

Civil AI Estimator

Automating Quantity Takeoff in Revit 2025

Powered by OpenAI & Revit API
Developer: Lynn

THE PROBLEM

Manual takeoff is slow (hours/days).

Coding standards (CESMM4) is complex.

Excel spreadsheets are disconnected from BIM.

THE SOLUTION

A native **Revit Add-in** that connects to **ChatGPT**.

- Reads Geometry (Volumes/Areas)
- Identifies Materials
- Assigns Cost Codes automatically

INSTALLATION WORKFLOW

1. Download Source from GitHub

<https://github.com/Lynn-hh/Revit-Civil-AI-Estimator/tree/main/RevitAiEstimator>

2. Download VisualStudio

<https://visualstudio.microsoft.com/>

3. Paste API Key in AiService.cs

4. Build Solution (Ctrl+Shift+B)

Revit magically loads the plugin!

1. Download Source from GitHub

<https://github.com/Lynn-hh/Revit-Civil-AI-Estimator/tree/main/RevitAiEstimator>

The screenshot shows the GitHub repository page for 'Revit-Civil-AI-Estimator'. The repository is public and has 1 branch and 0 tags. The main file listed is 'README.md'.

The 'Code' tab is selected, highlighted by a yellow box. On the right side of the code area, there is a dropdown menu with 'Local' selected, also highlighted by a yellow box. The 'Clone' section shows options for HTTPS, SSH, and GitHub CLI, with the HTTPS URL <https://github.com/Lynn-hh/Revit-Civil-AI-Estimator> displayed. A 'Download ZIP' button is highlighted by a yellow box. Below the code area, there is a section for adding a README with a 'Add a README' button.

2. Download Visual Studio and open the GitHub file: RevitAiEstimator.sln

<https://visualstudio.microsoft.com/>

The screenshot shows the Microsoft Visual Studio interface. The 'File' menu is open, with the 'Open' option highlighted by a yellow box. Below it, the 'Project/Solution...' option is also highlighted with a yellow box. The main window displays a C# code editor for 'AiService.cs' with several TODO comments. The 'Solution Explorer' tab is selected at the bottom right.

```
// TODO: PASTE YOUR API KEY HERE
// SECURITY WARNING: Never commit your real API key to GitHub!
private const string ApiKey = "sk-YOUR_OPENAI_API_KEY_HERE"; private const string Endpoint = "https://api.openai.com/v1/chat/completions";
private async Task<EstimationResult> CallOpenAiAsync(ElementData input, string context)
{
    using (HttpClient client = new HttpClient())
    {
        client.DefaultRequestHeaders.Add("Authorization", $"Bearer {ApiKey}");
        var prompt = $"You are a top Civil Engineering Quantity Surveyor. " +
                    $"Context: {context}. " + // Added User Context
                    $"Classify this element into CESMM4 (Civil Engineering Standard Method of Measurement). " +
                    $"Element: Category={input.Category}, Family={input.FamilyName}, Type={input.TypeName}, Material={input.MaterialName}, Qty={input.Metr
                    $"Return ONLY a strict JSON object with fields: 'cost_code' (string), 'description' (string), 'unit_rate' (number). Do not include mar
}
return Task.Run(() => CallOpenAiAsync(input, context)).GetAwaiter().GetResult();
}
```

The screenshot shows a GitHub repository page for 'Revit-Civil-AI-Estimator / RevitAiEstimator /'. The repository contains several files listed on the right, including 'AiService.cs', 'App.cs', 'Command.cs', 'ElementExtractor.cs', 'EstimationOptionsForm.cs', 'ParameterManager.cs', 'README.md', 'RevitAiEstimator.addin', 'RevitAiEstimator.csproj', and 'RevitAiEstimator.sln'. The 'RevitAiEstimator.sln' file is highlighted with a yellow box.

