

ARTIFICIAL INTELLIGENCE:
THE FUTURE IS NOW

NEW HIRE ORIENTATION



A white humanoid robot is seated in a grey office chair at a wooden desk. The robot has a smooth, white, featureless face and a black neck. Its arms are resting on its lap. In front of the robot on the desk is a white keyboard and a blue mug. The background is a blurred office environment with bookshelves.

AI Killed the Coding Interviews

Danit Nativ Navon

Engineering Manager, Meta

What Broke?



What Broke?





Meeting: Meeting 12:30

Merge Intervals Challenge

Can't get the latest record of first coding states of this city in response to the...
Recent state coding...
Recently state coding...
Recently state coding...

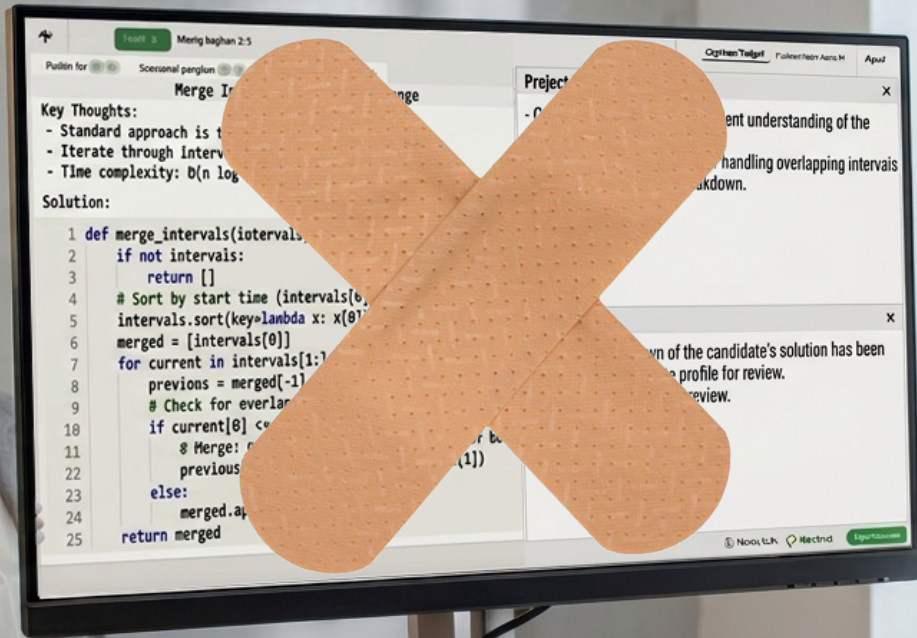
```
1 def merge_intervals(intervals):
2     # Pass intervals, where each is a list of two integers
3     # [start, end], where start < end
4     return []
5
6     # Sort intervals based on the starting line of each interval
7     intervals.sort(key=lambda x: x[0])
8
9     # Initialize the merged intervals list
10    merged = []
11
12    # Iterate over the sorted intervals
13    for interval in intervals:
14        # Check if the current interval overlaps with the last merged interval
15        # If it does, update the end of the merged interval
16        # If not, add the current interval to the merged list
17        if merged and interval[0] <= merged[-1][1]:
18            merged[-1][1] = max(merged[-1][1], interval[1])
19        else:
20            merged.append(interval)
21
22    # Return the merged intervals
23    return merged
24
25
```

