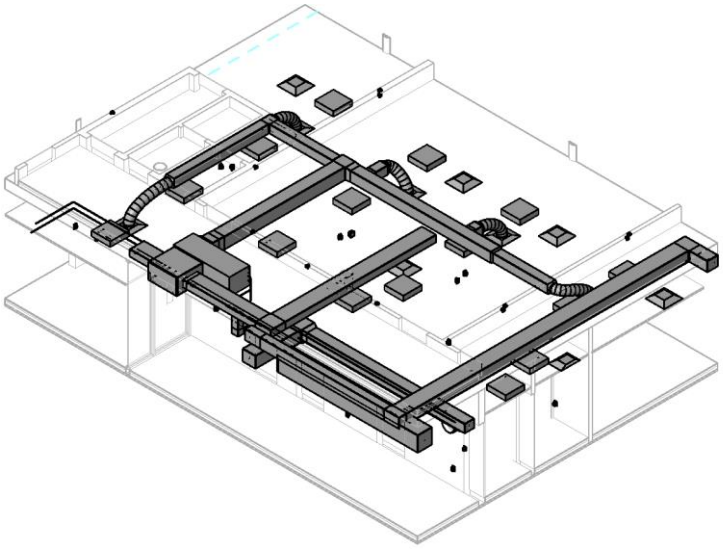


BIM Software Development

AutoRoute 1.14 Documentation

Prepared by Algoix

Document Revision 1 (Draft)
Prepared by Toste Wallmark (Algoix)

Algoix

Software Development Vendor

ACID
Advanced Construction Information Development

Project Client

Document Revisions

Date	Revision	Changes	Revised By
2018-03-17	**1**	**Initial revision**	**Algoix**

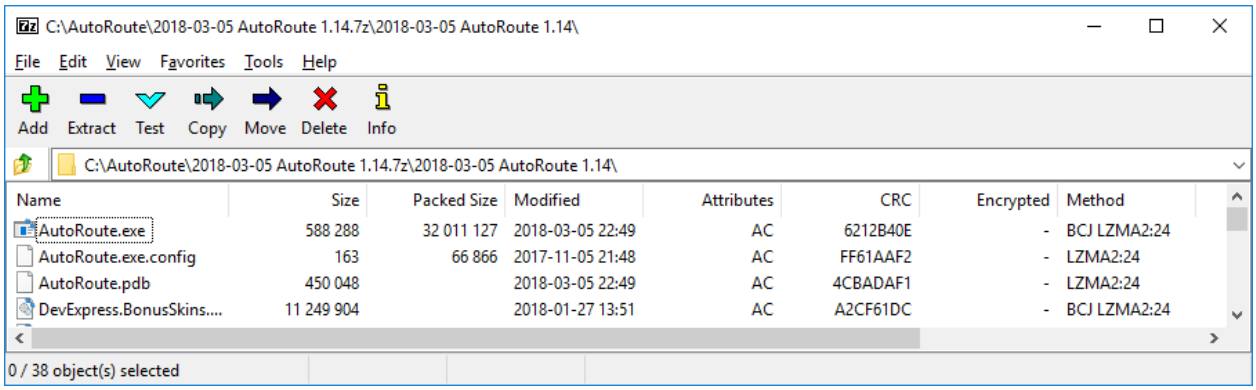
CONTENTS

INTRODUCTION

AutoRoute is a plug-in developed to run in Autodesk Revit. This guide provides instructions of usage and how the plug-in operates so that users can get the most out of the plug-in.

INSTALLATION

The plug-in is delivered as a 7-zip archive. The general file naming convention is Year-Month-Day AutoRoute Major.Minor.7z. For example, if the file name is 2018-03-05 AutoRoute 1.14.7z, it contains release 1.14 which was prepared on 5 of March 2018. Inside the archive, we can find the AutoRoute.exe which is the main executable, along with its dependencies. If we open the 7-Zip archive and inspect its contents, we should see a file list similar to the following.

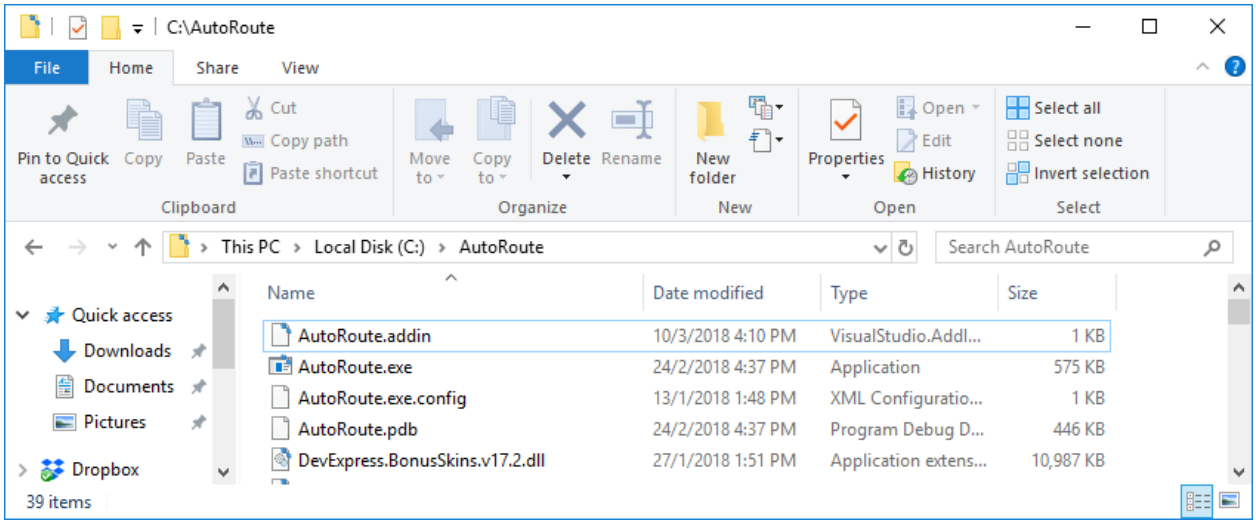


All of these files are required to be deployed on the user's workstation to start with Revit. It is necessary to deploy the files to a local drive, for example the C: drive for the plug-in to load correctly. Deploying to a network share is not supported and is not recommended as permission issues or network share outages can prevent loading.

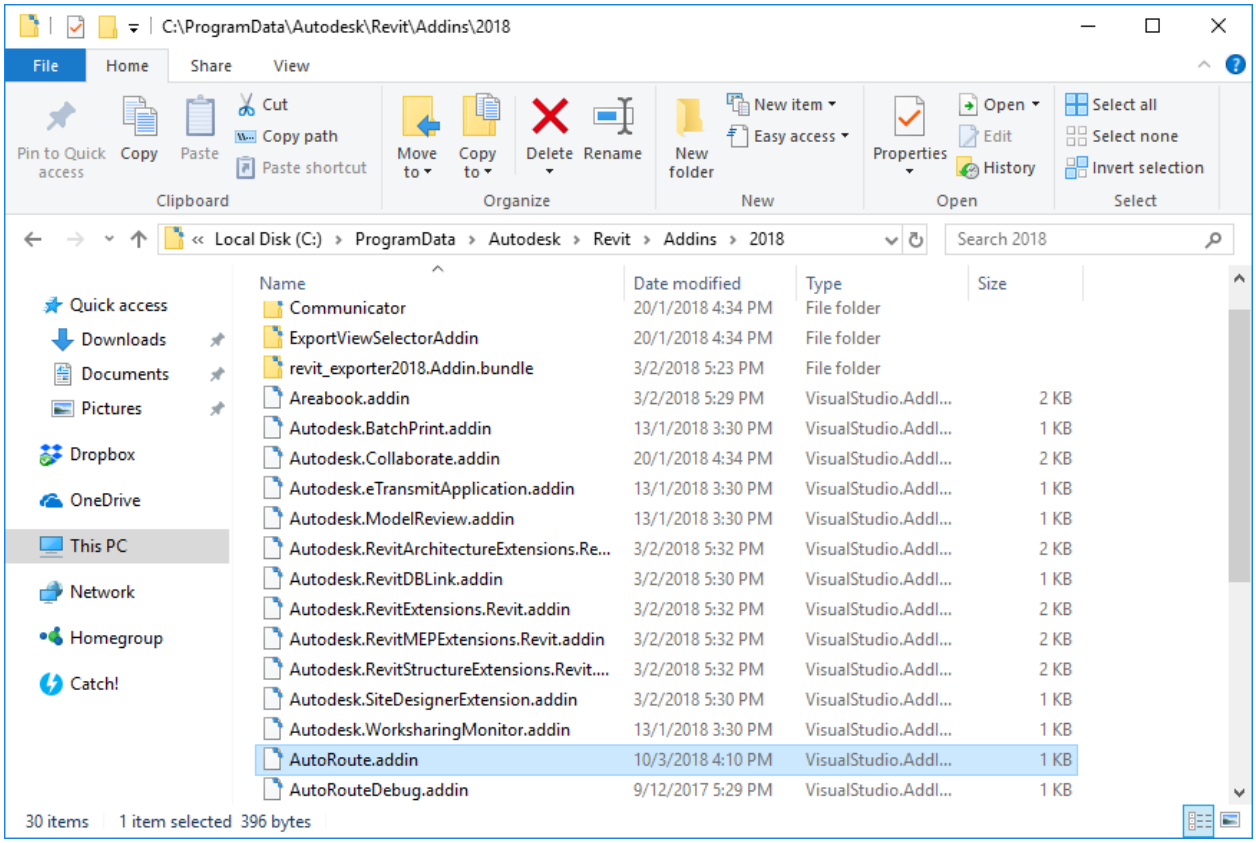
The plug-in can be installed to any local folder that the user has read and write permissions to. It is documentation, the simple path C:\AutoRoute is used. Users may alternatively choose a path which conforms to their environment's requirements, for example under Program Files.

