BỘ GIÁO DỤC VÀ ĐÀO TẠO

**TRƯỜNG ĐẠI HỌC XÂY DỰNG HÀ NỘI**

**ĐỒ ÁN**

**LẬP TRÌNH ỨNG DỤNG TRONG XÂY DỰNG**

**Add-in Vẽ nhanh thép cột tương tác với giao diện người dùng**

**<Lớp: XXXX>**

**<Số nhóm: XXXX>**

**Sinh viên thực hiện:**

|  |  |  |
| --- | --- | --- |
| NGUYỄN VĂN A | LỚP | MÃ SV |

**Giáo viên hướng dẫn:**

|  |
| --- |
|  |

**Hà Nội, 07/2025**

Lời cảm ơn

**CHƯƠNG 1. TỔNG QUAN**

Trong những năm gần đây, việc ứng dụng Building Information Modeling (BIM) ngày càng phổ biến trong thiết kế và thi công xây dựng. Autodesk Revit là một trong những phần mềm BIM được sử dụng rộng rãi nhất tại Việt Nam và thế giới. Tuy nhiên, Revit mặc định vẫn còn nhiều hạn chế trong việc hỗ trợ người dùng triển khai nhanh các chi tiết kỹ thuật – đặc biệt là cốt thép.

* 1. Các nghiên cứu và công cụ hiện có
     1. Các nghiên cứu và add-in về triển khai cốt thép

Một số add-in thương mại như **SOFiSTiK Reinforcement Detailing**, **Graitec PowerPack for Revit**, hay **IDEA StatiCa Detail** đã cung cấp các công cụ hỗ trợ vẽ và thống kê cốt thép tự động trong Revit. Tuy nhiên, các công cụ này chủ yếu phục vụ thị trường nước ngoài, có chi phí cao và đôi khi không phù hợp hoàn toàn với tiêu chuẩn thiết kế Việt Nam (TCVN 5574:2018, TCVN 5575:2012…).

Ngoài ra, một số nghiên cứu học thuật đã thử nghiệm viết add-in hỗ trợ vẽ thép bằng Revit API. Tuy nhiên, phần lớn các nghiên cứu chỉ dừng lại ở mức cơ bản (vẽ từng thanh), chưa có giải pháp tối ưu cho việc vẽ nhanh, đặc biệt là với cấu kiện bê tông có nhiều biến thể như cột.

* + 1. Ưu và nhược điểm của các tài liệu và phần mềm hiện có

|  |  |  |
| --- | --- | --- |
| **Công cụ / Tài liệu** | Ưu điểm | Nhược điểm |
| SOFiSTiK, Graitec, IDEA | |  | | --- | |  |   Mạnh về tính tự động, tích hợp nhiều tính năng | Chi phí cao, không hỗ trợ tiêu chuẩn Việt Nam |
| Add-in mã nguồn mở (trên GitHub, blog kỹ thuật) | Có thể tùy chỉnh, học hỏi từ mã nguồn | Giao diện kém thân thiện, thiếu tính năng cho thép cột |
| Tài liệu Autodesk Revit API | Hướng dẫn đầy đủ API | |  | | --- | |  |  |  | | --- | | Còn phức tạp, khó tiếp cận với sinh viên mới học lập trình | |

* 1. Nhận xét và khoảng trống nghiên cứu

Qua phân tích, có thể thấy rằng hầu hết các phần mềm hiện có đều chưa giải quyết tốt bài toán **tự động hóa và tối ưu quy trình vẽ thép cột theo tiêu chuẩn Việt Nam** trong Revit. Ngoài ra, các tài liệu hướng dẫn về Revit API tuy có đầy đủ về mặt kỹ thuật nhưng vẫn chưa có một giải pháp cụ thể được thiết kế riêng cho nhu cầu trong nước.

