GitHub Copilot transcript

Scene 1

Introduction

GitHub Copilot for Automation Scripting

Hello Everyone, welcome to this exciting course on **GitHub Copilot for Automation Scripting.**

In this course we will learn how to use GitHub Copilot tool for Selenium java Automation Scripts generation.

If you're someone who's working with Selenium or just getting started with automation testing, this course is going to show you, how AI can supercharge your scripting workflow.

Before we dive into the course overview, let me take a moment to introduce myself.

I am Rahul Ritesh,

I'm currently working as a Test Automation Lead, and I have over 9 years of hands-on experience in the world of automation engineering.

Over the years, I've had the chance to work across some of the most exciting and challenging domains — from Banking **and** Payments **to** E-commerce, Salesforce CRM, and even SAP ERP systems.

Each of these industries taught me something new —

- how to build smarter frameworks,
- solve real-world testing problems,
- and deliver automation solutions that actually make a difference.

And now, I'm here to share all of that with you, along with some powerful AI tools like GitHub Copilot that are changing the way we write automation scripts in our day to day automation activities.

So buckle up — we're about to make automation faster, smarter, and a whole lot more fun!

Scene 2

Alright — now that you know a bit about me, Let's jump right into the course introduction!

In the next few minutes,

- I'll walk you through exactly what you'll be learning,
- how we'll be using GitHub Copilot alongside Selenium with Java,
- and how each section is designed to help you build real-world automation scripts — faster and smarter.

Here i will be using Selenium with Java, along with the BDD Cucumber framework, to generate automation scripts.

GitHub Copilot isn't just for Java or Selenium — it's a smarter way to code fast.
Once you learn the concepts of using GitHub Copilot, you can apply it to any language or framework.

"We've divided this course into three well-organized sections, each designed to guide you through the learning process in a clear and structured way."

Section 1: Foundation of GitHub Copilot
In this section, we'll build a strong foundation by understanding what GitHub Copilot is, how it works, and how to set it up in your development environment. We'll also explore the concepts of Git, GitHub, version control systems, and the AI technology behind Copilot.

Section 2: Practical Demonstration on Script Generation
Here's where we get hands-on. You'll see how to use GitHub Copilot to generate automation scripts step by step — from creating feature files and page object classes to writing step definitions, test runners, and suite XML files. Everything will be demonstrated live using Selenium with Java and the BDD Cucumber framework.

Section 3: Best Practices and Limitations
To wrap up, we'll cover key tips for using GitHub Copilot the right way.

- You'll learn how to write clear prompts,
- spot its limits,
- and make sure the code it suggests is solid, easy to maintain, and ready for real-world use.

Each part is packed with hands-on tips and examples to help you feel confident using Copilot for automation tasks.

So Let's get started!

Scene 3

GitHub Copilot setup in IntelliJ

Scene 4

Version Control System

Alright, let's take a step back and talk about why tools like Git were even created in the first place.

Imagine you're working on a school project or a software app with a few friends.
You're all editing the same files, maybe even at the same time.
Now, here's where the chaos begins.
Let's say you make some changes to a file and save it.
Then your friend also makes changes to the same file and saves it.

So when you collate all the code written by different people —your changes are gone. Overwritten. Just like that.

Or maybe you're working alone, and you try something new in your code.
It breaks everything.
Now you want to go back to the version that was working... but you didn't save a backup.
So now you're stuck.

Before Git, people used to do things like:

- Saving multiple copies of the same file with names like final_version, final_final, or final_really_this_time.
- Emailing files back and forth.
- Copy-pasting code into Notepad just in case something broke.

It was messy. It was risky. And it wasted a lot of time.
So developers needed a better way to:

- Track changes in their code.
- Work together without overwriting each other's work.
- Go back in time if something broke.
- And see who did what and when.

That's where version control systems came in.

Git is a Version Control System tool which was created by Linus Torvalds in 2005, became one of the most powerful and popular ones.
With Git, you can:

- Save snapshots of your project at any point.
- Work on different features without messing up the main code.
- Merge everyone's work together safely.
- And undo mistakes easily.

So in short, Git was created to solve the real pain of working on code—especially with teams—and to make sure no one ever had to name a file final_final_v2_really_done_THIS_ONE.zip again.

Scene 5

GITHUB Platform

Alright, so now that we understand what Git is and why it was created, let's talk about GitHub—because this is where things get even more powerful.
So, Git helps you track changes in your code and manage versions.