1. If the plug-in has been loaded into Revit already, close all instances of Revit that has the plug-in loaded, as these instances can lock the files and prevent successful upgrade
2. Create the local destination folder C:\AutoRoute, if it does not exist
3. Delete any previously deployed files, if they exist, as they may otherwise interfere with the upgraded version

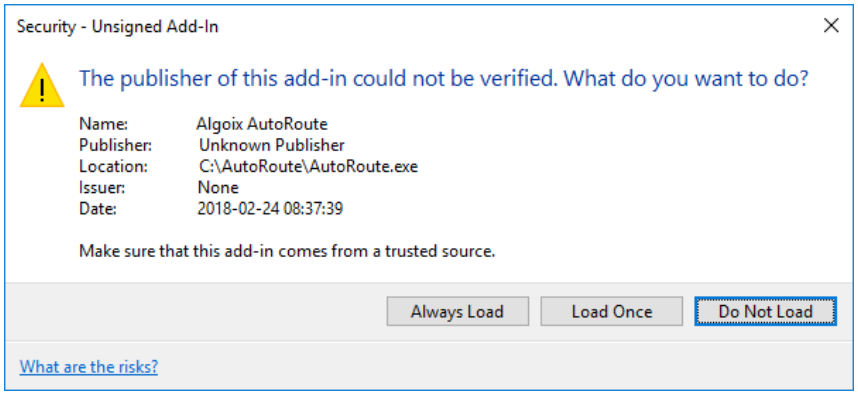
4. Extract all files in the 7-Zip archive to the destination folder. When using C:\AutoRoute as the destination folder, the main executable will be located in C:\AutoRoute\AutoRoute.exe



5. Copy the AutoRoute.addin file to the add-ins folder for your Revit installation. For Revit 2018, the folder is: **C:\ProgramData\Autodesk\Revit\Addins\2018**

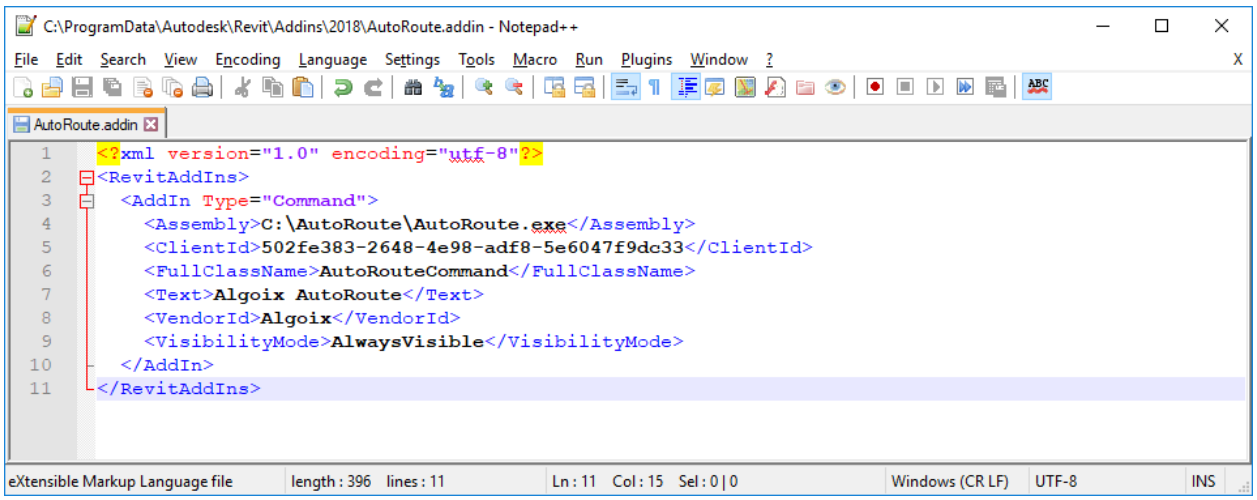


6. If you have installed AutoRoute in a location that is not C:\AutoRoute, open the AutoRoute.addin file and modify the path within the <Assembly> element and re-save the file. This also needs to be done if the plug-in is moved
7. Restart Revit. You may now see the following prompt. In case this window appears, verify that it indicates Algoix AutoRoute, and click “Always Load”. This alert will be remembered as long as the ClientId is unchanged in the addin file. If the ClientId is changed, the alert will pop up again



8. If user clicked Always Load in the previous dialog, there will no longer be any warning prompt tiled "Security – Unsigned Add-In" the next time Revit starts
9. It is not necessary to restart Windows after the installation

For reference, here are the default contents of the AutoRoute.addin file, as displayed in the application Notepad++:



To prevent the plug-in from loading, delete the AutoRoute.addin file created in the step above.

To uninstall the plug-in, delete the AutoRoute.addin file created in the step above, and delete the folder where AutoRoute is extracted, for example C:\AutoRoute.

To upgrade the plug-in, delete the deployed files in the C:\AutoRoute folder and extract the updated files there so that the extracted executable is located in C:\AutoRoute\AutoRoute.exe. It is normally not necessary to modify the AutoRoute.addin file when upgrading the plug-in.

Plug-in Security Settings

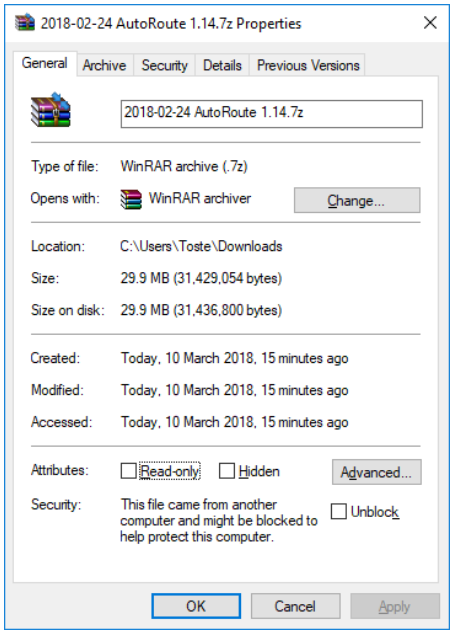
The plug-in is loaded into the Revit executable process and requires permissions to run. If the plug-in fails to start, the following are some common issues that may occur and possible solutions.

Troubleshooting guide: Unblocking the executable

When files are extracted from an archive which has been downloaded from the Internet, the archive is tagged in the Windows file system which may block its execution. This is a security feature of Windows that can help

to prevent unintended execution of downloaded code. However, it may also block the execution of the AutoRoute plug-in.

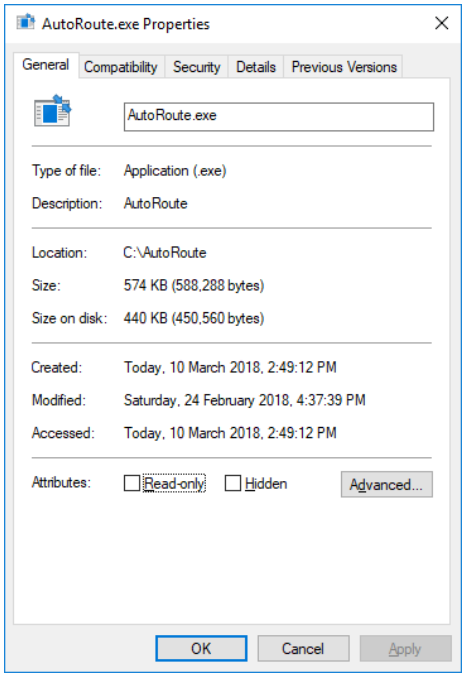
When an archive is downloaded from the Internet using a web browser, Windows typically adds a security attribute to the file for security reasons. Below is an illustration of how this looks like for the AutoRoute 1.14 archive. Please note the text under the "Security" heading in the bottom section:



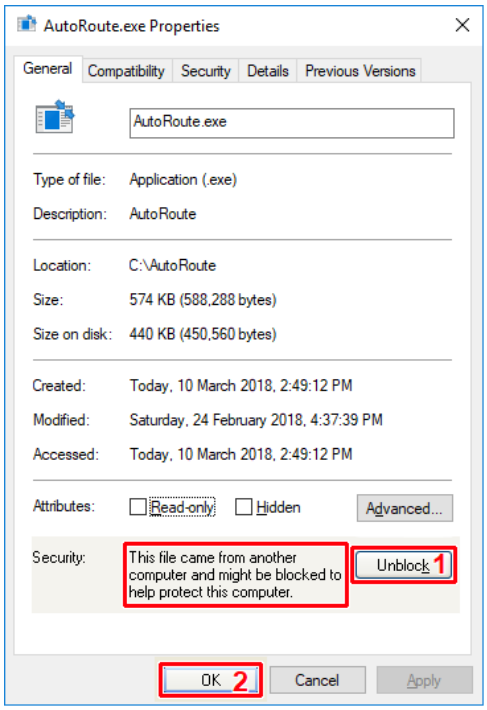
Recent versions of AutoRoute, including release version 1.14, have been prepared using the 7-Zip[1] archiving utility. As of 2018-03-10, 7-Zip remains free software with open source that can be used in a commercial organization without registration or payment. For more details and information on how to use 7-Zip, please refer to the 7-Zip web site. With regards to archive permissions, 7-Zip has a different behaviour when it comes to the blocking attribute. If a file is prepared using conventional zip archives and it is downloaded from the internet and subsequently extracted, the block attribute on the zip file will propagate to the extracted files, and all extracted executables will also be blocked as a consequence. However, with 7-Zip, the blocking flag is not set on the extracted files even when the originally downloaded archive is blocked. Hence, after the migration to use 7-Zip, the permission issue should not occur. Please note that this behaviour was observed in the last released non-beta version 18.01, and may be changed in future releases of 7-Zip by the 7-Zip vendor. We currently rely on this behaviour to prevent the blocking prompt from appearing, but if it still appears, please use the following guideline to resolve the issue.