* 1. Tên đề tài và lý do chọn đề tài

Từ các đánh giá trên, tôi chọn đề tài:  
**“Lập trình add-in Revit API hỗ trợ vẽ nhanh thép cột theo tiêu chuẩn Việt Nam”**.

**Lý do chọn đề tài** là nhằm tạo ra một công cụ add-in đơn giản, dễ sử dụng, có khả năng tự động hóa quy trình vẽ cốt thép cột trong Revit – từ đó giảm thời gian triển khai, đảm bảo tính chính xác, đồng thời phù hợp với tiêu chuẩn và thói quen thiết kế trong nước.

**CHƯƠNG 2. CƠ SỞ LÝ THUYẾT**

* 1. Khái quát về cấu kiện cột bê tông cốt thép

Cột bê tông cốt thép là cấu kiện chịu lực thẳng đứng chủ yếu trong công trình, có nhiệm vụ truyền tải trọng từ dầm, sàn xuống móng. Cột có thể có tiết diện vuông, chữ nhật, tròn hoặc các dạng đặc biệt, trong đó phổ biến nhất là vuông và chữ nhật.

Tùy vào nội lực tính toán (moment, lực dọc) và yêu cầu thiết kế, thép cột thường bao gồm:

* **Thép dọc (thép chủ):** chịu lực nén hoặc kéo, thường bố trí dọc theo chiều cao cột.
* **Thép đai:** có chức năng giữ các thanh thép dọc, chống phình, ngăn chặn sự mất ổn định cục bộ và tăng khả năng dẻo của cột.
  1. Các quy định thiết kế liên quan (TCVN 5574:2018)

Theo tiêu chuẩn Việt Nam hiện hành:

* Số lượng và đường kính thép chủ được xác định theo yêu cầu chịu lực, nhưng tối thiểu không ít hơn 4 thanh.
* Khoảng cách giữa các đai tuân theo quy định, thường từ 100–200 mm tùy chiều cao và vị trí của cột (đầu, giữa, chân).
* Đường kính đai tối thiểu là 6 mm, và phải đảm bảo neo buộc đủ chiều dài (thường có dạng móc 135° hoặc 90°).

Các quy định này cần được phản ánh đúng trong quá trình lập trình tự động vẽ cốt thép.

* 1. Cấu kiện cột trong Revit

Cột trong Revit thuộc nhóm Structural Column (BuiltInCategory.OST\_StructuralColumns). Cột có thể là Family dạng Loadable Family (thường là Family cột chữ nhật hoặc vuông).

* 1. XYZ

XYZ là kiểu dữ liệu định nghĩa một điểm hoặc một vector chỉ phương trong không gian mô hình Revit. Kiểu dữ liệu XYZ dùng để định nghĩa các thông tin tọa độ, vector chỉ phương của các đối tượng hình học trong Revit.

Định nghĩa XYZ trong thư viện Revit API:

namespace Autodesk.Revit.DB

{

public class XYZ

{

public XYZ();

public XYZ(double x, double y, double z);

public double this[int idx] { get; }

public static XYZ Zero { get; }

public static XYZ BasisX { get; }

public static XYZ BasisY { get; }

public static XYZ BasisZ { get; }

public double Z { get; }

public double Y { get; }

public double X { get; }

}

}

Các thuộc tính tĩnh đại diện cho kiễu dữ liệu XYZ có thể truy xuất để lấy thông tin hệ tọa độ gốc của mô hình dự án.

