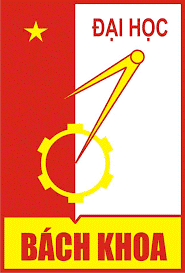
**TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI**

**\*\*\***

**VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG**

**---- -------------------------------------------------------**

**GVHD: Nguyễn Thị Thu Trang**

**Nhóm: 8**

**Thành Viên: Trần Minh Quang – 20176853**

**Lê Hoàng Anh Trung – 20176892**

**Môn học: Lý thuyết ngôn ngữ hướng đối tượng**

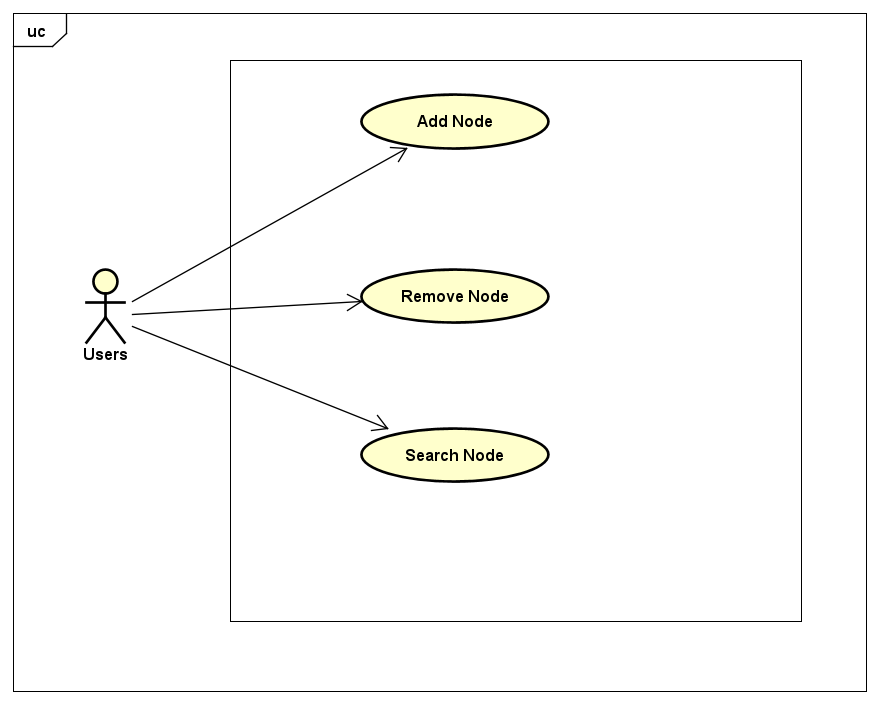
**Mã Lớp: 113833**

Mini-projects

OOLT

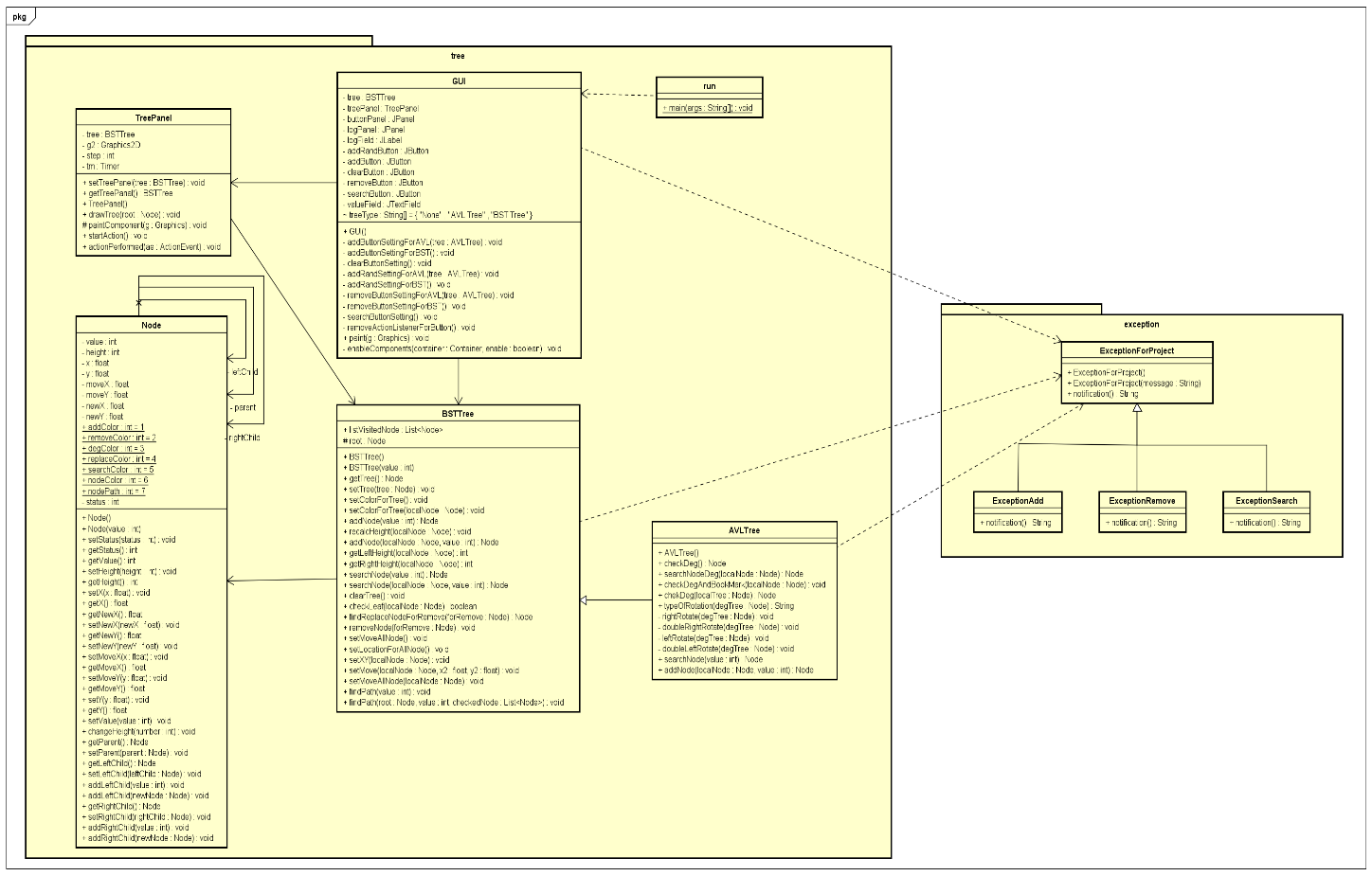
1. **Mô tả Project**
   1. Mô tả yêu cầu bài toán
      * Đề tài 2.1.7 : An application to (visually) demonstrate following data structure and algorithm

* Binnary Search Tree
* AVL Tree
* Tạo ra ứng dụng trực quan cho phép người dùng tìm hiểu về hoạt động của thuật toán *cây nhị phân tìm kiếm* và *cây cân bằng*
  1. Các chức năng và giải thích
     + Biểu đồ UseCase:



* + - Ứng dụng cho phép người dùng lựa chọn thuật toán mà mình muốn chạy minh họa.
    - Ở mỗi kiểu minh họa thuật toán thì ứng dụng cho phép người dùng thực hiện các chức năng là:
* Thêm 1 Node vào cây
* Xóa 1 Node khỏi cây
* Tìm kiếm 1 Node trong cây
* Chức năng *thêm node* thì cho phép người dùng nhập vào số muốn thêm (nhập giá trị muốn thêm vào trường *ValueField* và bấm nút *Add number*, hoặc có thể thêm Node với giá trị ngẫu nhiên (nút *Add Random number*). Thông báo kết quả và vẽ cây minh họa ra màn hình.
* Chức năng *xóa node* thì cho phép người dùng nhập vào giá trị của node muốn xóa trong cây ở trường *ValueField* sau đó tiến hành xóa node, in thông báo và vẽ lại cây minh họa.
* Chức năng *tìm kiếm node*  cho phép người dùng nhập vào giá trị cho node muốn tìm kiếm trong cây sau đó thông báo kết quả tìm kiếm ra màn hình.