To diagnose this issue, when the archive is extracted, users can right-click the extracted executable and inspect the "General" tab, and if the button Unblock is visible, the executable is blocked by Windows. This is how the screen shot would look like if the plug-in executable is not blocked (note that details may vary, such as file size, date and path):

[1] http://www.7-zip.org/



This is how the screen shot would look like if the file was blocked. This is an illustration and the details may vary. To resolve this, the user needs to first click Unblock and then click OK.



After clicking OK in the dialog above, close and re-launch Revit for the permission change to take effect.

Troubleshooting Guide: Resolve Revit plug-in loading warnings

When Revit launches, there will be an additional permission check on the loaded executables before they are loaded. This may prevent the plug-in from being loaded. This issue appears to be common and is the expected behaviour for deployments where files are copied directly and not added to the file system by an installer package. It is therefore recommended to set Revit to load plug-ins without warnings. This is a one-time manual

change that needs to be done for each major Revit version as they maintain a separate configuration file with the relevant settings.

The following guideline has been tested and verified to work with Revit 2018. It should generally also work on other versions as the start-up permission check is similar. For further details, please refer to Autodesk support forums.

To resolve the issue:

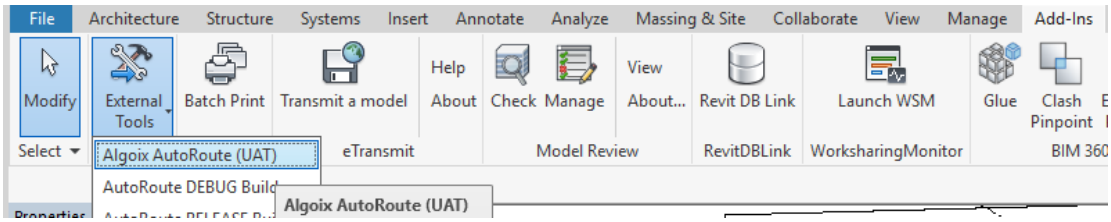
1. For Revit 2018, please open the following configuration file: C:\Program Files\Autodesk\Revit 2018\Revit.exe.config. The Revit configuration is formatted in XML.
   It is recommended to always make a back-up copy of the file first before modification, so that the original settings can be restored at any time.

2. Under the <runtime> element, add the following tag:
   `<loadFromRemoteSources enabled="true"/>`
   This is an example of how the Revit.exe.config file should look like after adding the line

```
21      </settings>
22    </system.net>
23    <runtime>
24      <loadFromRemoteSources enabled="true"/>
25      <assemblyBinding xmlns="urn:schemas-microsoft-com:asm.v1">
26        <probing privatePath="SDA\bin"/>
```

   The configuration file should already have a <runtime> element after a clean installation. The additional element is inserted as a child element. To clarify the terminology in XML, an XML "element" is all contents between angle brackets < and >. Elements can contain other elements, like this: <element><other/><element/>, or just be stand-alone as in <element/>. Note that if an element has no ending, it needs an ending slash. If the XML is not formatted correctly, it may cause issues with Revit functionality or start-up.

3. Save the file.
4. Close Revit if open
5. Relaunch Revit
6. The plug-in and any subsequent updates to it should now load.
7. Load a Revit file. The plug-in should be visible under Add-Ins > External Tools with the title Algoix AutoRoute. The screen shot below is an illustration.



Reference links with more details are available in footnotes[2][3].

[2] http://thebuildingcoder.typepad.com/blog/2016/04/windows-10-security-blocks-external-command.html

[3] http://help.autodesk.com/view/RVT/2018/ESP/?guid=GUID-8EB25D2A-3CAF-486A-BA8E-C2BEF3DB68F6

METHODOLOGY

In this version, AutoRoute provides functionality that allows designers to analyse the space utilization when building services elements are placed, and review the results at a glance in the plan view. This allows for immediate identification of available space, and identification of congested area. It also allows for identifying the degree of congestion and amount of free space available. This representation is referred to as a **height map**. It is a map, in the sense that it is a **2D representation**, and it represents height, in the sense of **available headroom height**. Hence it constitutes a **2D representation of available headroom height**.

The height map is generated by drawing filled regions on the plan view. This means that the regions will follow the view when the user navigates the views, and users can modify and delete the filled regions after regeneration just like any other Revit element. The filled regions are not automatically synchronized with the model, so once the model is updated, the user needs to remove any existing filled region that would overlap, and re-generate the height map for an up to date representation.

REQUIRED MODEL INPUT

AutoRoute The plug-in requires certain conventions to be followed in the model. This requires attention and verification by the modeller to ensure that the model is accurate. This includes, but is not limited to the following:

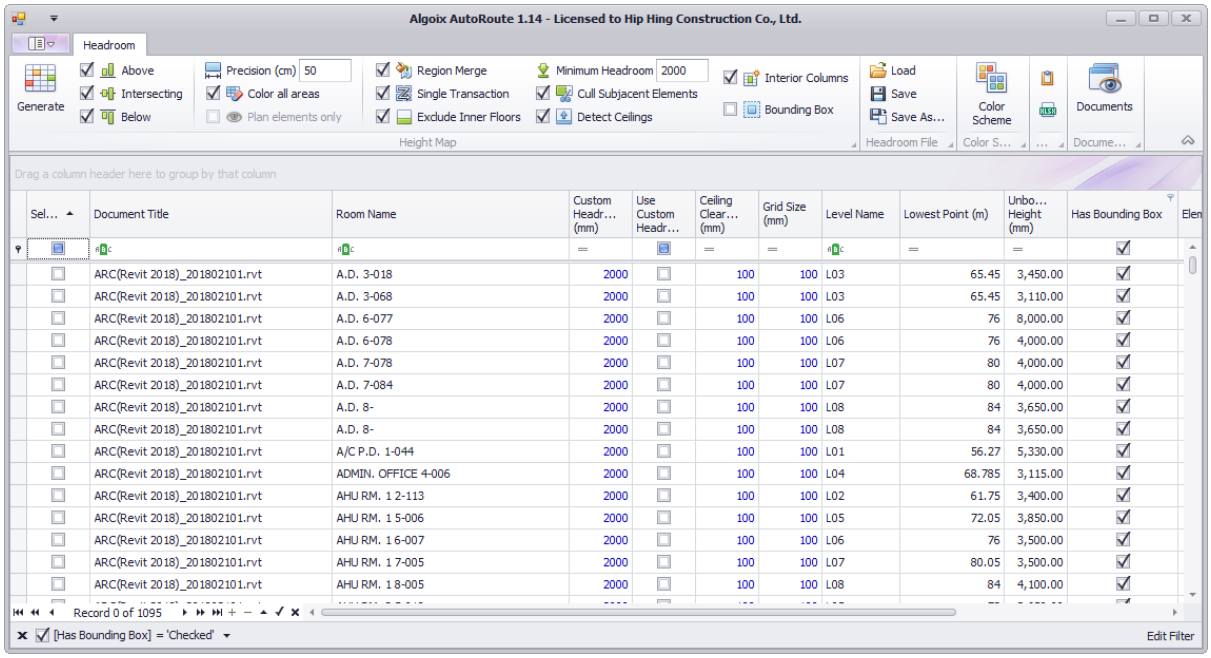
- Every space that is going to be analysed must be defined using a valid room element. If there is no room element, no analysis can be done.
- The room element will be tessellated and subdivided according to the grid precision configured. Hence the geometry across the entire room element must be correct to ensure correct output.
- Room bounding setting must be correctly defined on all room bounding elements, so that the room extends correctly in all dimensions. For instance, if a modeller has two layers of floor slabs, and forgets to set the upper slab to be room bounding, the room element will include the slab as if it was inside the space, and exclude it from the headroom calculation, which would as a result tend to over-estimate the amount of headroom available and yield an overly optimistic result.
- Correct Revit element types need to be selected for the plug-in to include them. For example, when drawing a duct, use the built-in duct element and not a plain mass element.
- Correct color settings need to be used. If users need to detect congestion within a particular range, ensure that there are appropriate color settings defined within that range, so that thresholds can be viewed.

MEASUREMENT TECHNIQUE

The plug-in measures the headroom using the geometry of the room element boundary. The room boundary is defined by the Revit room element. It is therefore important to ensure that the room boundary is accurately represented in the model, or else the result of the plug-in may not be as accurate as required.

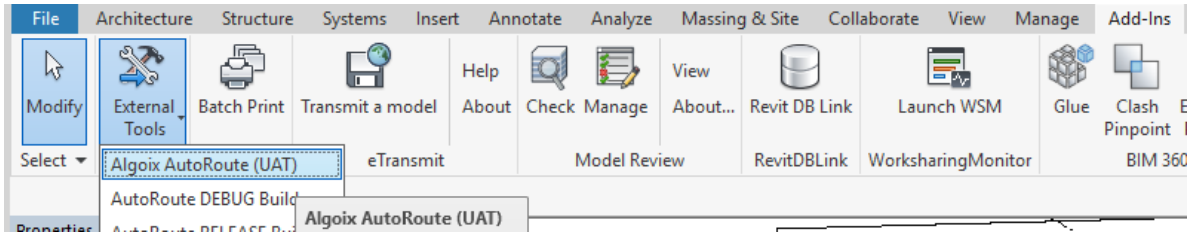
USER INTERFACE OF AUTOROUTE

Below is the sample of the user interface of AutoRoute. There are numerous options for the users to choose. Users can check it one by one and see if any one of them is useful. It will show all the existing rooms in the Revit project and some important information of the rooms such as the floor level, lowest point of a room and the unbounded height. Besides, the users can customize the headroom for each room, ceiling clearance and grid size if needed.