XYZ origin = XYZ.Zero; // Gốc tọa độ, có giá trị là {0; 0; 0}

XYZ vecX = XYZ.BasisX; // Vector chỉ phương trục X, có giá trị là {1; 0; 0}

XYZ vecY = XYZ.BasisY; // Vector chỉ phương trục Y, có giá trị là {0; 1; 0}

XYZ vecZ = XYZ.BasisZ; // Vector chỉ phương trục Z, có giá trị là {0; 1; 0}

Ngoài ra, thư viện Revit API còn cung cấp các toán tử và phương thức để hỗ trợ việc tính toán các đối tượng XYZ như:

*Truy xuất các giá trị tọa độ X, Y, Z theo các phương trong hệ trục XYZ:*

double x = point.X;

double y = point.Y;

double z = point.Z;

*Truy xuất chiều dài Vector:*

double length = vec.GetLength();

*Di chuyển một điểm theo (hoặc ngược chiều) vector cho trước:*

XYZ endPnt1 = startPnt + vector;

XYZ endPnt2 = startPnt - vector;

*Phóng to, thu nhỏ độ lớn của một vector với một số:*

XYZ multiVector = vector \* factor;

XYZ divideVector = vector / factor;

*Tích vô hướng, tích có hướng giữa hai vector:*

double d = vector1.DotProduct(vector2);

XYZ vector3 = vector1.CrossProduct(vector2));

* 1. Các thông số (parameters) và cách lấy giá trị của cột trong Revit API
     1. Tên loại cột (Family Name)
* **Ý nghĩa:** Xác định loại family cột mà người dùng đang triển khai, giúp phân biệt kết cấu và hình học của cột.
* **Parameter trong Revit:** Family.Name
* **Cách lấy giá trị bằng API:**

string familyName = column.Symbol.Family.Name;

* + 1. Kích thước tiết diện (b x h)
* **Ý nghĩa:** Xác định hình dạng tiết diện của cột để bố trí thép chủ và thép đai hợp lý.
* **Parameter trong Revit:**
  + Width (chiều rộng)
  + Depth (chiều sâu)
* **Cách lấy giá trị bằng API:**

Parameter widthParam = column.get\_Parameter(“b”);

Parameter depthParam = column.get\_Parameter(“h”);

double width = widthParam.AsDouble(); // đơn vị feet

double depth = depthParam.AsDouble();

* + 1. Chiều cao cột
* **Ý nghĩa:** Dùng để tính toán số đai và chiều dài thép chủ.
* **Parameter trong Revit:**
  + Unconnected Height (nếu cột không gắn level trên)
  + hoặc tính từ Top Offset – Base Offset
* **Cách lấy giá trị bằng API:**

Parameter heightParam = column.get\_Parameter(BuiltInParameter.INSTANCE\_LENGTH);

double height = heightParam.AsDouble();

* + 1. Vị trí đáy / đỉnh cột (Base Level, Top Level)
* **Ý nghĩa:** Xác định cao độ vị trí cột trong công trình.
* **Parameter trong Revit:**
  + Base Level
  + Top Level
* **Cách lấy giá trị bằng API:**

Parameter baseLevelParam = column.get\_Parameter(BuiltInParameter.FAMILY\_BASE\_LEVEL\_PARAM);

Parameter topLevelParam = column.get\_Parameter(BuiltInParameter.FAMILY\_TOP\_LEVEL\_PARAM);

ElementId baseLevelId = baseLevelParam.AsElementId();

ElementId topLevelId = topLevelParam.AsElementId();

Sau đó có thể lấy tên level:

Level baseLevel = doc.GetElement(baseLevelId) as Level;

string baseLevelName = baseLevel.Name;

* + 1. Offset đáy / đỉnh (Base Offset, Top Offset)
* **Ý nghĩa:** hiệu chỉnh cao độ gốc khi đặt cột.
* **Parameter trong Revit:**
  + Base Offset
  + Top Offset
* **Cách lấy giá trị bằng API:**

double baseOffset = column.get\_Parameter(BuiltInParameter.FAMILY\_BASE\_LEVEL\_OFFSET\_PARAM).AsDouble();

double topOffset = column.get\_Parameter(BuiltInParameter.FAMILY\_TOP\_LEVEL\_OFFSET\_PARAM).AsDouble();

* + 1. Vị trí trong không gian (Location)
* **Ý nghĩa:** Lấy tọa độ điểm đặt cột.
* **Cách lấy giá trị bằng API:**

LocationPoint loc = column.Location as LocationPoint;

XYZ point = loc.Point;

double x = point.X;

double y = point.Y;

double z = point.Z;

* 1. Các thông số (Parameter) của đối tượng Thép – Rebar trong Revit

Trong Revit, một đối tượng Rebar đại diện cho một thanh cốt thép đặt trong mô hình. Việc lấy các parameter của Rebar giúp lập trình viên điều khiển chính xác hình dáng, vị trí, số lượng và cấu tạo của thép trong mô hình.