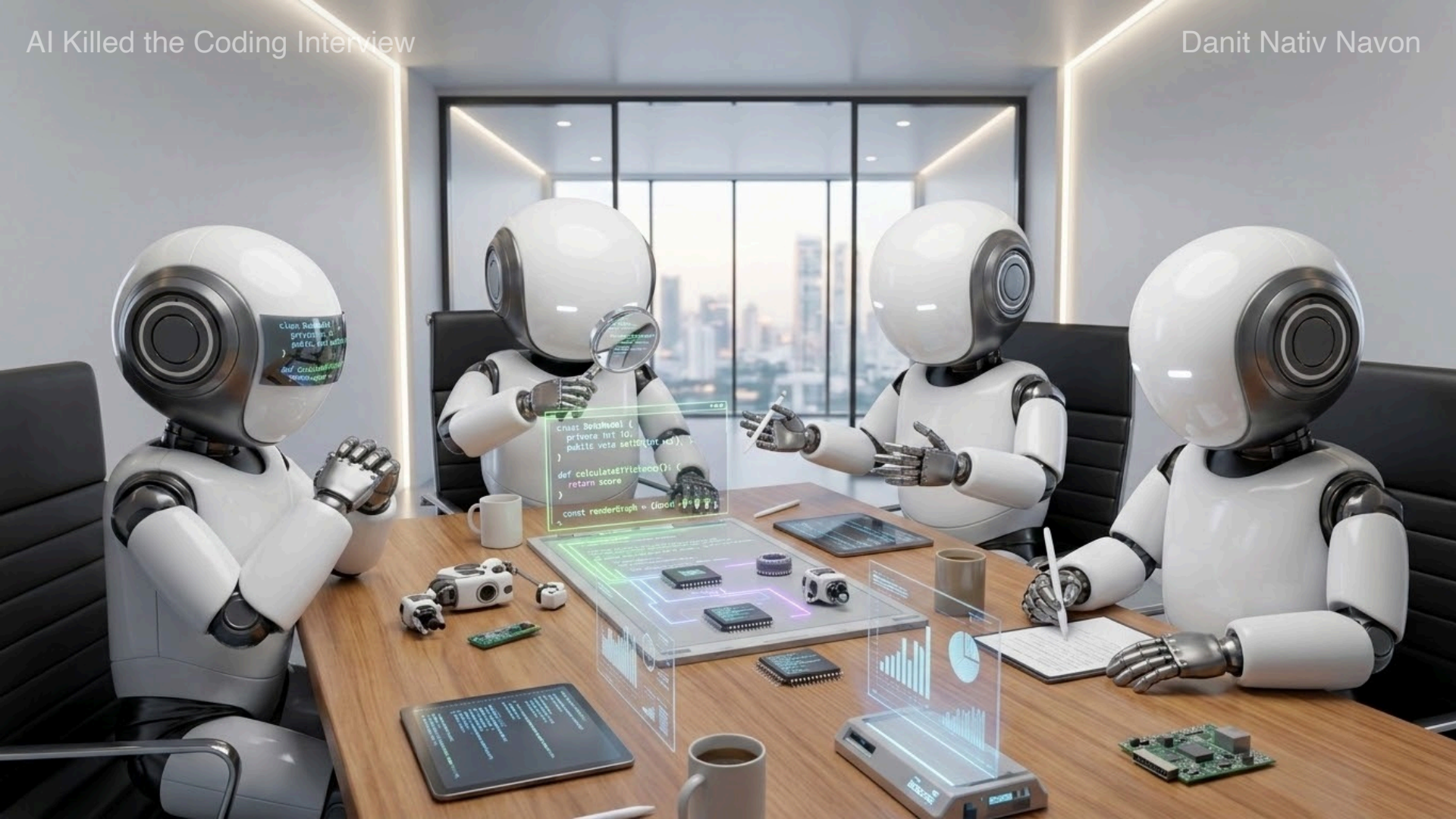
Project Dashboard: Merge Intervals

Mail events | Filter items | Notifications | Close

Comgality

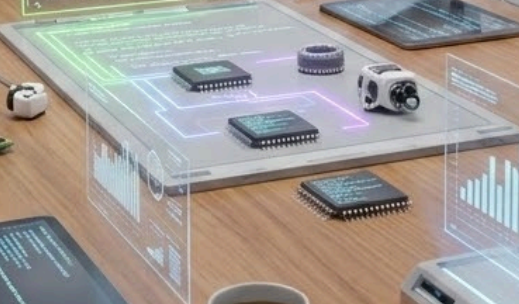






```
Client: Rollback  
server: Rollback  
public void rollback()  
def rollback()  
return true
```

```
class Bot {  
private int id;  
public void setid(int id) {}  
}  
def calculateScore():  
return score  
const renderGraph = () => {}
```



```
const data = [...]  
const chart = [...]  
const table = [...]
```

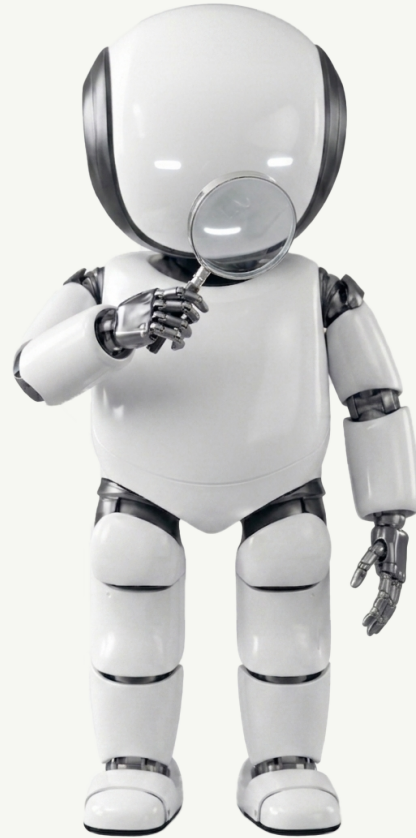


The image shows a computer monitor displaying a coding IDE interface. The interface is divided into several sections:

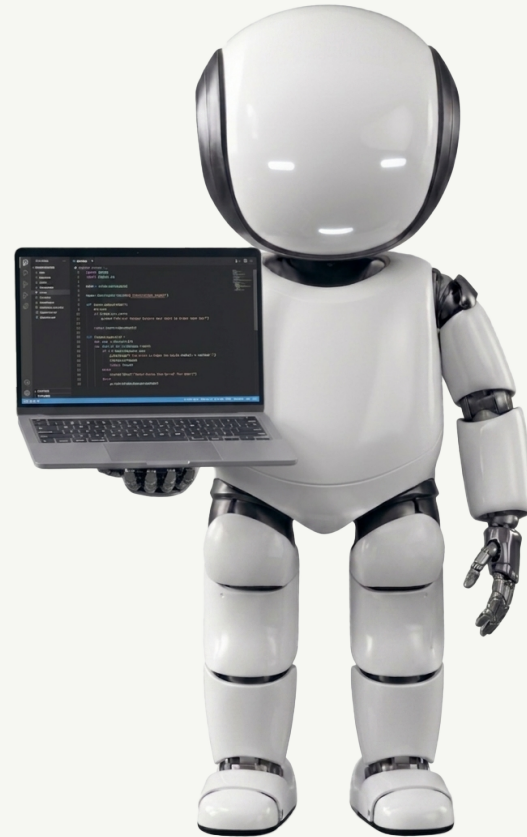
- Files Panel (Left):** Shows a project structure with folders like 'generic_project/' and 'sub_folder_A/'. Files include 'file1.txt', 'sub_file_2.py', 'file_2.py', 'notes.md', 'README.txt', 'configuration_v1.config', 'utility.py', 'folder1/', and 'file3.txt'.
- Code Editor (Center):** Displays Python code for a test scenario. The code includes imports, class definitions, and test methods. A 'Run Main' button is visible above the code.
- Code Execution Panel (Top Right):** Contains tabs for 'Instructions', 'Program Output', 'Private Interviewer Notes', 'Solutions', and 'AI Aeslet'.
- AI Assistant Panel (Right):** Features the Luma logo and the text 'AI Assistant', 'Ask Meta...', and 'Need help understanding, fixing, or improving your code? Ai can assist, but be sure to review the results critically.' Below this is a search bar with 'Add Coolest...' and 'File X' buttons, and a 'Luma 4 Maverick' dropdown.
- Bottom Bar:** Includes 'Invite', 'Danit N', 'Untitled Ped - WFZEXRTO', 'What's New', 'Feedback', and 'End Interview' buttons.

```
1 import unittest
2
3 class TestScenario(unittest.TestCase):
4     """
5     This is a generic template for writing tests in this environment.
6     Use this structure to test your own code logic.
7     """
8
9     def setUp(self):
10        # Set up test environment and initialize variables here.
11        # Examples: self.app = MyClass(), self.db = Database()
12        pass
13
14    def test_context(self):
15        """
16        Generic test case to verify a core aspect of your feature.
17        """
18        // here is the code context:
19        // You should implement your code logic and tests below.
20
21        # Define test inputs and conditions
22        input_data = "generic_data"
23        expected_result = "processed_data"
24
25        # Execute the function to be tested
26        actual_result = input_data # Placeholder logic
27
28        # Use assertions to verify correctness
29        self.assertEqual(actual_result, expected_result)
30
31    def test_additional_case(self):
32        """
33        Example for another test scenario.
34        // here is another code context:
35        // Feel free to add multiple test cases like this one.
36
37        # Add test logic with different inputs and assertions
38        pass
39
40 if __name__ == '__main__':
41     unittest.main()
```