BEGINNING

When the users have opened the Revit project, there will be an external add-in found in the project. Users may follow the below steps to start the AutoRoute.

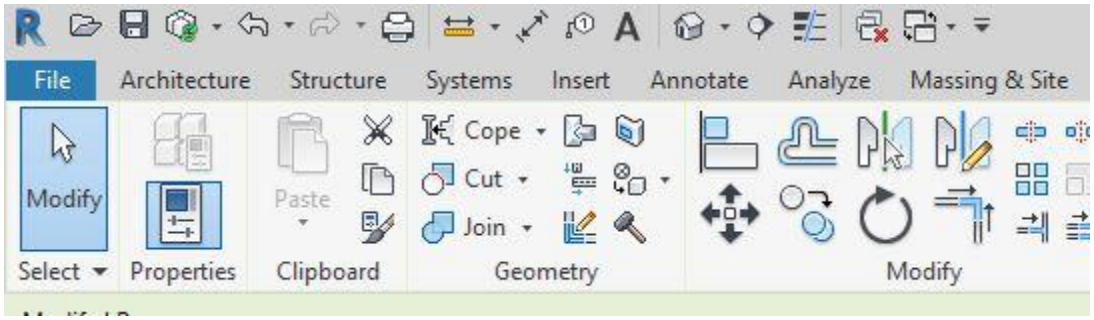


Users can click in the tab “Add-Ins”. It will list different functions provided by the tap “Add-Ins”. For example, “Modify”, “External Tools”, “Batch Print”, “Transmit a model”, etc. Once users find the icon “External Tools”, they should click on it. There will be a drop-down list showing all the existing add-ins in the project. Users should click on the option “Algoix AutoRoute (UAT)” to start the program. A user interface of AutoRoute will show.

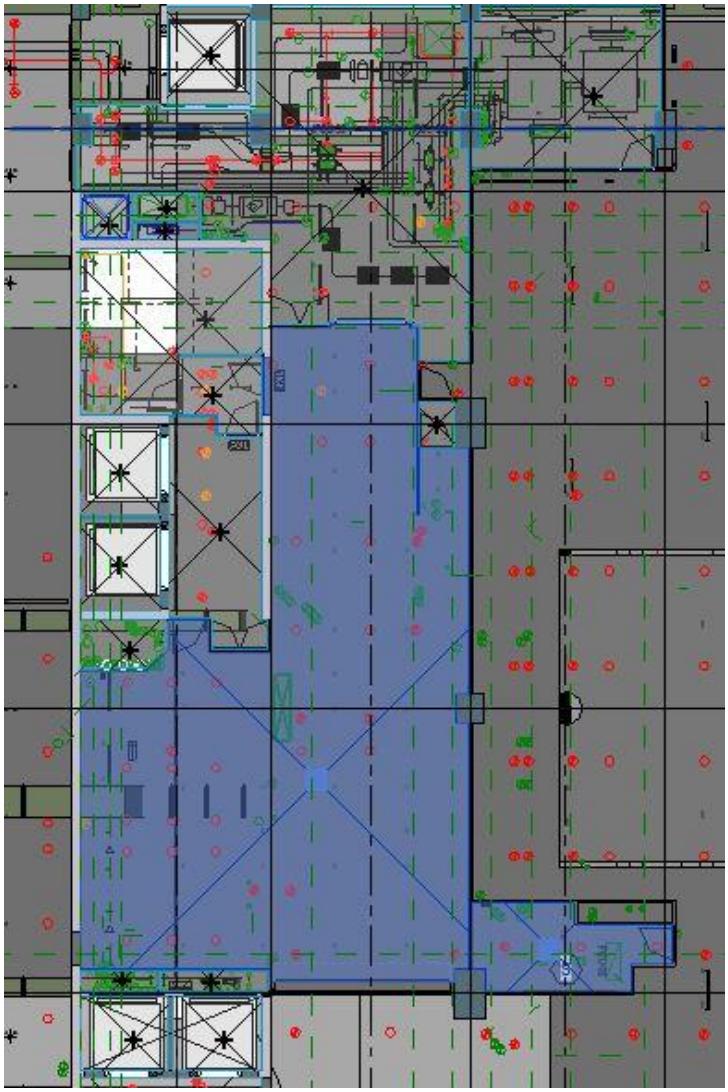
GENERATION OF COLORED REGIONS OVER ROOM(S)

Once the users get familiar with the color scheme, they may generate colored grid over the room(s) selected.

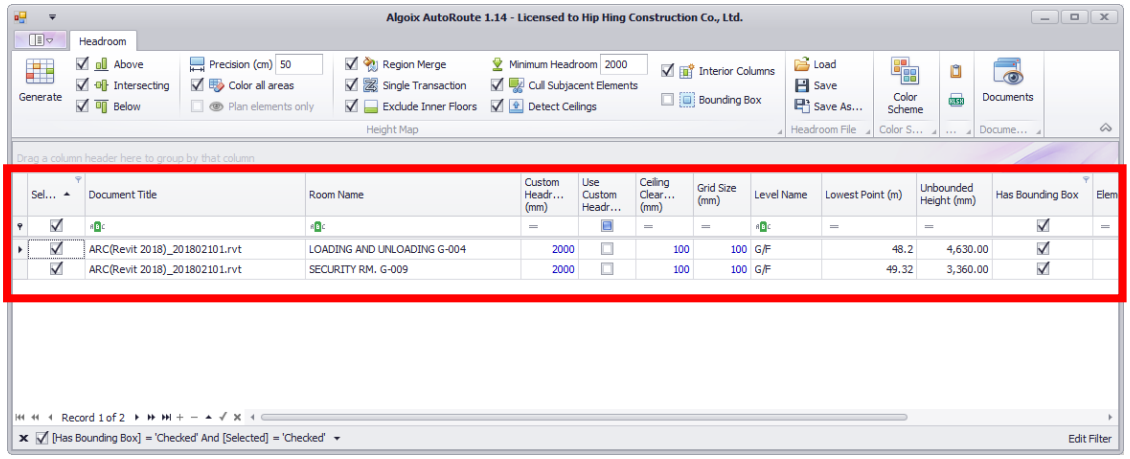
Users first click on the icon “Modify”.



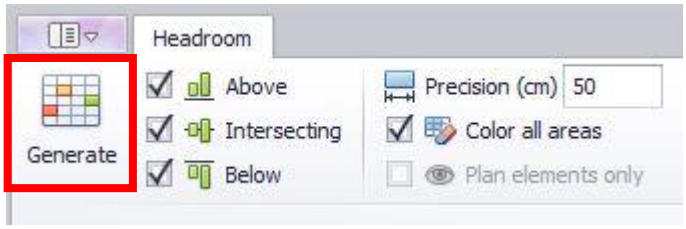
Then, they may select one or multiple rooms to be inspected like below.



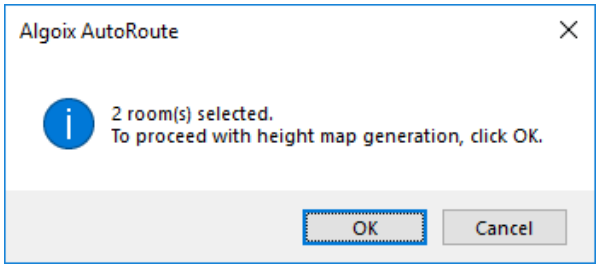
After that, they can click of the tap “Add-Ins”, open the drop-down list of “External Tools” and choose the “AutoRoute” add-in. In this sense, the AutoRoute will automatically detect which room(s) are selected and show them in the room list.



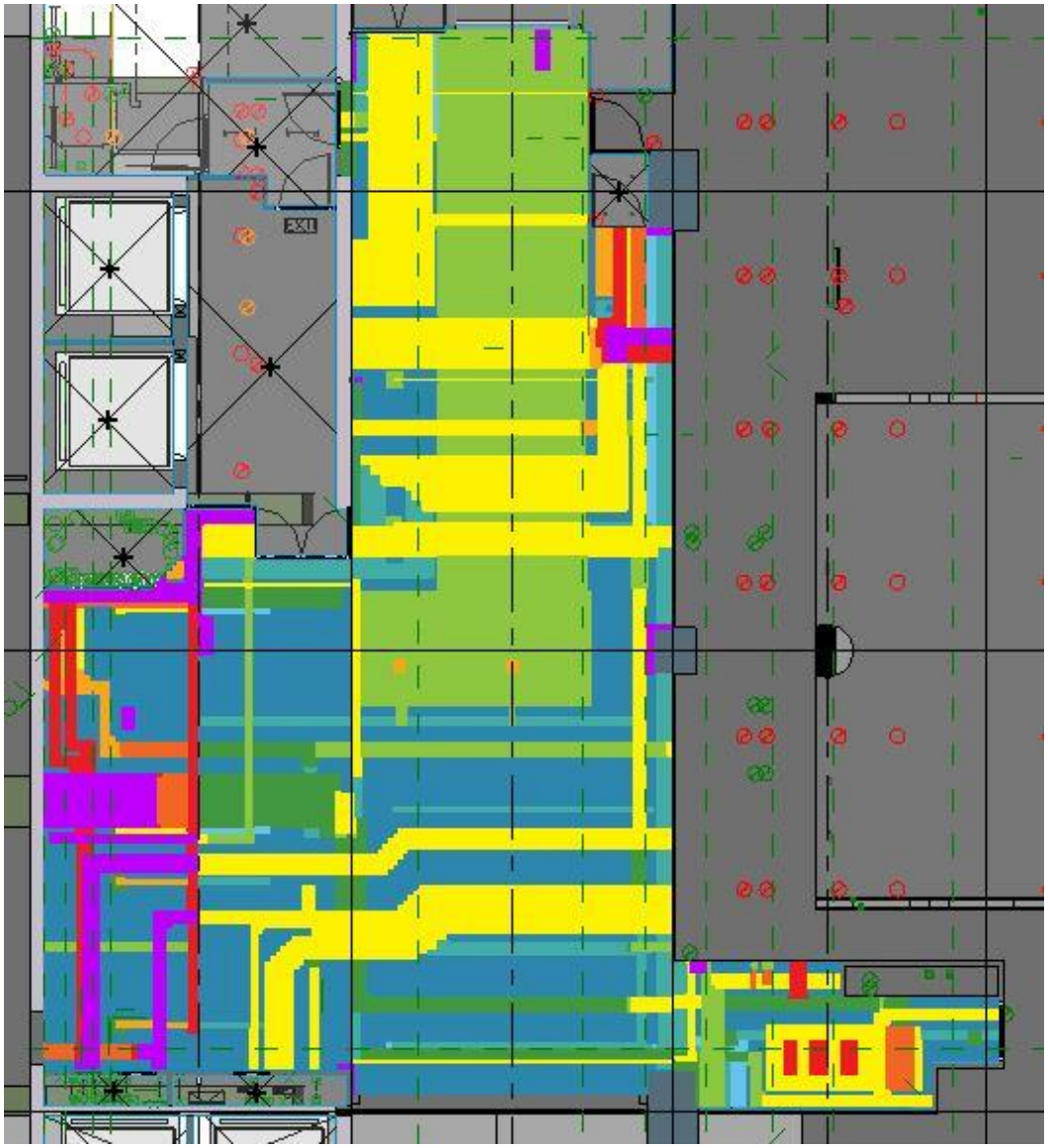
Users then can click on the button “Generate”.



