LUNG CANCER

ASSESSMENT 0.1

by: Mustafa Maytham Mahmood
Mustafa Yarub Abdul-Hussein
Rusul Hayder Abd Zaid

Supervised by: Dr.Suhad F. Shihan

Our Project's OOP Refactoring Journey

01 The Project

A Lung Cancer Prediction Application.

02 The Goal

We're showing not just what our project does, but how its architecture was engineered.

03 The Journey

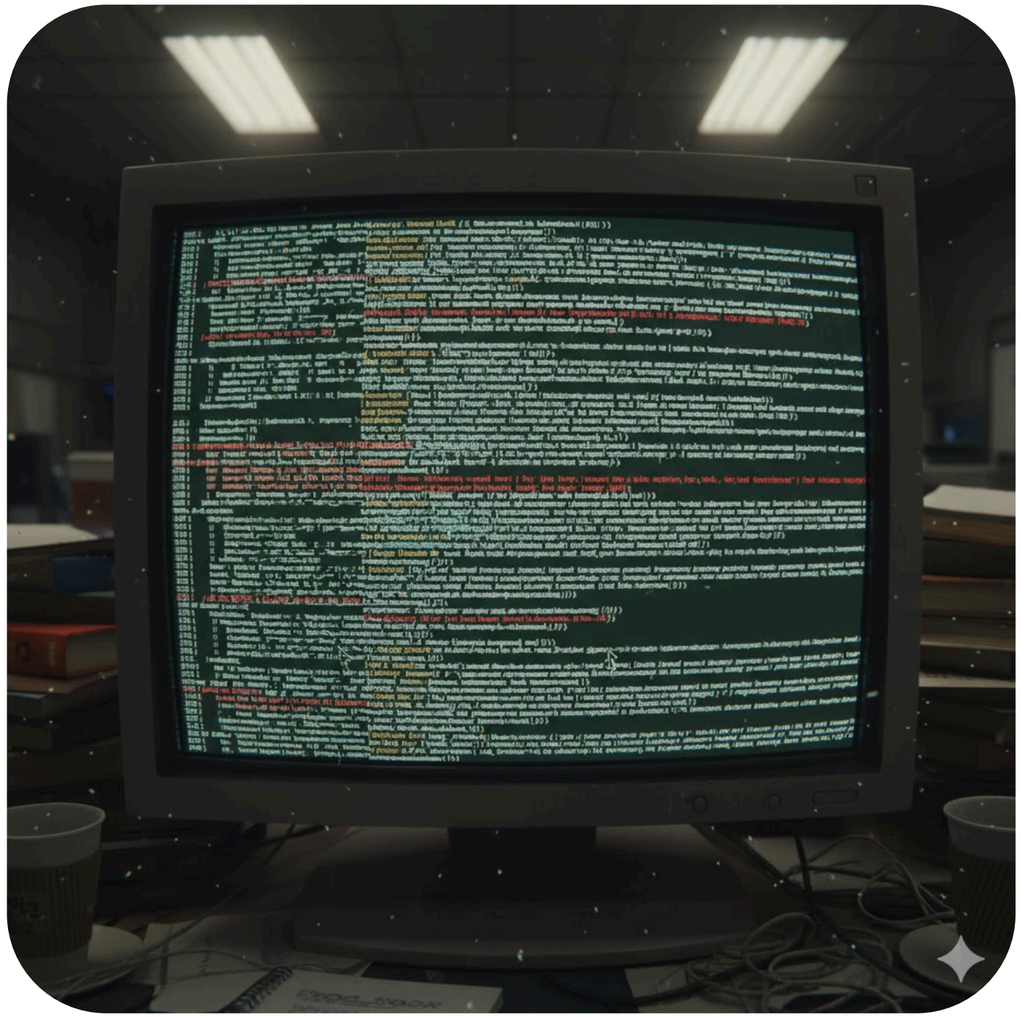
How we moved from a simple "script" to an organized "system" using Object-Oriented Programming (OOP).

Where Was the Problem?

- **Description**: The code started as one, very long file.

- **The Problems:**

  a. **Chaos**: Data loading, model training, and GUI code were all mixed in one place.
  b. **Hard to Maintain**: A simple change in one part could accidentally "break" another.
  c. **Hard to Develop**: It was impossible for two people to work on the code at the same time.
  d. **No Reusability**: We couldn't take any part of it for another project.

