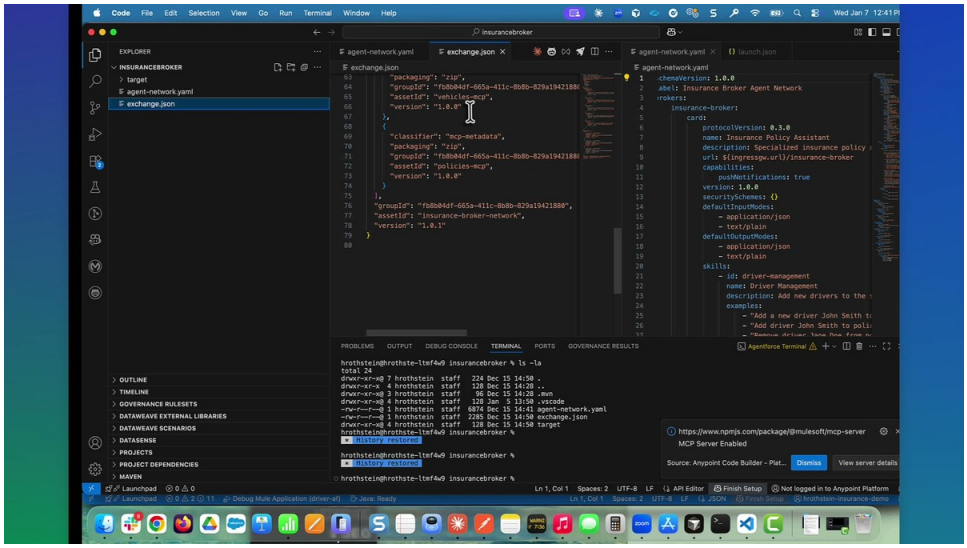


Action: Click **Quick Actions** on the left sidebar.

Location: Sidebar menu in Visual Studio Code.

Details: Select "Implement an API" from the Quick Actions menu.

Step 6: Configure API Implementation Settings



Action: Set up project preferences, including project name, location, and runtime details.

Location: API specification configuration window.

Details:

• Enter project name, e.g., "demo-test1."

• Browse and select project location.

• Choose Mule Runtime (e.g., 4.10.0) and Java version (e.g., 17).

Step 7: Select API Specification

Action: Navigate to API specification.

Location: File Explorer popup for project location.

Details: Browse folders or create a new folder — name it appropriately, e.g., "demo-test."

Step 8: Search and Add Specification

Action: Search and select the required specification, e.g., "driver-catalog."

Location: Specification search box under "Search an API Specification from Exchange."

Details: Use the text box or filters for refined search and click **Add Asset** for preferred APIs.

Step 9: Create Project

Action: Click **Create Project** to finalize and establish your integration setup.

Location: Bottom of the configuration window.

Details: Monitor progress bar for initialization and scaffolding setup.

Step 10: View Created Project

Action: Check the created project structure.

Location: Explorer sidebar.

Details: Validate project scaffolding (e.g., `mule`, `.src`, `mule-artifact.json`, etc.).

Step 11: Open API Flows

Action: Access API flows from the project.

Location: Flow designer view.

Details: Visualize "Flow canvas" components like listeners, routers, and transform nodes.

Step 12: Edit Integration Workflows

Action: Use the editor to modify integration workflows for API specification implementation.

Location: Code editor view.

Details: Update XML or JSON files for routing and transformation.

Step 13: Export Project

Action: Click **Export** in Camtasia for recording and exporting workflow for project documentation.

Location: Top bar of Camtasia software.

Details: Ensure project workflows are fully documented visually for later reference.

Notes

• Ensure the correct versions of Mule Runtime and Java are used as per project needs.

• Regularly validate project settings before creation to avoid integration errors.

• Export illustrated workflows for references when working in large teams.