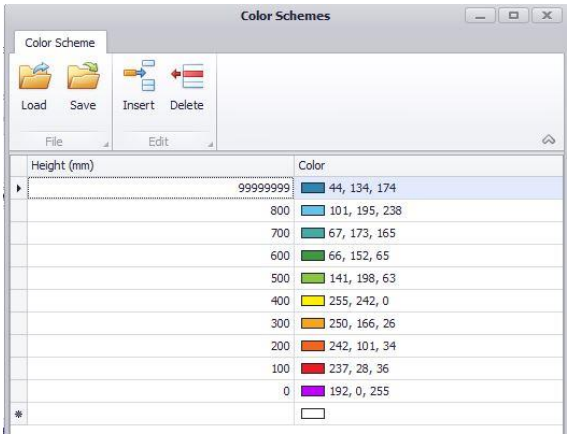
And then, a message prompt will show to ask if the users want to proceed with the rooms selected.



After clicking “ok”, AutoRoute will start to analyze the rooms. After a few seconds, the result will show of the projects.

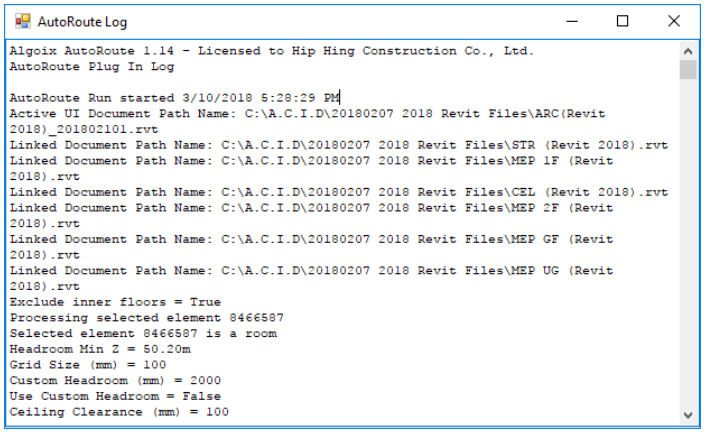


For the above result, the purple color refers to the most congested area in a room. The less congested areas will be the yellow, green and blue colors. Users should refer to the implemented color scheme defined in the AutoRoute’s user interface shown as below.

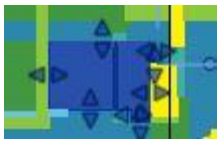


Height (mm)	Color
99999999	44, 134, 174
800	101, 195, 238
700	67, 173, 165
600	66, 152, 65
500	141, 198, 63
400	255, 242, 0
300	250, 166, 26
200	242, 101, 34
100	237, 28, 36
0	192, 0, 255

At the same time, an AutoRoute log showing the details of the elements found within a room will prompt. Users can check the details of the result for the enquiries.



To find the intersecting elements, click a filled region and check its comment property.



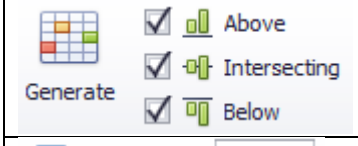
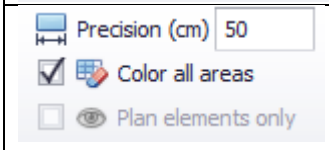
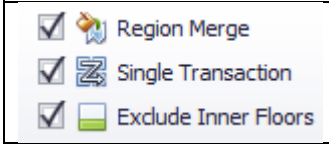
The Properties > Identity Data > Comments shows the following:

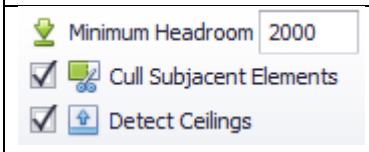
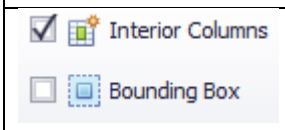
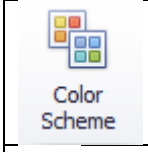
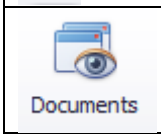
> *600.00 mm @ Id=3421374, "DN50" of "PD_CI Fitting_Anti-siphon P-trap DN50" (Pipe Accessories), Box=Min=[X=9.43m, Y=21.91m, Z=52.35m], Max=[X=9.62m, Y=21.96m, Z=52.59m] in MEP GF (Revit 2018).rvt*

This tool tip means that the element **"DN50" of "PD_CI Fitting_Anti-siphon P-trap DN50" (Pipe Accessories)** is **at least** 600mm or above the headroom level.

MAIN FUNCTIONALITY IN TOOLBAR

AutoRoute is a plug-in developed to run in Autodesk Revit. This guide provides instructions of usage and how the plug-in operates so that users can get the most out of the plug-in.

Toolbar Group	Functionality
	The **Generate** button will run height map generation for the selected rooms. One or more room should be selected first. By default, the generation will consider elements **above**, **intersecting** and **below** the headroom level. To exclude elements, uncheck the check boxes.
	The **Precision** setting will determine when AutoRoute switches from bounding box detection to tessellation detection. Tessellation is slower but more accurate. The measurement in cm refers to the size threshold when the plug-in switches to tessellation. The setting to **Color all areas** can be disabled if the user does not wish to color areas that have no intersecting elements. The Plan elements only feature will filter elements that are not visible in the plan. This feature is only available when a single document is loaded and is not supported for linked files.
	**Region Merge** will merge the filled regions into a single region if they have the same minimum height. This will significantly speed up the creation of the filled region. One use case for disabling region merging is to see tool tips for each individual tested cell that shows the lowest element in that

	cell. **Single transaction** will combine all rooms into one transaction, so that users can undo all rooms at once. If this is deselected, rooms are committed individually and can/have to be undone manually. **Exclude inner floors** will exclude floor elements that are inside the space, which happens when the room object is inside the space. This occurs when a slab element is modelled inside the room and the room element includes this part.
	**Minimum headroom** defaults to 2000 mm and is used as the default for newly detected rooms**. Cull subjacent elements** will remove certain types of elements that should not be part of intersection testing that are below the minimum headroom. This currently includes the category plumbing fixtures. Note that elements above the minimum headroom but below the headroom setting for the room are still included. The **Detect Ceilings** setting will adjust the headroom to a lower value if a ceiling element is detected. Ceiling detection works on each individual cell, and will use the lowest value out of any ceiling in the cell, and the room's headroom setting. For example, if the room's default headroom setting is 2500 mm and a ceiling is at 2300 mm, the ceiling will be used as the constraint.
	**Interior columns** are columns that form voids inside the space, but are not part of the space area. This typically includes structural columns in large rooms. If Interior Columns is enabled, AutoRoute will draw filled regions to indicate that these areas are below the headroom requirement. If this is unchecked, no filled regions will be drawn on these interior columns, because they are outside the room element. **Bounding Box** prevents using any tessellated results and uses only bounding box data to test. While this can be faster, it may also lead to overly pessimistic results and is not the default setting. However, if all surfaces of all elements are axis aligned, that is, neither sloping nor forming curved shapes, there should not be any loss of precision from using this setting.
	The **Load** / **Save** / **Save** As settings allows the user to load and save headroom settings for the model. These settings are maintained externally in an XML file, and when loaded, they are mapped onto the rooms in the model.
	The **Color Scheme** feature allows users to save or load color schemes. AutoRoute offers a default color scheme which is automatically used the first time the plug in is used. The current setting is automatically restored the next time AutoRoute is opened.
	The **report designer and Excel export** functions allows users to design and create reports from the room data, as well as exporting the raw data from the table to Excel for further processing.
	The **Documents** window lists the loaded model. This function is mainly for diagnostics, to inspect which models that AutoRoute has detected as being loaded.

DEFAULT SETTING OF TABLE GRID AND COLOR SCHEME

Users can change the default setting of table grid and color scheme to the desired one. The default setting can be changed in the local files, located inside a folder of **%AppData%\Algoix\AutoRoute.** Users can search the link **%AppData%\Algoix\AutoRoute**. Two files are located inside.