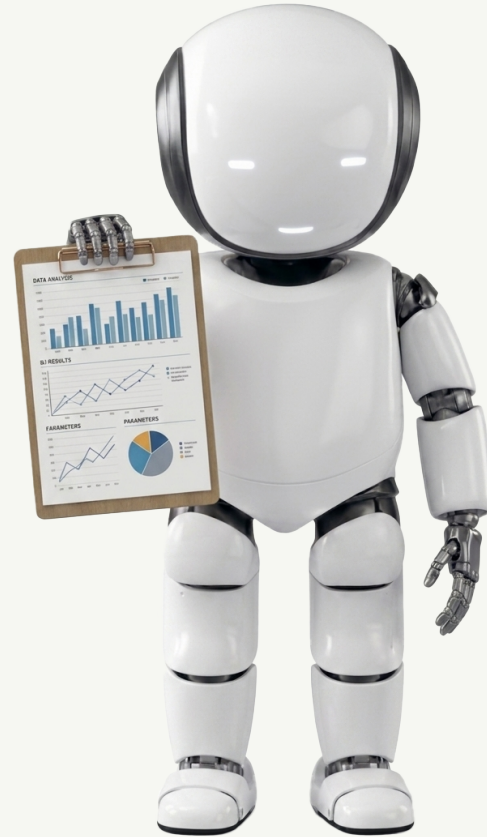
Problem Exploration



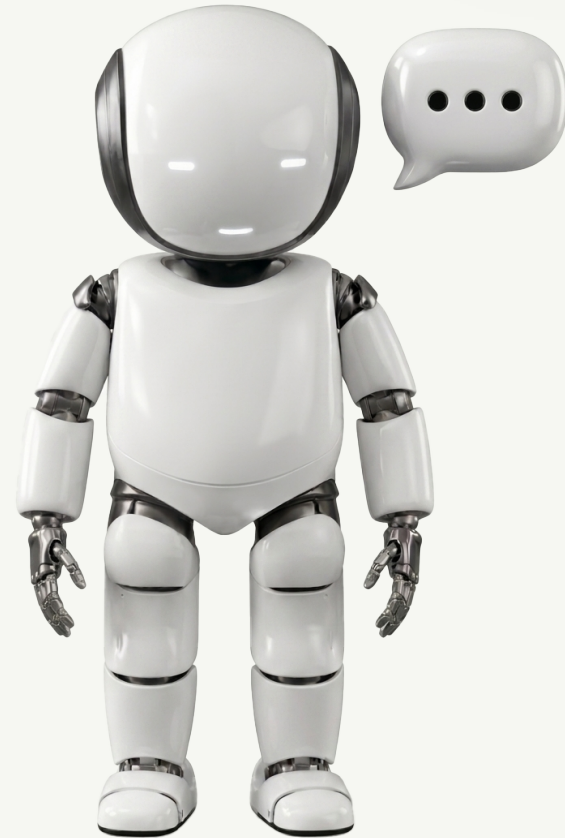
Coding



Validation



Communication



Sep 2025

First Interview

9,000+

Interviews