* + 1. Rebar Number
* **Ý nghĩa**: Mã đánh số thanh thép, dùng để phân biệt giữa các nhóm thép khác nhau.
* **Cách lấy**:

string number = rebar.get\_Parameter(BuiltInParameter.REBAR\_NUMBER).AsString();

* **Công dụng**: Phục vụ cho việc gán mã hiệu, phân nhóm trong bảng thống kê.
  + 1. Bar Diameter
* **Ý nghĩa**: Đường kính danh nghĩa của thanh thép (ví dụ: 12mm, 16mm).
* **Cách lấy**:

double diameter = rebar.get\_Parameter(BuiltInParameter.REBAR\_BAR\_DIAMETER).AsDouble(); // feet

* **Công dụng** Dùng để xác định kích thước thép, tính diện tích thép As.
  + 1. Rebar Shape
* **Ý nghĩa**: Hình dạng thanh thép (L, U, thẳng, móc, …).
* **Cách lấy**:

ElementId shapeId = rebar.get\_Parameter(BuiltInParameter.REBAR\_SHAPE).AsElementId();

Element shape = doc.GetElement(shapeId);

string shapeName = shape.Name;

* **Công dụng** Dùng để xác định kích thước thép, tính diện tích thép As.
  1. Tham số thiết kế bổ sung do người dùng nhập

Ngoài ra, phần mềm nên cho phép người dùng khai báo thêm:

* Số lượng thép chủ
* Đường kính thép chủ
* Đường kính và khoảng cách thép đai
* Lớp bảo vệ bê tông
* Kiểu móc neo đai

Các giá trị này có thể lưu vào file cấu hình (JSON) hoặc đặt trong form khi chạy add-in, để chủ động thay đổi mà không phải sửa code.

**CHƯƠNG 3: THUẬT TOÁN**

**A diagram of a process

AI-generated content may be incorrect.**

**CHƯƠNG 4**

* 1. Giới thiệu chung về ngôn ngữ C#

C# (phát âm là "C sharp") là một ngôn ngữ lập trình hiện đại, hướng đối tượng (Object-Oriented Programming - OOP), được phát triển bởi Microsoft và ra mắt lần đầu vào năm 2000 trong khuôn khổ nền tảng .NET Framework. Ngôn ngữ này được thiết kế nhằm kết hợp giữa tính mạnh mẽ của ngôn ngữ C++ và sự đơn giản, dễ học của Java, giúp lập trình viên có thể xây dựng các ứng dụng đa dạng với hiệu suất cao và độ an toàn trong quản lý bộ nhớ.

C# nổi bật với cú pháp rõ ràng, khả năng kiểm soát lỗi chặt chẽ, cùng khả năng tương tác tốt với các thư viện và công cụ hỗ trợ của hệ sinh thái .NET. Điều này giúp C# được ứng dụng rộng rãi trong phát triển phần mềm từ các ứng dụng desktop truyền thống, ứng dụng web, ứng dụng di động cho tới game, hệ thống nhúng và các ứng dụng doanh nghiệp.

* 1. Lịch sử và quá trình phát triển của C#

Ngôn ngữ C# được Anders Hejlsberg cùng đội ngũ Microsoft phát triển nhằm mục đích tạo ra một ngôn ngữ có tính hiện đại, an toàn và dễ dàng mở rộng trên nền tảng .NET.