- Frist, it is the colorScheme.xml
- Second, it is the GridLayout.xml

For the colorScheme.xml, it looks like below.

```
<?xml version="1.0" encoding="utf-8"?>
<AutoRouteColorSchemes xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <ColorSchemes>
    <ColorScheme Height="99999999" Alpha="255" Color="2C86AE" />
    <ColorScheme Height="800" Alpha="255" Color="65C3EE" />
    <ColorScheme Height="700" Alpha="255" Color="43ADA5" />
    <ColorScheme Height="600" Alpha="255" Color="429841" />
    <ColorScheme Height="500" Alpha="255" Color="8DC63F" />
    <ColorScheme Height="400" Alpha="255" Color="FFF200" />
    <ColorScheme Height="300" Alpha="255" Color="FAA61A" />
    <ColorScheme Height="200" Alpha="255" Color="F26522" />
    <ColorScheme Height="100" Alpha="255" Color="ED1C24" />
    <ColorScheme Height="0" Alpha="255" Color="C000FF" />
  </ColorSchemes>
</AutoRouteColorSchemes>
```

For each color, it represents the range of available space height between the ceiling/headroom and the upper floor. In the above, there are 10 different colors which represent different ranges. Users can change the color code to set it as the preferred one. And they can delete the existing range(s) or insert more ranges into the color scheme. It is fully controlled by the users.

For the GridLayout.xml, it shows the details of each room found in the project in the table format. The design of the table format is defined in the GridLayout.xml. Users are able to change it as the desired one. The table is like below.

Sel...	Document Title	Room Name	Custom Headr... (mm)	Use Custom Headr...	Ceiling Clear... (mm)	Grid Size (mm)	Level Name	Lowest Point (m)	Unbounded Height (mm)	Has Bounding Box	Elem
☑	ABC	ABC	—	☐	—	—	ABC	—	—	☑	—
☑	ARC(Revit 2018)_201802101.rvt	LOADING AND UNLOADING G-004	2000	☐	100	100	G/F	48.2	4,630.00	☑	
☑	ARC(Revit 2018)_201802101.rvt	SECURITY RM. G-009	2000	☐	100	100	G/F	49.32	3,360.00	☑	

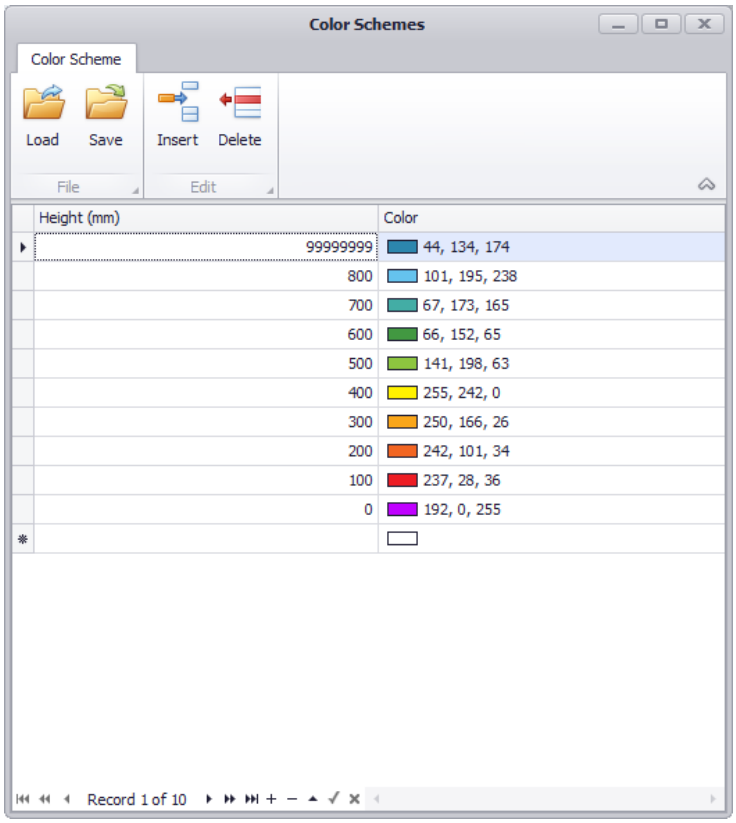
The xml file is like below. Users can make changes to the xml file to re-define the structure of the table but it is recommended to let AutoRoute manage it automatically.

```
<property name="ViewCaptionHeight">-1</property>
<property name="Columns" iskey="true" value="26">
  <property name="Item1" isnull="true" iskey="true">
    <property name="Name">SelectedColumn</property>
    <property name="VisibleIndex">0</property>
    <property name="Visible">true</property>
    <property name="Width">60</property>
  </property>
  <property name="Item2" isnull="true" iskey="true">
    <property name="Name">DocumentTitleColumn</property>
    <property name="VisibleIndex">1</property>
    <property name="Visible">true</property>
    <property name="Width">230</property>
  </property>
  <property name="Item3" isnull="true" iskey="true">
    <property name="Name">RoomNameColumn</property>
    <property name="VisibleIndex">2</property>
    <property name="Visible">true</property>
    <property name="Width">225</property>
  </property>
  <property name="Item4" isnull="true" iskey="true">
    <property name="Name">CustomHeadroomMillimetersColumn</property>
    <property name="VisibleIndex">3</property>
    <property name="Visible">true</property>
    <property name="Width">60</property>
  </property>
  <property name="Item5" isnull="true" iskey="true">
    <property name="Name">UseCustomHeadroomColumn</property>
    <property name="VisibleIndex">4</property>
    <property name="Visible">true</property>
    <property name="Width">60</property>
  </property>
  <property name="Item6" isnull="true" iskey="true">
    <property name="Name">CeilingClearanceMillimetersColumn</property>
    <property name="VisibleIndex">5</property>
    <property name="Visible">true</property>
    <property name="Width">60</property>
  </property>
```

COLOR SCHEME

Once the user interface opens, users can check what the color of the grid represents. Users can click on the color scheme in the tool bar.

After clicking on it, a detail of color scheme will prompt. Users can edit the color scheme according to their preference. Below is the default one. The field "Height" means the maximum height of available space between the ceiling/headroom and the upper floor.



Each color represents different height of the available space between the ceiling/headroom and the upper floor in a room.

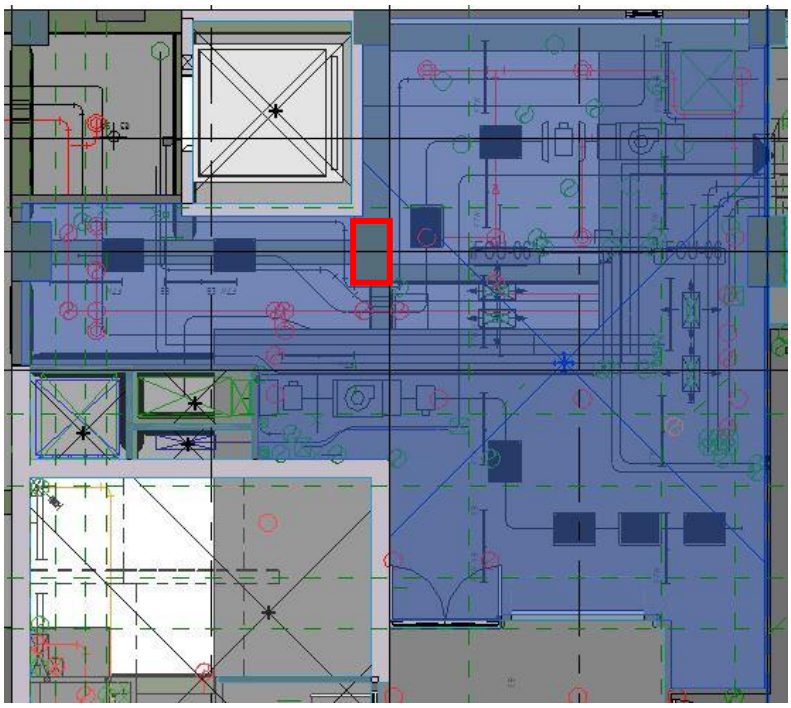
- From 0mm to 100mm, the grid will have purple color.
- From 100mm to 200mm. the grid will have red color.
- From 200mm to 300, the grid will have deep orange color, etc.

Users also can insert or delete some color according to their needs. As told before, the default color setting can be changed in the xml file located at **%AppData%\Algoix\AutoRoute\colorScheme.xml.**

INTERIOR COLUMN

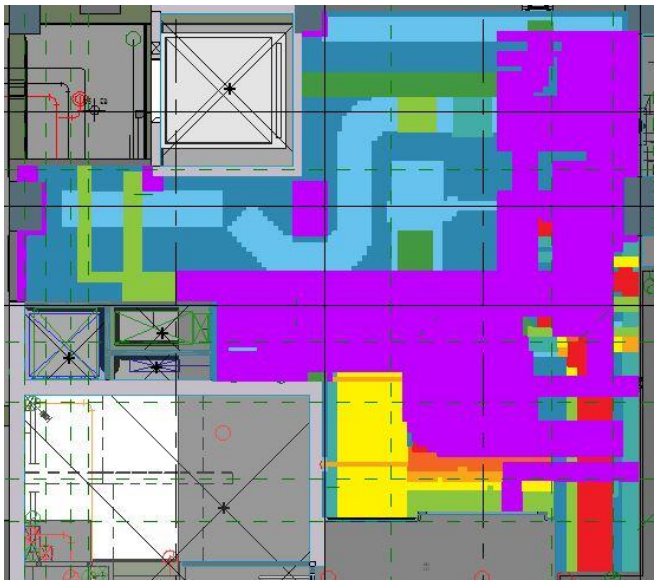
It is important to know that there may be some interior columns in a room. The design of each room differs from that of one another.