3. Add your GPT API Key in file AiService.cs

How to get GPT key (next slide)

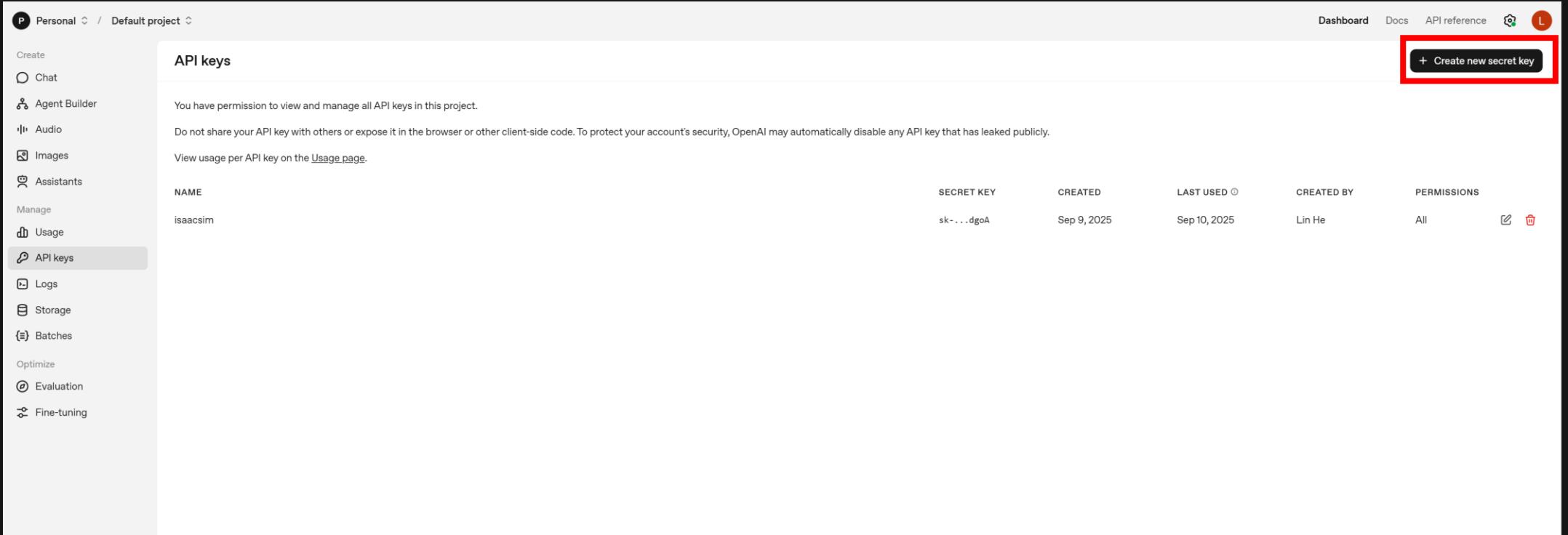
The screenshot shows the Visual Studio IDE interface. The main window displays the code for `AiService.cs`. The code defines two classes: `EstimationResult` and `AiService`. The `AiService` class contains a `private const string ApiKey` field and a `GetEstimation` method. A yellow box highlights the `ApiKey` declaration. The `Endpoint` field is also highlighted with a yellow box. The Solution Explorer on the right shows the project structure, with `AiService.cs` selected.

```
1  using System;
2  using System.Net.Http;
3  using System.Text;
4  using System.Threading.Tasks;
5  using Newtonsoft.Json;
6  using Newtonsoft.Json.Linq;
7
8  namespace RevitAiEstimator
9  {
10     public class EstimationResult
11     {
12         public string CostCode { get; set; } = "N/A";
13         public string Description { get; set; } = "AI Request Failed";
14         public double UnitRate { get; set; } = 0.0;
15     }
16
17     public class AiService
18     {
19         // TODO: PASTE YOUR API KEY HERE
20         // SECURITY WARNING: Never commit your real API key to GitHub!
21         private const string ApiKey = "sk-YOUR_OPENAI_API_KEY_HERE";
22
23         private const string Endpoint = "https://api.openai.com/v1/chat/completions";
24
25         public EstimationResult GetEstimation(ElementData input, string context)
26         {
27             try
28             {
29                 // Fix for UI Freeze: Run async task on a background thread to avoid deadlocking the Revit UI thread
30             }
31         }
32     }
33 }
```