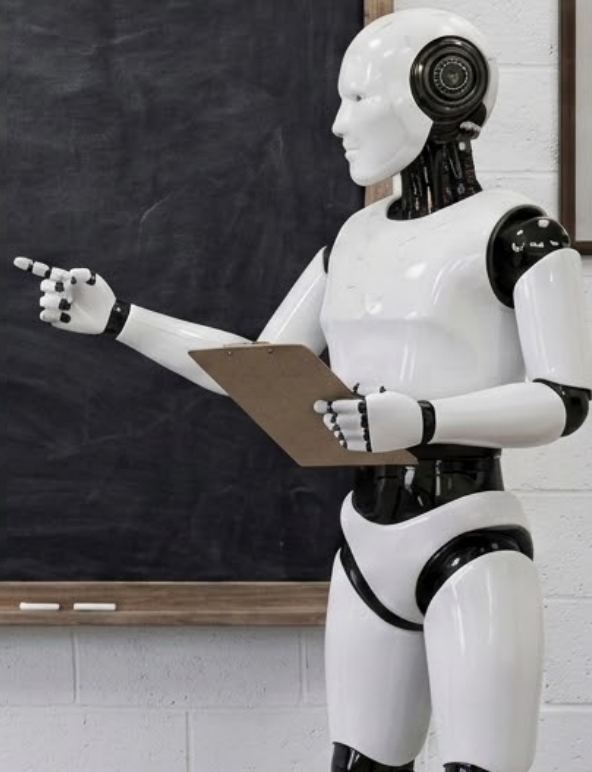
5,000

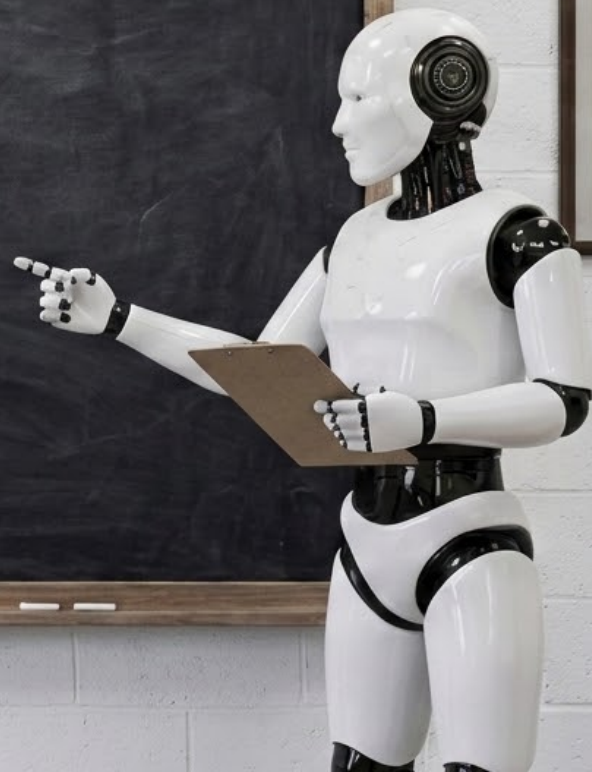
Trained
Interviewers

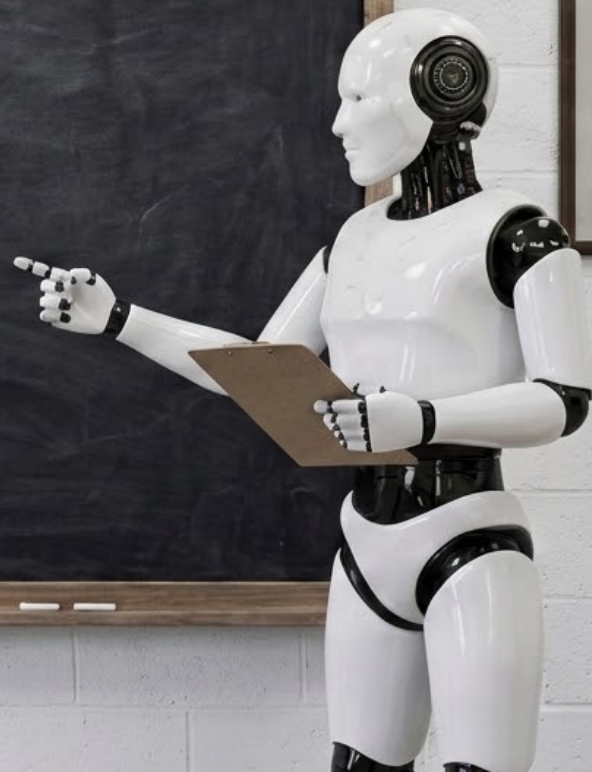
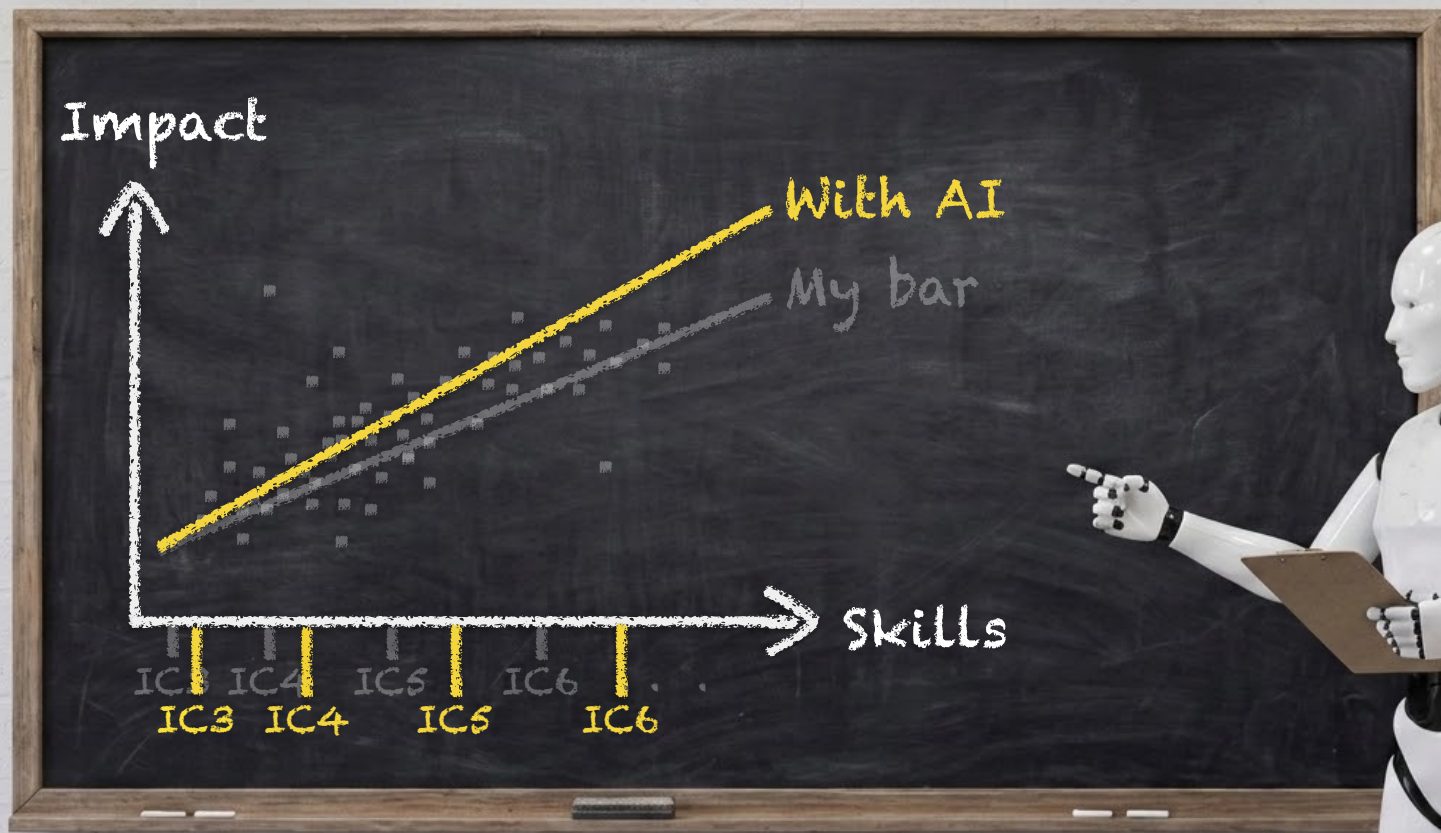
Apr 2026