Here is one example. The rectangle in red color is an interior column.

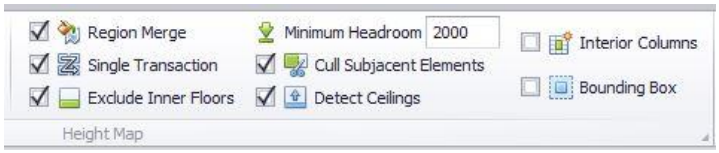


If the users check the box of “Interior Columns”, it will analyze the interior column(s) as well.

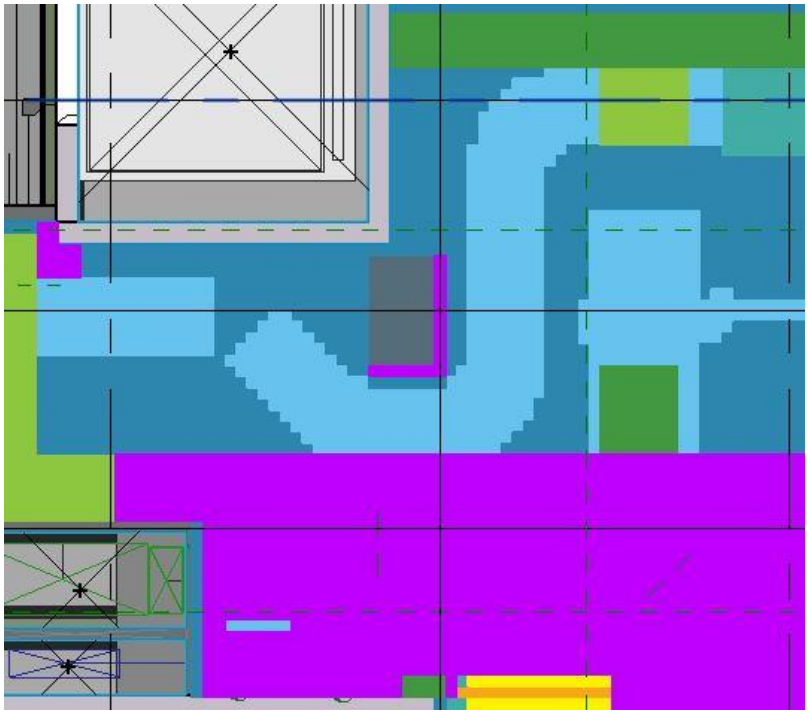
The result will be like that. The interior column will be in purple because there are absolutely no available space between the headroom/ceiling and the upper floor.



If the users do not check the box of “Interior Columns”, the interior columns within a room will not be analyzed.



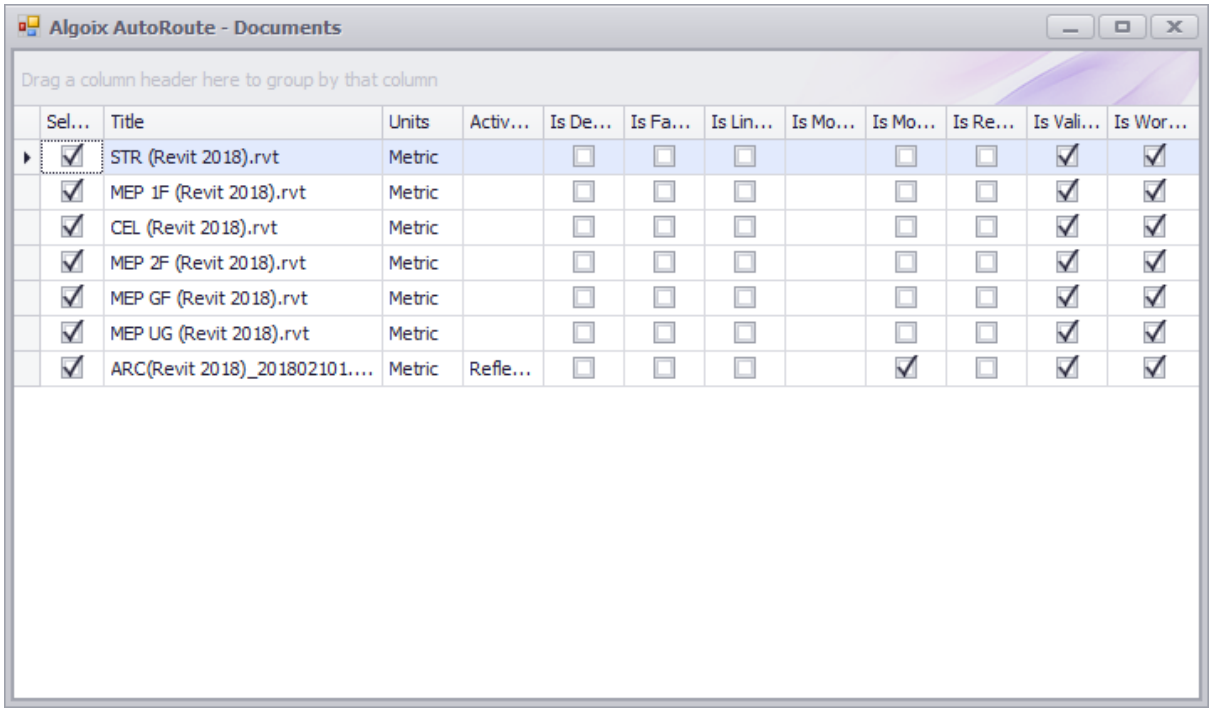
The result will be like that. AutoRoute will not generate any colored region over the interior columns and it remains grey as before the running of AutoRoute.



DOCUMENTS

Also, users can read what document(s) the project includes.

Once the users click on the button “Documents”, a “Documents” panel will show to let the users know which documents are included in the project.

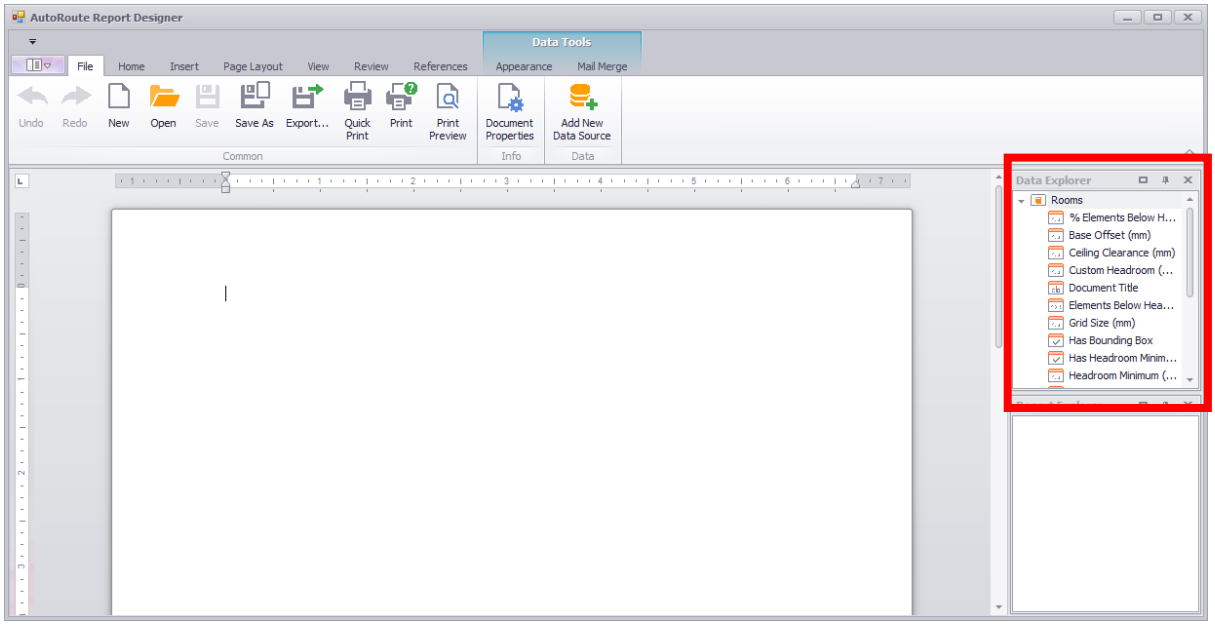


In the above example, ARC, MEP, CEL and STR Revit documents are included in the projects so that the users can know if there are any missing documents.

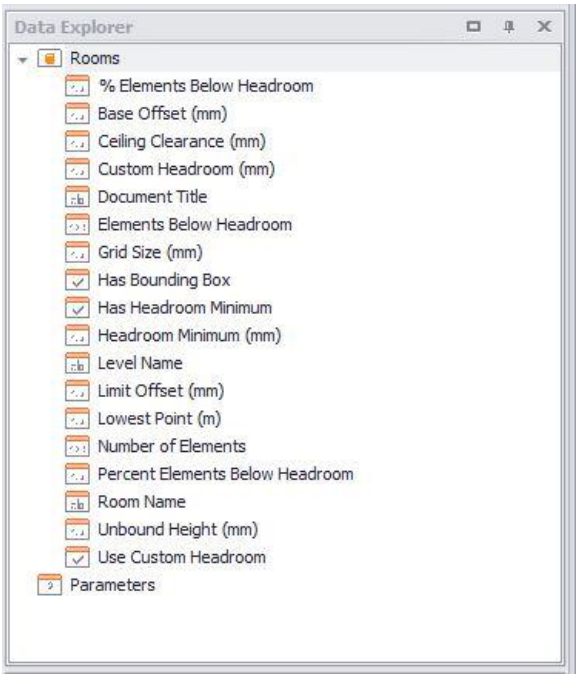
REPORT

Also, users can generate a report of the AutoRoute.