* **Năm 2000**: C# được công bố cùng với .NET Framework 1.0, giới thiệu những tính năng cơ bản của ngôn ngữ lập trình hướng đối tượng.
* **Các phiên bản tiếp theo**: Từ C# 2.0 đến 4.0 (giai đoạn 2005 - 2010), ngôn ngữ được bổ sung nhiều tính năng quan trọng như:
  + Generics (cho phép viết code tổng quát, tái sử dụng cao)
  + LINQ (Language Integrated Query) giúp truy vấn dữ liệu theo cú pháp đơn giản, mạnh mẽ
  + Lambda expressions, extension methods nâng cao khả năng lập trình hàm
* **C# 5.0 trở đi**: Hỗ trợ lập trình bất đồng bộ (async/await), cải tiến cú pháp giúp code ngắn gọn và dễ đọc hơn.
* **C# 9, 10, 11, 12 (hiện đại)**: Tập trung vào các tính năng nâng cao như record types (loại dữ liệu bất biến), pattern matching, cải tiến null safety, và khả năng biểu diễn dữ liệu hiệu quả hơn.
  1. Đặc điểm nổi bật của ngôn ngữ C#
* **Hướng đối tượng đầy đủ**  
  C# hỗ trợ đầy đủ các tính chất cơ bản của lập trình hướng đối tượng như kế thừa, đa hình, đóng gói và trừu tượng. Điều này giúp tổ chức mã nguồn logic, dễ bảo trì và mở rộng.
* **Quản lý bộ nhớ tự động**  
  C# sử dụng cơ chế Garbage Collector (GC) để tự động quản lý bộ nhớ, giải phóng bộ nhớ không còn được sử dụng, giúp giảm nguy cơ lỗi tràn bộ nhớ hoặc lỗi do quản lý thủ công.
* **Kiểm tra kiểu tĩnh và an toàn**  
  C# là ngôn ngữ có kiểu dữ liệu tĩnh (static typing), tức là các kiểu biến được xác định rõ ràng khi biên dịch, giúp phát hiện lỗi sớm và tránh lỗi trong quá trình chạy.
* **Tích hợp sâu với .NET Framework và .NET Core**  
  C# có khả năng sử dụng hàng ngàn thư viện và công cụ của .NET, từ việc xử lý giao diện người dùng (UI), làm việc với dữ liệu, mạng, đến các tính năng nâng cao như đa luồng, xử lý bất đồng bộ.
* **Hỗ trợ đa nền tảng**  
  Với sự phát triển của .NET Core và .NET 5+, các ứng dụng viết bằng C# có thể chạy trên nhiều hệ điều hành khác nhau như Windows, Linux và macOS.
* **Phát triển nhanh, năng suất cao**  
  Ngôn ngữ này cho phép lập trình viên viết code sạch, dễ hiểu, và sử dụng nhiều công cụ hỗ trợ, giúp tăng hiệu quả và giảm thời gian phát triển phần mềm.
  1. C# trong lĩnh vực xây dựng và kỹ thuật tính toán

Trong ngành xây dựng, kỹ thuật tính toán đóng vai trò quan trọng trong việc thiết kế, phân tích và tối ưu kết cấu công trình. C# đã và đang được sử dụng để phát triển nhiều phần mềm hỗ trợ kỹ sư xây dựng trong các công việc như tính toán kết cấu bê tông cốt thép, thép, phân tích ứng suất, tối ưu vật liệu, và mô phỏng xây dựng. Các phần mềm tiêu biểu sử dụng C# bao gồm:

* **Tekla Structures**  
  Một phần mềm nổi tiếng trong thiết kế kết cấu thép và bê tông, cung cấp các giải pháp mô hình hóa thông tin công trình (BIM). Tekla sử dụng .NET và C# để phát triển các module mở rộng, cho phép người dùng tùy chỉnh và tự động hóa các quy trình thiết kế.
* **Autodesk Revit (Add-ins phát triển bằng C#)**  
  Revit là phần mềm BIM phổ biến, nhiều plugin và add-in hỗ trợ tính toán, kiểm tra, và lập bản vẽ kết cấu được viết bằng C#. Lập trình viên có thể sử dụng API của Revit dựa trên C# để phát triển các công cụ tùy chỉnh phục vụ thiết kế và kiểm tra cấu kiện.
* **SOFiSTiK**  
  Đây là phần mềm phân tích và thiết kế kết cấu có hỗ trợ lập trình tùy chỉnh với ngôn ngữ .NET, trong đó C# được sử dụng để viết các script tự động hóa tính toán, xuất báo cáo, và xử lý dữ liệu kết cấu.
* **Phần mềm tự phát triển (Custom Software)**  
  Nhiều công ty xây dựng và tư vấn kỹ thuật sử dụng C# để xây dựng phần mềm riêng, phục vụ tính toán nhanh diện tích thép, tải trọng, phân tích kết cấu theo tiêu chuẩn Việt Nam (TCVN) hoặc tiêu chuẩn quốc tế.

Nhờ tính mạnh mẽ, dễ tích hợp với các hệ thống khác và môi trường phát triển hiện đại, C# trở thành ngôn ngữ ưu tiên hàng đầu trong phát triển phần mềm kỹ thuật và xây dựng hiện nay.

* 1. Ví dụ minh họa một đoạn mã C#

using System;

namespace ConstructionCalculation

{

class Program

{

static void Main(string[] args)

{

// Tính diện tích cốt thép cần thiết cho cột theo công thức đơn giản

double load = 1500; // tải trọng tính toán (kN)

double concreteStrength = 25; // cường độ bê tông (MPa)

double steelYieldStrength = 415; // cường độ chịu kéo thép (MPa)

// Giả sử diện tích thép cần = tải trọng / cường độ thép

double steelArea = load \* 1000 / steelYieldStrength; // cm2 (đơn giản)

Console.WriteLine("Diện tích thép cần thiết cho cột: {0:F2} cm2", steelArea);

}

}

}

**Giải thích:**

* Đoạn code tính diện tích thép đơn giản dựa trên tải trọng và cường độ thép.
* Sử dụng biến kiểu double để tính toán số thực.
* Kết quả được in ra màn hình với định dạng 2 chữ số thập phân.
  1. Kết luận

Ngôn ngữ lập trình C# với các đặc điểm mạnh mẽ về lập trình hướng đối tượng, quản lý bộ nhớ tự động, tích hợp sâu với nền tảng .NET và khả năng đa nền tảng đã trở thành công cụ đắc lực trong phát triển phần mềm nói chung và phần mềm kỹ thuật xây dựng nói riêng. Nhiều phần mềm quan trọng trong ngành xây dựng hiện nay, từ các hệ thống BIM đến các phần mềm tính toán kết cấu đều ứng dụng C# để nâng cao hiệu quả, độ chính xác và tính mở rộng của sản phẩm.

Việc học và ứng dụng C# không chỉ giúp lập trình viên phát triển các ứng dụng chuyên sâu mà còn hỗ trợ các kỹ sư xây dựng tối ưu hóa công việc, nâng cao chất lượng thiết kế và thi công công trình.

**CHƯƠNG 5: PHÂN TÍCH ĐÁNH GIÁ CHƯƠNG TRÌNH**

* 1. Mô tả chương trình
  + Chương trình là 1 add-in của Revit

A screenshot of a computer

AI-generated content may be incorrect.

Panel ( DATN\_Kien) được thêm vào như những panel khác ở trong Revit.

Panel có 1 button “Vẽ thép cột”

Khi click vào button “Vẽ thép cột” màn hình chính của chương trình sẽ hiện lên

A screenshot of a computer

AI-generated content may be incorrect.