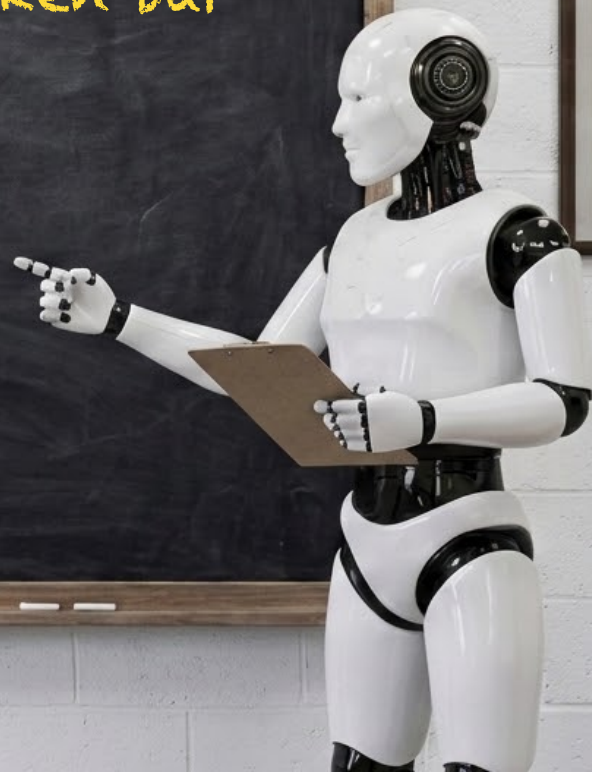
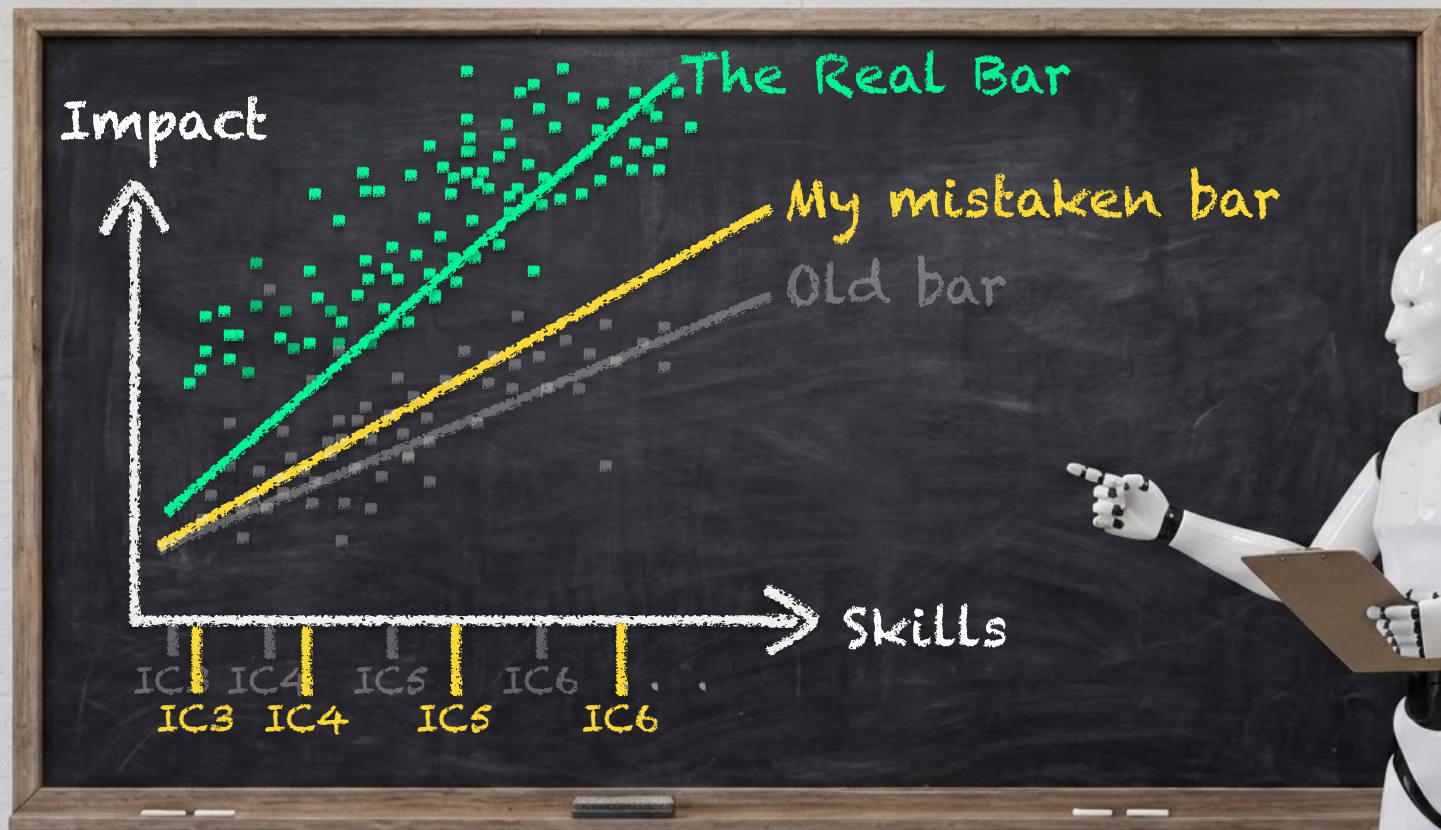
Fully AI-Native
Interviews