How to get GPT key

First: go to <https://platform.openai.com/api-keys>

Second: create new secret key

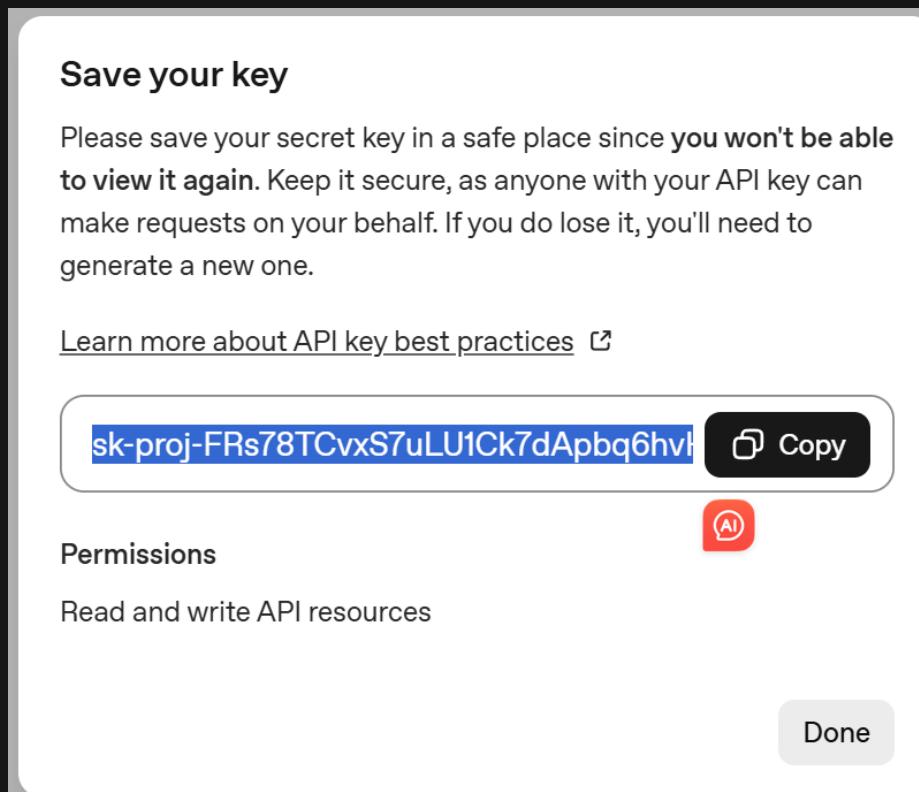


The screenshot shows the 'API keys' section of the OpenAI Platform. On the left, there's a sidebar with various project management and AI service links. The 'API keys' link is currently selected and highlighted in grey. The main content area is titled 'API keys' and contains instructions: 'You have permission to view and manage all API keys in this project.' and 'Do not share your API key with others or expose it in the browser or other client-side code. To protect your account's security, OpenAI may automatically disable any API key that has leaked publicly.' Below this, there's a table showing existing API keys. The table has columns for NAME, SECRET KEY, CREATED, LAST USED, CREATED BY, and PERMISSIONS. One row is visible, showing 'isaacsim' as the name, 'sk-...dgoA' as the secret key, 'Sep 9, 2025' as the creation date, 'Sep 10, 2025' as the last used date, 'Lin He' as the creator, and 'All' under permissions. A red rectangular box highlights the 'Create new secret key' button at the top right of the table area.

NAME	SECRET KEY	CREATED	LAST USED	CREATED BY	PERMISSIONS
isaacsim	sk-...dgoA	Sep 9, 2025	Sep 10, 2025	Lin He	All