But here's the thing—Git works locally on your computer.

That means if your laptop crashes or gets lost, your project could be gone.
Also, if you're working with a team, how do you share your code?
How do you collaborate without emailing zip files back and forth?
That's where GitHub comes in.

GitHub is like a cloud-based home for your Git projects.
It's a website where you can:

- Store your code online
- Back it up safely
- Share it with others
- And collaborate with teammates in real time

Think of GitHub as a social network for developers—but instead of posting photos, you're posting code.
You can:

- Create repositories (which are like folders for your projects)
- Track issues and bugs
- Review each other's code
- Suggest changes

- And even roll back to earlier versions if something breaks

And the best part?
GitHub makes it super easy to work with Git.
You don't need to be a command-line expert.
You can do a lot of things right from the GitHub website or through tools like GitHub Desktop or Code Editors.
So in short, GitHub exists to:

- Make Git easier to use
- Help teams collaborate
- And keep your code safe and accessible from anywhere

It's like having a shared workspace in the cloud where everyone can contribute, review, and improve code—together.

Scene 6

how GitHub Copilot work------

Alright, so let's talk about what's actually powering GitHub Copilot behind the scenes.
GitHub Copilot is powered by AI—specifically, a type of AI called a large language model, which was developed by OpenAI.
Now, what does that mean in simple terms?

Well, imagine this AI has read billions of lines of code—yes, billions!
It's been trained on tons of public code from GitHub and other sources.
So it's kind of like a super smart assistant that's seen almost every kind of code pattern out there.
Because of all that training, it's learned how code usually looks, how functions are written, how bugs are fixed, and so on.

So when you start typing code, Copilot is like:
"Hmm... based on everything I've seen before, I think I know what you're trying to do."
And then it suggests the next line or even a whole block of code for you.
It's kind of like predictive text on your phone—but for code, and way more powerful.
Now here's something important:

Copilot doesn't learn from your private code—unless you specifically give it permission.
So your personal or company code stays private by default.

Scene 7

who built GitHub Copilot and how it works behind the scenes:

Alright, so let's clear up who's behind GitHub Copilot and how it all fits together.

The AI technology behind Copilot was developed in collaboration with OpenAI—they're the folks known for creating models like ChatGPT.

But here's the thing:

Even though OpenAI helped build the AI brain, it's GitHub that actually manages Copilot.

That means GitHub handles how Copilot works inside your coding tools, how it's sold, and how it's updated.

So, just to connect the dots:

- Microsoft owns GitHub
- GitHub owns and runs Copilot
- OpenAI helped build the AI that powers it

So it's kind of like this team effort:

- OpenAI builds the engine
- GitHub installs it in the car
- Microsoft owns the garage

Now, let's talk about the AI models used in Copilot.
Copilot uses large language models, or LLMs.

Some of the models it uses include:

- Claude Sonnet 4, which is specially optimized for coding tasks
- And OpenAI's GPT-4, which is great at understanding and generating code

These models are what make Copilot so smart when it comes to predicting what you're trying to code next.

Scene 8

Benefits of using GitHub Copilot:

So, let's talk about the real benefits of using GitHub Copilot—and why so many developers are excited about it.
Copilot is like having a coding buddy sitting right next to you.
You start typing, and it instantly suggests the next line of code.
Sometimes, it even writes the whole function for you!

1. Saves Time
   You don't have to Google every little thing or copy-paste from Stack Overflow.
   Copilot gives you suggestions right inside your editor—so you can stay focused and move faster.
2. Boosts Productivity
   It helps you write more code in less time.
   You can build features quicker, fix bugs faster, and spend less time on repetitive tasks.
3. Great for Learning
   If you're a beginner, Copilot can actually help you learn.
   It shows you how certain things are written, and you can learn by seeing real examples as you type.
4. Reduces Mental Load
   You don't have to remember every syntax rule or function name.
   Copilot fills in the blanks for you, so you can focus on the logic and creativity of your code.
5. Works with Many Languages
   It's not just for Python or JavaScript.
   Copilot supports tons of programming languages—so whatever you're working on, it's got your back.

6. Helps with Documentation and Tests

   It can even help you write comments, documentation, and unit tests.

   So it's not just about writing code—it's about writing better, cleaner code.

Speaker:

So in short, GitHub Copilot is like a smart assistant that helps you code faster, learn better, and stay in the flow.

It's not perfect, but it's a huge boost—especially when you're stuck or just want to speed things up.