THE WAY
WE HIRE











ARTIFICIAL INTELLIGENCE: THE FUTURE IS NOW

NEW HIRE ORIENTATION





**Give the microphone
to the power users**

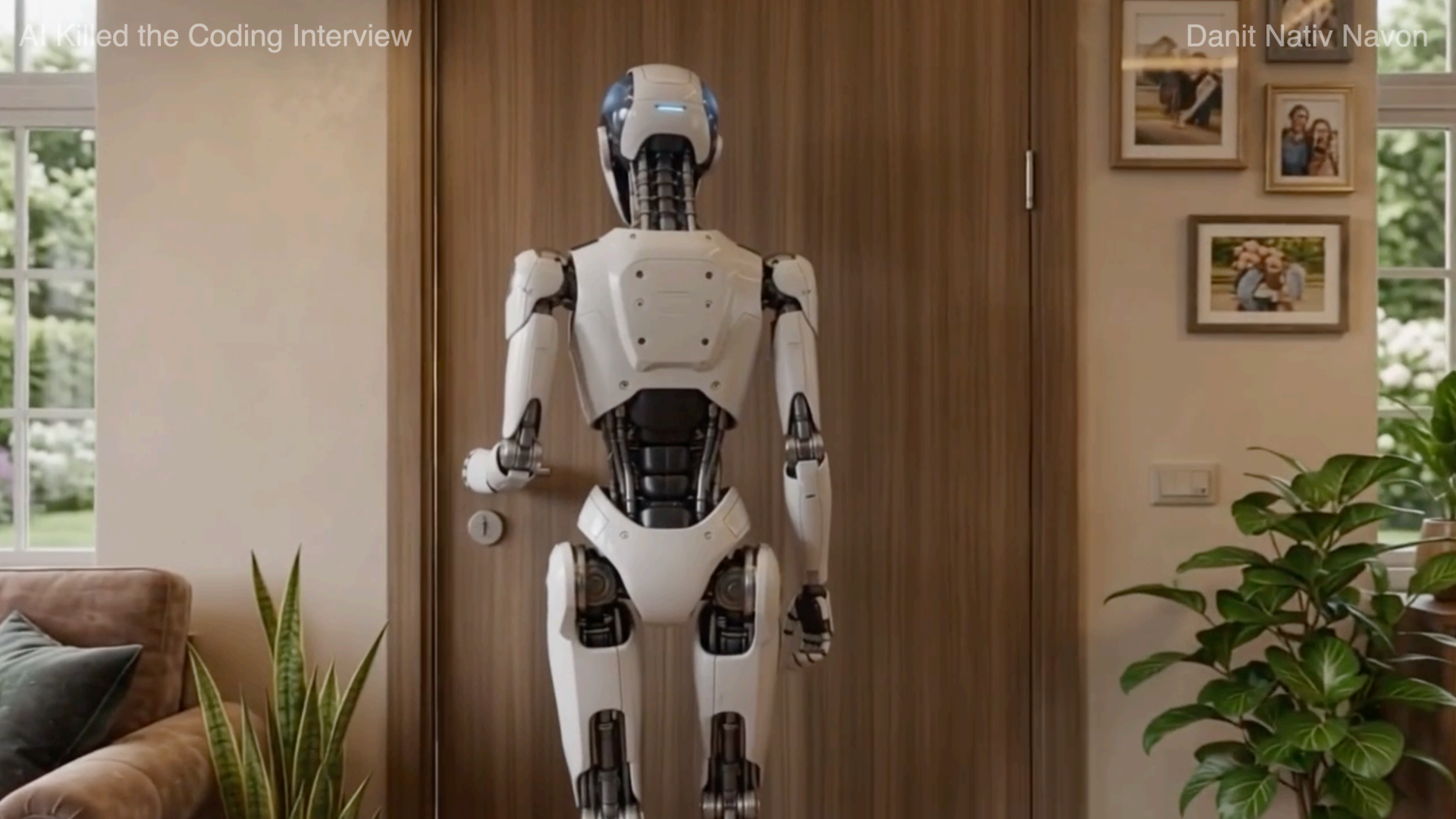


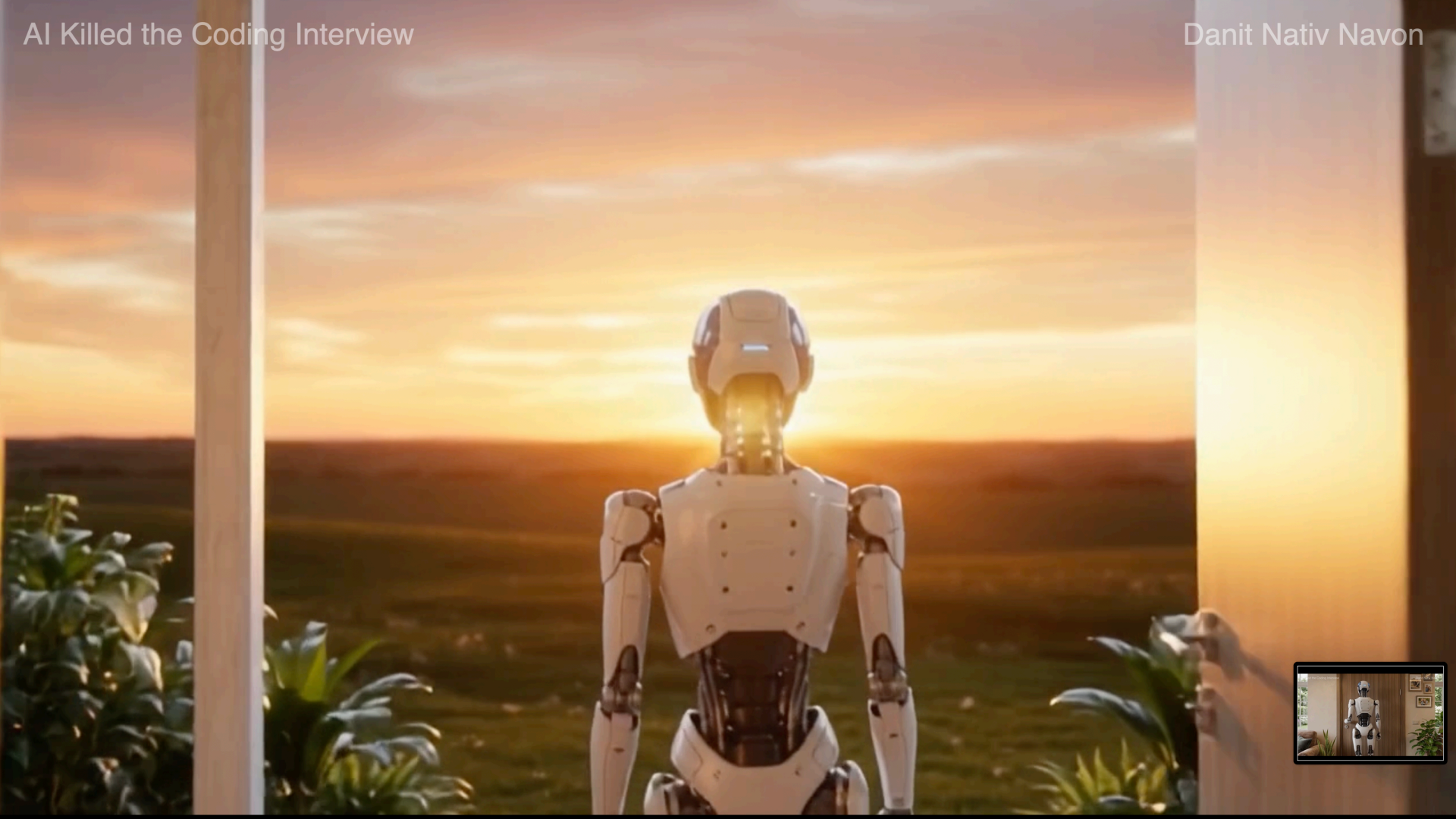
**Make it a
team effort**



**Provide time and
resources to learn**







A sunset over a field with the text "Thank you" overlaid. The sun is low on the horizon, creating a bright glow and lens flare. The sky is filled with soft, wispy clouds in shades of orange and yellow. The foreground shows a dark, silhouetted field with some small white flowers.

Thank you