How to get GPT key

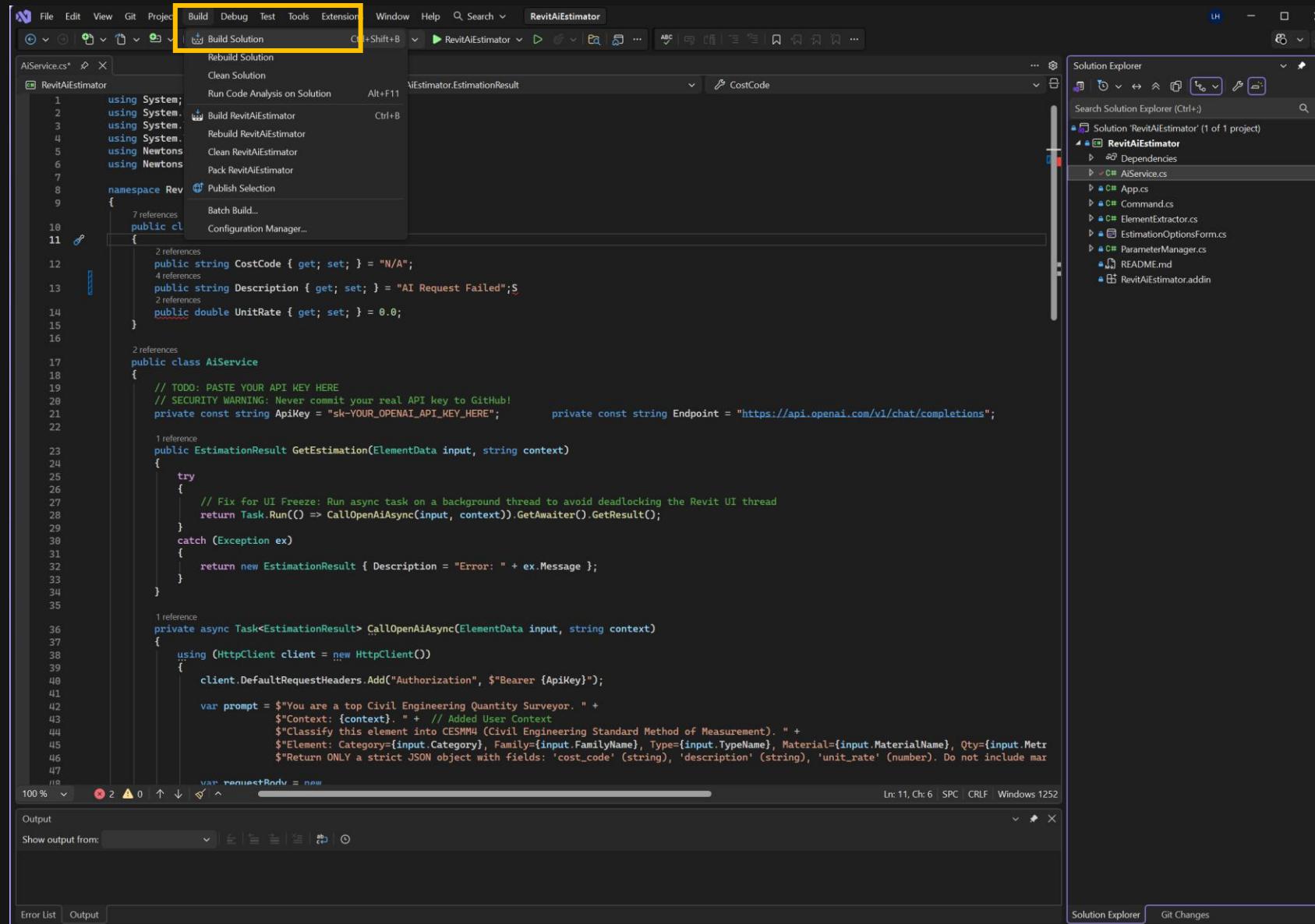
Third: get your key (Make sure not to expose your key to anyone on the internet, since it can generate expenses) and put it to excel