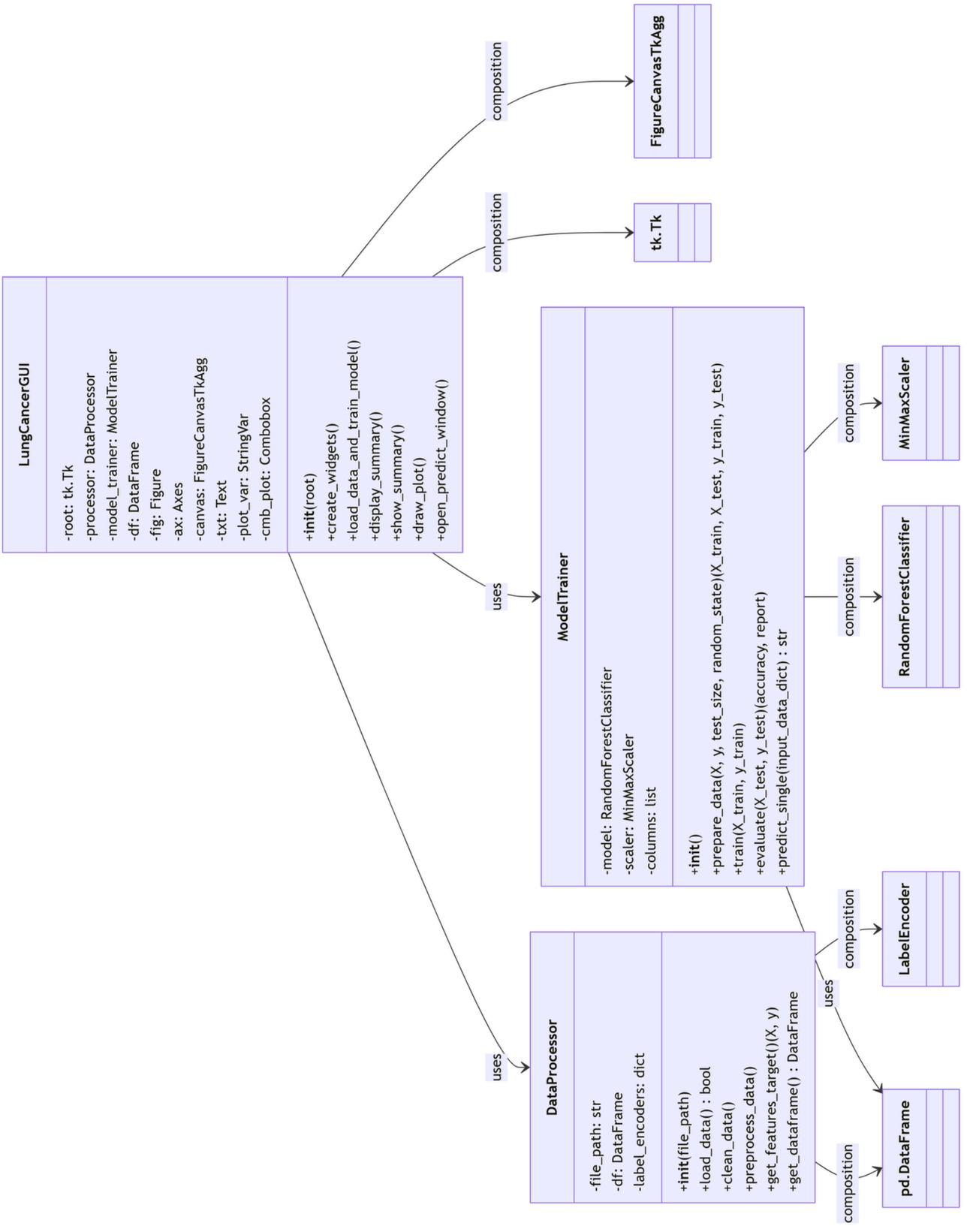


Applying OOP Principles (Separation of Concerns)

- **The Principle: "Separation of Concerns".**
- **The Plan: Instead of one worker doing everything, we created a "team of specialists" – these are our Classes.**

a. **DataProcessor** Class: Its only job: Load, clean, and prepare data.
b. **ModelTrainer Class**: Its only job: Train and evaluate the model.
c. **LungCancerGUI Class**: Its only job: Display buttons and screens.

It proves the system is organized, logical, and well-structured.



The Blueprint (The UML Diagram)

What Did We Gain From This?

- **Organized Code: We know exactly where to go to modify any feature.**
- **Easy Maintenance: We can update the GUI without fearing we'll break the model.**
- **Reusability: We can now take the ModelTrainer class and reuse it in a new project.**
- **Teamwork: Our team can work on different classes at the same time.**

so ..

Moving from a "script" to "classes" isn't just a cosmetic change.
It's what turns "code that works" into an "engineered system" that is professional, scalable, and easy to maintain.