* + Bên trong giao diện chính có nhiều thông số, đủ để vẽ thép cột

Mark cột, tất cả nhưng mark của cột sẽ hiện ở trong combobox này, khí người dùng chọn vào mark nào thì cột mang mark đó sẽ được selected

A screenshot of a computer

AI-generated content may be incorrect.

Tiếp theo là combobox có tất cả các đường kính thép ở trong dự án

A screenshot of a computer

AI-generated content may be incorrect.

Sau khi setting đầy đủ người dùng click OK nếu muốn thực hiện rải thép cho cột có mark như đã chọn

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

Từ hình ảnh cho thấy, kết quả phản ánh đúng:

* + Cột có mark C3 được vẽ
  + Thép vẽ 1 lớp có đường kính M13.
* Chương trình hoạt động hoàn toàn đúng với setting trên giao diện.

**CHƯƠNG 6: HƯỚNG PHÁT TRIỂN CỦA ĐỀ TÀI**

* 1. Đánh giá đề tài

Sau thời gian nghiên cứu và thực hiện, đề tài “Phát triển Add-in Revit hỗ trợ vẽ nhanh thép cột theo thiết lập của người dùng” đã cơ bản hoàn thành các mục tiêu đặt ra ban đầu. Cụ thể:

* **Tính ứng dụng**: Add-in đã hỗ trợ tự động hóa quy trình vẽ thép cột trong Revit dựa trên thông số mà người dùng nhập. Việc này giúp giảm đáng kể số thao tác thủ công, từ đó tiết kiệm thời gian triển khai mô hình.
* **Tính linh hoạt**: Người dùng có thể điều chỉnh các thông số cấu tạo thép ngay trên giao diện, áp dụng nhanh cho nhiều đối tượng cột khác nhau.
* **Khả năng triển khai**: Công cụ đã được thử nghiệm trên một số mô hình thực tế và hoạt động ổn định, phù hợp với quy trình thiết kế kết cấu theo mô hình BIM.

Tuy nhiên, với đặc thù đây là **đồ án tốt nghiệp của sinh viên**, thời gian thực hiện có hạn và kinh nghiệm về lập trình Revit API còn ở mức cơ bản, nên đề tài vẫn tồn tại những hạn chế nhất định:

* **Phạm vi áp dụng còn hẹp**

Mới xử lý được một số loại tiết diện cột phổ biến (chữ nhật, vuông), chưa hỗ trợ đầy đủ cho các dạng tiết diện tròn, cột biến tiết hoặc cấu tạo đặc biệt.

* **Chưa tích hợp kiểm tra tiêu chuẩn thiết kế**

Add-in chưa thực hiện tính toán và so sánh thông số cấu tạo với tiêu chuẩn TCVN 5574:2023 hoặc các tiêu chuẩn quốc tế, dẫn đến người dùng vẫn phải tự kiểm tra thủ công.

* **Giao diện người dùng đơn giản**

UI chưa trực quan, chưa có tính năng xem trước (preview) cấu tạo trước khi vẽ.

* **Chưa tối ưu hiệu năng**

Khi áp dụng cho số lượng cột lớn, tốc độ vẽ chưa đạt mức tối ưu.

* **Thiếu tính năng quản lý và chia sẻ cấu hình**

Hiện tại, người dùng chưa thể lưu trữ hoặc xuất/nhập các bộ thông số cấu tạo để áp dụng cho dự án khác.

Nhìn chung, mặc dù còn một số hạn chế, nhưng đề tài đã chứng minh được khả năng **ứng dụng thực tế cao**, thể hiện sự kết hợp giữa kiến thức chuyên ngành xây dựng và kỹ năng lập trình. Đây là nền tảng quan trọng để tiếp tục phát triển các tiện ích hỗ trợ triển khai mô hình BIM.