The screenshot shows a Visual Studio code editor with an open file named "AIService.cs". The code defines two classes: "EstimationResult" and "AiService". The "AiService" class contains a comment "// TODO: PASTE YOUR API KEY HERE" above a line where the variable "ApiKey" is defined. This line is highlighted with a yellow box. The "ApiKey" variable is set to the placeholder value "sk-YOUR_OPENAI_API_KEY_HERE". The "AiService" class also contains a method "GetEstimation" which makes a call to the OpenAI API endpoint "https://api.openai.com/v1/chat/completions". The "Solution Explorer" panel on the right shows the project structure, with "AIService.cs" highlighted by a yellow box.

```
1  using System;
2  using System.Net.Http;
3  using System.Text;
4  using System.Threading.Tasks;
5  using Newtonsoft.Json;
6  using Newtonsoft.Json.Linq;
7
8  namespace RevitAiEstimator
9  {
10    public class EstimationResult
11    {
12      public string CostCode { get; set; } = "N/A";
13      public string Description { get; set; } = "AI Request Failed";
14      public double UnitRate { get; set; } = 0.0;
15    }
16
17    public class AiService
18    {
19      // TODO: PASTE YOUR API KEY HERE
20      private const string ApiKey = "sk-YOUR_OPENAI_API_KEY_HERE";
21      private const string Endpoint = "https://api.openai.com/v1/chat/completions";
22
23      public EstimationResult GetEstimation(ElementData input, string context)
24      {
25        try
26        {
27          // Fix for UI Freeze: Run async task on a background thread to avoid deadlocking the Revit UI thread
28          return Task.Run(() => CallOpenAiAsync(input, context)).GetAwaiter().GetResult();
29        }
30        catch (Exception ex)
31        {
32          return new EstimationResult { Description = "Error: " + ex.Message };
33        }
34      }
35    }
36 }
```

4. Build Solution (Ctrl+Shift+B)



5. Install to Revit:

Go to the output folder bin\Debug\net8.0-windows\.

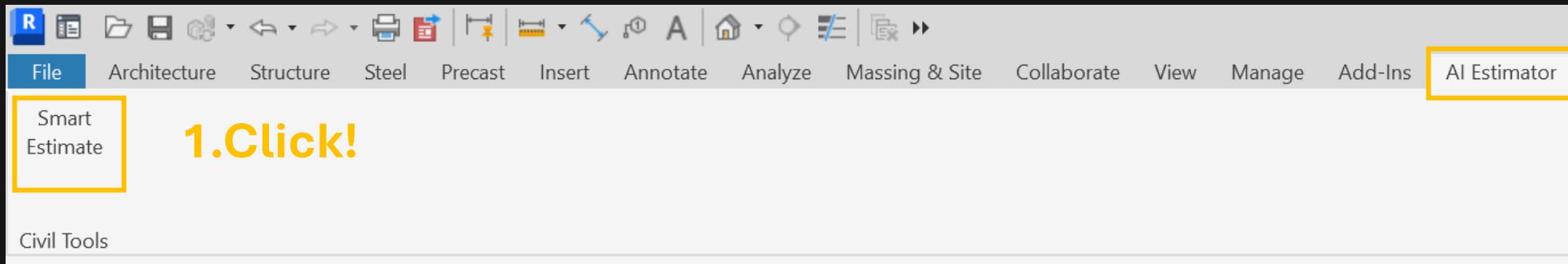
Open RevitAiEstimator.addin and check the path.

Copy **RevitAiEstimator.addin** to
%ProgramData%\Autodesk\Revit\Addins\2025\

Open Revit and choose **always load**

Revit magically loads the plugin!

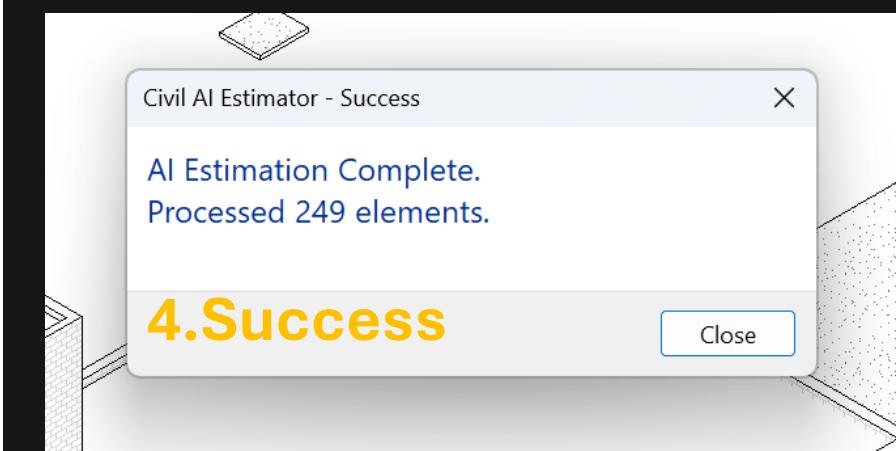
6. Revit magically loads the plugin



1.Click!