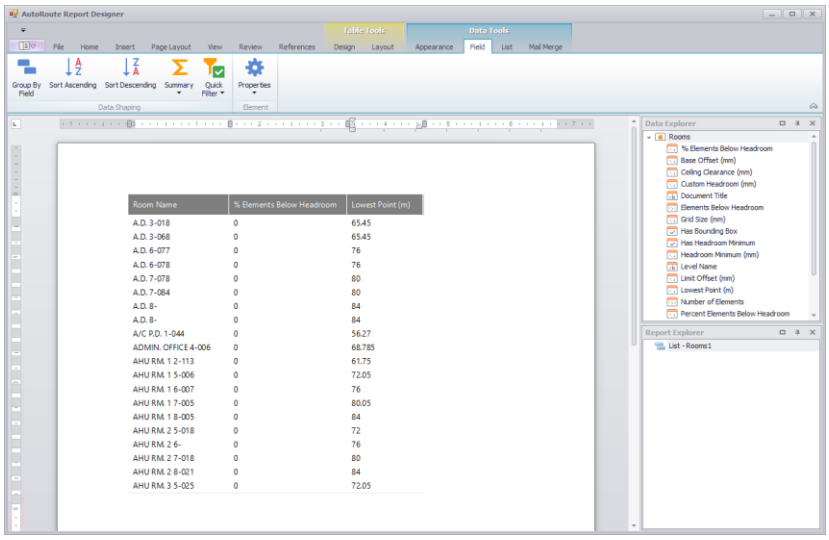
When the users click on the above “Report” button, an AutoRoute Report Designer, which is provided by the DevExpress, shown as below, will show to allow the users to design the report including some parameters that the users want to include.



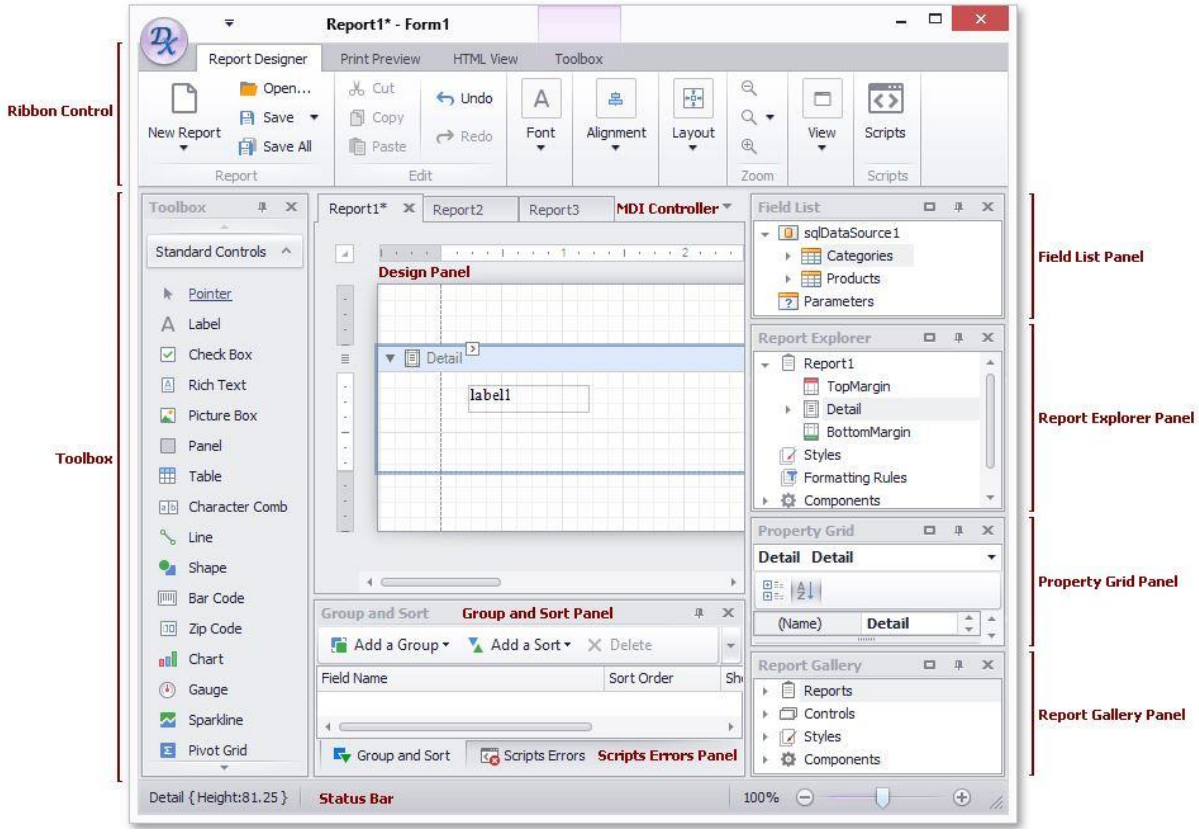
On the left side of the designer, users can choose the fields they like and in the list, there exists some fields that AutoRoute has already provided.



For example, users can make the table like that.



For the details of the End-User Report Designer with a Ribbon Toolbar, here is the example to describe the toolbar.



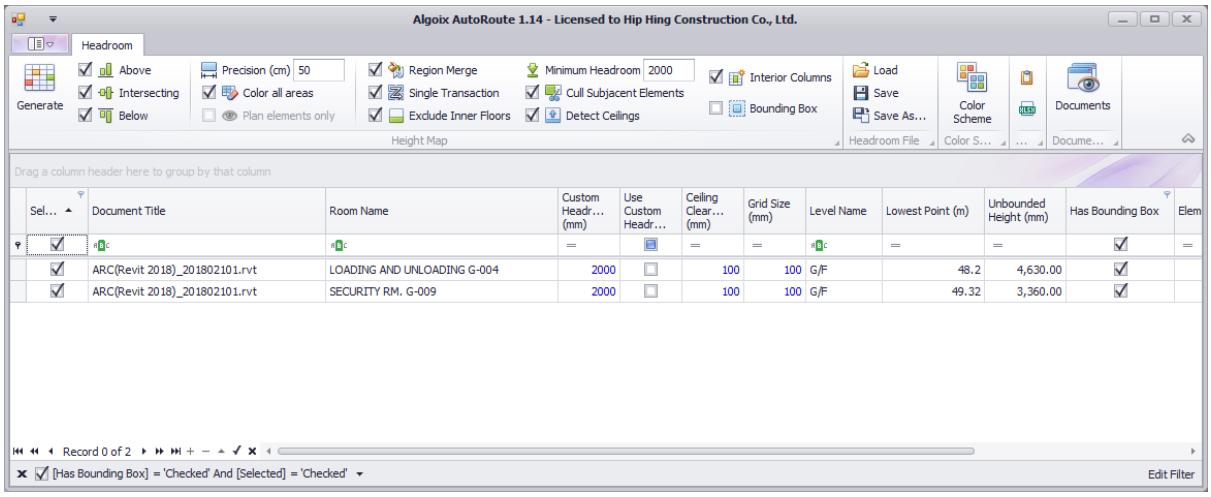
Reference links with more details about DevExpress are available in footnotes[4][5].

[4] https://documentation.devexpress.com/XtraReports/8546/Creating-End-User-Reporting-Applications/WinForms-Reporting/Report-Designer/GUI/End-User-Report-Designer-with-a-Ribbon-Toolbar

[5] https://documentation.devexpress.com/WindowsForms/833/Controls-and-Libraries/Data-Grid/End-User-Capabilities

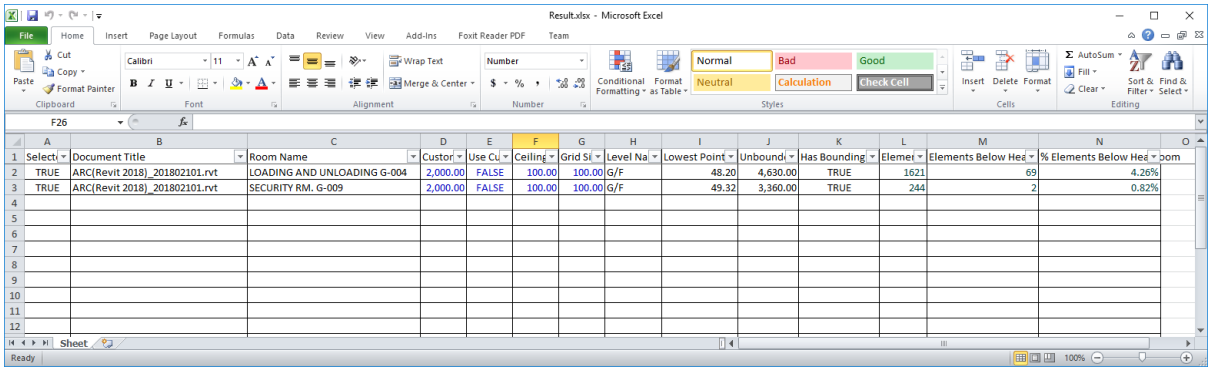
EXPORT TO EXCEL

Users can also export the result of the displayed table on the AutoRoute Interface.



Users click on the icons below. The above table of rooms will be exported to an excel file and saved in the designated location.

The created excel file contains the below information.



RECOMMENDED SETTINGS

In conclusion, AutoRoute can help the users to analyze the space utilization when building services elements are placed, and review the results at a glance in the plan view. For the most accurate and the fastest calculation, the check boxes of "**Above**", "**Intersecting**", "**Below**", "**Color all areas**", "**Region Merge**", "**Single Transaction**", "**Exclude Inner Floors**", "**Cull Subjacent Elements**", "**Detect Ceilings**" and "**Interior Columns**". The box of "**Bounding Box**" should not be checked.