* 1. Hướng phát triển của đề tài

Để nâng cao hiệu quả và mở rộng phạm vi ứng dụng, trong tương lai đề tài có thể được phát triển theo các hướng sau:

**6.2.1. Mở rộng phạm vi áp dụng**

* Bổ sung khả năng vẽ thép cho các loại tiết diện cột đa dạng hơn: tròn, elip, biến tiết, hoặc có hình dạng bất quy tắc.
* Áp dụng nguyên lý hoạt động cho các cấu kiện khác như dầm, sàn, vách, móng… nhằm tạo ra một bộ công cụ Revit Add-in đồng bộ cho cấu tạo kết cấu bê tông cốt thép.

**6.2.2. Nâng cao khả năng tùy biến và giao diện người dùng (UI/UX)**

* Phát triển giao diện sử dụng WPF hoặc WinUI hiện đại, có hỗ trợ xem trước trực tiếp (3D preview) cấu tạo thép trước khi vẽ.
* Cho phép người dùng lưu nhiều bộ **setting cấu tạo** khác nhau, dễ dàng chuyển đổi tùy theo loại công trình.
* Bổ sung chức năng **Import/Export setting** dưới dạng JSON hoặc XML để chia sẻ cấu hình giữa các kỹ sư.

**6.2.3. Tích hợp tiêu chuẩn thiết kế**

* Thêm chức năng kiểm tra thông số cấu tạo so với các tiêu chuẩn TCVN 5574:2023, Eurocode, ACI, BS…
* Cảnh báo khi cấu hình không đáp ứng yêu cầu về khoảng cách, đường kính, số lượng thanh.

**6.2.4. Tối ưu hiệu năng và quy trình làm việc**

* Tối ưu thuật toán xử lý hình học để giảm thời gian vẽ, đặc biệt khi làm việc với mô hình có hàng trăm cột.
* Bổ sung chế độ vẽ hàng loạt nhiều cột dựa trên một mẫu cấu hình có sẵn.
* Nghiên cứu áp dụng đa luồng (multi-threading) hoặc kỹ thuật xử lý song song để cải thiện tốc độ.

**6.2.5. Liên kết và xuất dữ liệu**

* Xuất dữ liệu cấu tạo thép sang Excel, IFC hoặc JSON để phục vụ cho công tác bóc tách khối lượng và lập kế hoạch vật liệu.
* Kết nối với các phần mềm Visualization (như Unity, Unreal Engine) để mô phỏng trực quan tiến trình thi công.

**CHƯƠNG 7: TÀI LIỆU THAM KHẢO**

1. Autodesk, Revit 2024 API Developer Guide.  
   Truy cập tại: https://help.autodesk.com/view/RVT/2024/ENU
2. Autodesk, Revit API Docs – Hướng dẫn và tham khảo lập trình Revit API.  
   Truy cập tại: https://www.revitapidocs.com/
3. Microsoft, C# Programming Guide – Hướng dẫn lập trình ngôn ngữ C#.  
   Truy cập tại: https://learn.microsoft.com/en-us/dotnet/csharp/programming-guide/
4. Tiêu chuẩn quốc gia TCVN 5574:2023 – Kết cấu bê tông và bê tông cốt thép – Tiêu chuẩn thiết kế.  
   Hà Nội: Bộ Khoa học và Công nghệ, 2023.
5. Troelsen, A., Japikse, P., Pro C# 8 with .NET Core 3, Apress, 2020. ISBN: 978-1-4842-5584-4.
6. Autodesk Knowledge Network, Revit API Samples – Ví dụ lập trình API Revit.  
   Truy cập tại: https://knowledge.autodesk.com/support/revit-products
7. Xuân Thu Lab, Học lập trình C# từ cơ bản đến nâng cao.  
   Truy cập tại: https://xuanthulab.net/
8. Smith, D., Tardif, M., Building Information Modeling: A Strategic Implementation Guide, John Wiley & Sons, 2009. ISBN: 978-0470135451.