2.In the dialog box, enter your Project Context
3.And Click Start



4.Success

Demo: The "Magic" Button

We built a context-aware dialog:

"Project located in Austin, Texas.

Use local labor rates & limestone excavation costs.

Currency: USD."

(AI adjusts prices based on THIS input!)

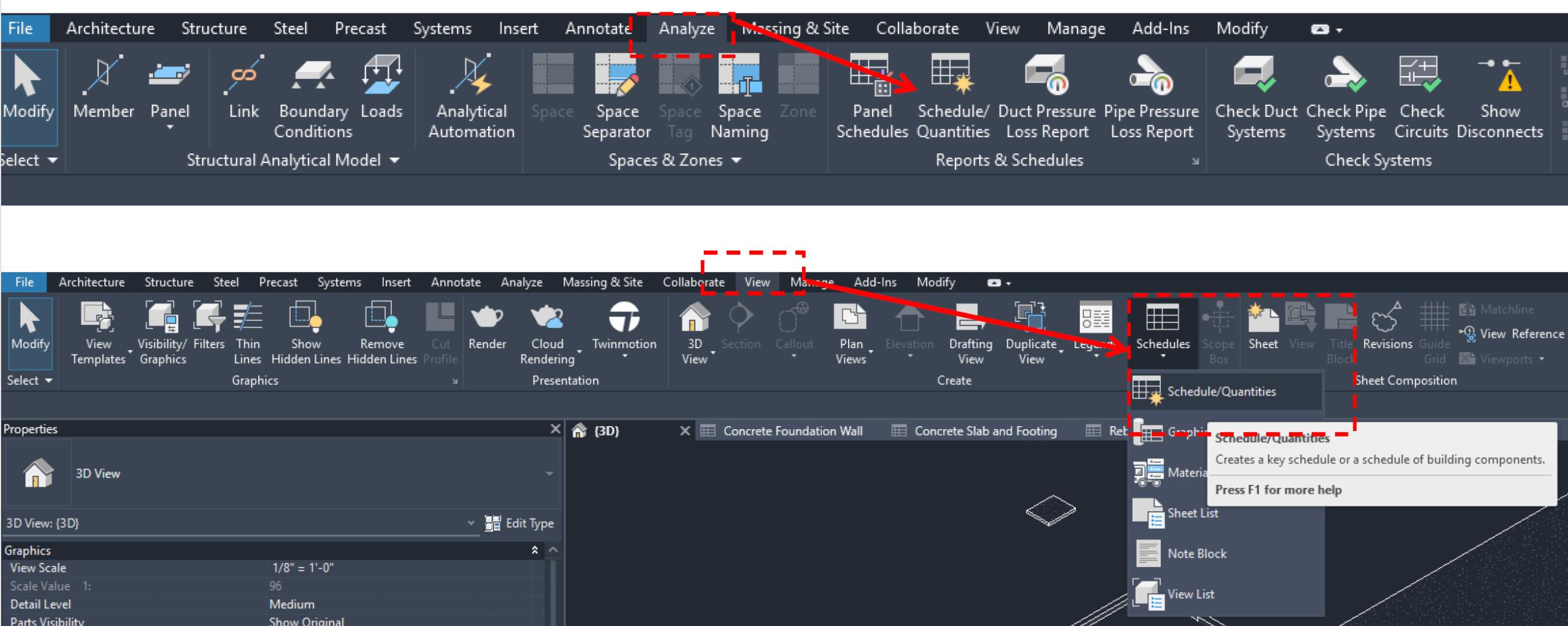
7. AI Data

The screenshot displays a BIM software interface showing a 3D model of a building and its associated properties.