1. **Giải thích ý tưởng thiết kế**
   1. Biểu đồ lớp:



* 1. ***Ý tưởng***: Project sẽ có 2 công việc chính là xây dựng, cài đặt thuật toán của 2 loại cây và phần giao diện cho ứng dụng mô phỏng trực quan cho người dùng.
     + Về phần cài đặt thuật toán thì mỗi cây được tạo nên từ 1 đối tượng ‘Node root’ cùng các hành vi của nó như thêm, xóa, tìm kiếm. Mỗi ‘Node’ sẽ có các thuộc tính và hành vi riêng của nó như màu sắc, vị trí trong cây … Cây AVL sẽ thừa kế từ cây BST, ngoài những hành vi có trong cây BST thì trong cây AVL sẽ có các hành vi khác để cân bằng lại cây khi nó mất cân bằng.
     + Về phần giao diện thì sẽ có phần để hiển thị cây, và có các nút cho phép người dùng tương tác với ứng dụng, cùng với đó là phần thông báo thông tin khi cây hoạt động ra cho người dùng.
  2. ***Trong giao diện của ứng dụng có 3 phần chính***:
     + Phần điều khiển, gồm các nút chức năng và trường nhập dữ liệu
     + Phần dùng để vẽ cây minh họa thuật toán
     + Phần hiện các thông báo khi ứng dụng hoạt động
  3. ***Project gồm có 2 package lớn (Tree và ExceptionForProject) và chia nhỏ cụ thể thành 4 package như sau***:
     + Package *tree.logic* gồm 3 class là *Node, BSTTree, AVLTree*. Mô tả các thuộc tính cũng như các hành động của mỗi node trong cây, của cây BST và cây AVL. Trong đó cây AVL thừa kế cây BST.
     + Package *tree.gui* gồm 2 class là *GUI và TreePanel.* Chứa toàn bộ cài đặt cho giao diện của project. Class *GUI* có nhiệm vụ thiết kế giao diện cho ứng dụng ( vị trí màu sắc các nút chức năng, trường thông báo thông tin …). Class *TreePanl* có nhiệm vụ là vẽ ra cây minh họa.
     + Package *tree.run* khởi tạo GUI() và chạy chương trình.
     + Package *exception*  gồm có Class *ExceptionForProject* cùng với đó là các Class con thừa kế nó tương ứng với các ngoại lệ cụ thể của chương trình như *ExceptionAdd, ExceptionRemove, ExceptionSearch.*

1. **Phân công nhiệm vụ từng thành viên**
   1. Phân công nhiệm vụ

* Trần Minh Quang:
  + Tìm hiểu và lập trình thuật toán cây BST
  + Xử lý phần hiển thị (GUI Class) của cây BST khi thao tác với nó trong ứng dụng
  + Xử lý exception cho project
* Lê Hoàng Anh Trung:
  + Tìm hiểu và lập trình thuật toán cho cây AVL.
  + Xử lý phần hiển thị (GUI Class) của cây AVL khi thao tác với nó trong ứng dụng.
  + Vẽ UseCase, ClassDiagram.
* Cùng nhau tìm hiểu và thực hiện phần animation cho ứng dụng
  1. Tham khảo ý tưởng ban đầu tại link:

<https://github.com/Zurmansor/AVL-Tree/tree/5877817d8b9c7232b85ac8ea1d609a27ce7c9bcd>