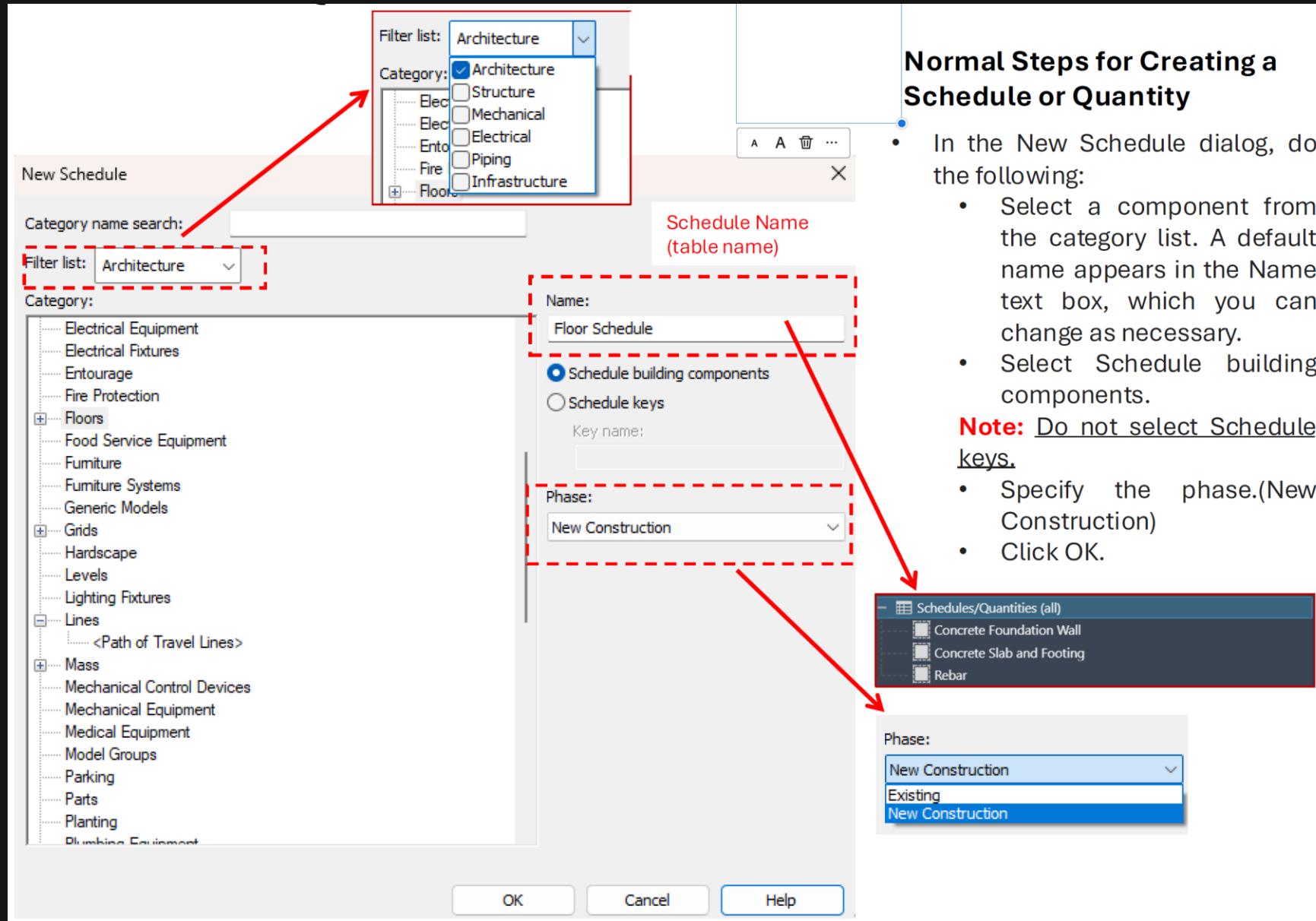
Properties Panel:

- Basic Wall:** Exterior_1-5 1/2"
- Finish Face: Interior:** First Floor and Ground Level
- Base Constraint:** -2' 0"
- Base Offset:** 0' 0"
- Base is Attached:** Unconnected
- Base Extension Distance:** 39' 3 1/2"
- Top Constraint:** Top Offset
- Unconnected Height:** 0' 0"
- Room Bounding:** Checked
- Related to Mass:** Unchecked
- Cross-Section Definition:** Cross-Section: Vertical
- Structural:** Structural Usage: Non-bearing
- Dimensions:**
 - Length:** 11' 2 1/2"
 - Area:** 422.94 SF
 - Volume:** 616.66 CF
- Identity Data:**
 - Image:** Unselected
 - Comments:** Unselected
 - Mark:** Unselected
- Phasing:**
 - Phase Created:** New Construction
 - Phase Demolished:** None
- IFC Parameters:** Export to IFC: By Type
- Export to IFC As:** IFC Predefined Type
- IfcGUID:** 0TtNtd09rAywY7\$pOLIJbl
- Data:**
 - AI_CostCode:** B1
 - AI_Description:** Masonry walls using concrete masonry units
 - AI_Rate:** 120.50000

8. Schedule/Quantities



8. Schedule/Quantities



Choose Wall

7. Schedule/Quantities

Schedule Properties

Fields Filter Sorting/Grouping Formatting Appearance

Select available fields from:
Walls

Parameter Name Search:

► Filter Available Fields

Available fields: 62 items

- Absorptance
- Angle From Vertical
- Area
- Assembly Code
- Assembly Description
- Assembly Name
- Base Constraint
- Base Offset
- Bottom Width
- Comments
- Cross-Section
- Default Exterior Angle
- Default Interior Angle
- Description
- Enable Angle Overrides
- Estimated Reinforcement Volume
- Excavation Volume on Toposolid
- Exchange Entity ID
- Exchange ID

Scheduled fields (in order):

- AI_CostCode
- AI_Description
- AI_Rate
- Family and Type
- Cost
- Count

Must be selected

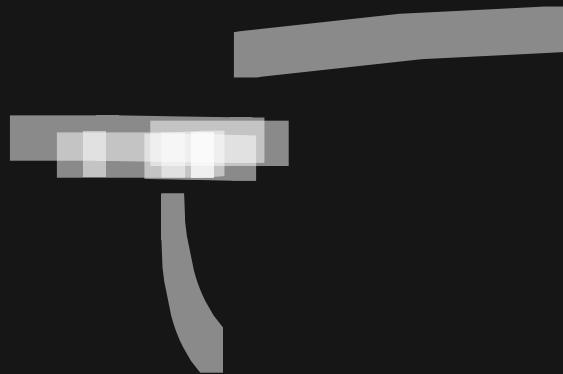
The rest can be customized using calculated parameters.

(Add Parameter).
(Remove Parameter).
(Move Up) or (Move Down).
(Combine Parameters).
(Edit Parameter).
(Delete Parameter).
(New Parameter)
(Calculated Parameter).

OK Cancel Help

9. Get the result

The screenshot shows the Autodesk Revit interface with the 'Wall Schedule' dialog box open. The dialog box contains a table with columns: A (AI_CostCode), B (AI_Description), C (AI_Rate), D (Cost), and E (Count). The table lists various wall types with their descriptions, rates, costs, and counts. The 'View Template' dropdown is set to '<None>'. The 'Edit Type' button is visible. The 'Project Browser - BIM_Model' window is also visible at the bottom left, showing a tree view of the project structure including 'Views (all)', 'Structural Plans', 'Reference', 'Floor Plans', and specific items like '00_Driveway and concrete pads', '00_Driveway with reinforcement', '00_Footing', and '00_Footing with reinforcement'. The top menu bar includes 'File', 'Architecture', 'Structure', 'Steel', 'Precast', 'Insert', 'Annotate', 'Massing & Site', 'Collaborate', 'View', 'Manage', 'Add-Ins', 'AI Estimator', 'Modify', and 'Modify Schedule/Quantities'. The status bar at the bottom right indicates 'Autodesk Revit 2025.2 - BIM_Model - Schedule: Wall Schedule'.



THANK YOU!

EXTEND & CUSTOMIZE

This add-in can be customized. Just extend the source code