

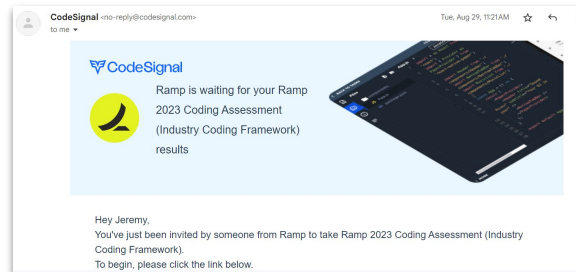


Mockmate: AI Powered Technical Job Interview Preparation

Rudra Barua, Andrew Sima, Jeremy Zhang
AC215, Fall 2023

Motivation

- Inefficient traditional interview prep
 - Requires domain expertise in interviewer
 - Inconsistent feedback
 - Misses nuances of specific roles or industries
 - Easy to over-prepare, under-prepare, or prepare for the wrong skills
- Provide immediate, actionable, and unbiased feedback
- Identify weaknesses in specific areas
- Breakthroughs in LLMs allow for personalization
- We're graduating seniors that need jobs

A screenshot of a coding assessment dashboard. The top navigation bar includes tabs for "All Topics", "Algorithms", "Database", "pandas", "JavaScript", and "Shell". Below this is a table of questions with columns for "Status", "Title", "Solution", "Acceptance", "Difficulty", and "Frequency". The table lists 7 questions, including "Find Largest Value in Each Tree Row", "Add Two Numbers", "Longest Substring Without Repeating Characters", "Median of Two Sorted Arrays", "Longest Palindromic Substring", "Zigzag Conversion", and "Reverse Integer". To the right of the table is a sidebar with a "Pick One" button and a list of companies with their respective scores: Facebook (60), Bloomberg (60), Amazon (50), Apple (40), TikTok (40), Microsoft (40), Salesforce (40), Google (40), Uber (40), Walmart Labs (40), Adobe (40), PayPal (40), Goldman Sachs (40), Oracle (40), Citibank (40), Nvidia (40), Snapchat (40), Doordash (40), and JPMorgan (40).

Problem Statement

- **Problem definition:**
 - Provide an automated yet personalized interview preparation experience for prospective software engineers
- **Project goals:**
 1. Generate technical questions relevant to software engineering
 2. Evaluate candidate responses in real-time
 - Offer constructive insights on coding efficiency and response quality
 3. Provide insights into candidate's weaknesses and strengths



Data and Preprocessing

- **Source:** Original questions from Leetcode, scraped into a Kaggle dataset [here](#).
 - ~2360 problem statements and sample solutions (in plaintext and a handful of programming languages)
- **Preprocessing:**
 - Separated problem components with Python solutions
 - Created custom PyTorch dataset that guarantees fixed-length token chunks
 - Separated data into training and validation sets to train the model

```
▼ 0:
  id:      "1"
  slug:    "two-sum"
  difficulty: "Easy"
  ▶ cpp_sol:  """cpp\n#include <vector>... return {};\n}\n""`\n"
  ▶ java_sol:  """java\nimport java.util... solution");\n}\n""`\n"
  ▼ python_sol:  """python\ndef twoSum(nums, target):\n    map = {}\n    = i\n    return []\n""`\n"
  ▶ javascript_sol:  """javascript\nfunction ... return [];\n}\n""`\n"
  ▶ explanation:  "The algorithm leverages _lexity of O(n) as well."
  ▼ question:  "Given an array of integers `nums` and an integer `target` may not use the _same_element twice.\n\nYou can return t

▼ examples:
  ▶ 0:  """Example 1:**\n\nInput: we return \[\[0, 1\]\].""
  ▶ 1:  """Example 2:**\n\nInput:\nOutput:** \[\[1, 2\]\].""
  ▶ 2:  """Example 3:**\n\nInput:\nOutput:** \[\[0, 1\]\].""
  ▶ constraints:  "2 <= nums.length <= valid answer exists.""
  ▶ followup:  "Can you come up with an _O(n^2)` time complexity?"
  ▶ 1:  {}
  ▶ 2:  {}
  ▼ 3:
    id:      "4"
    slug:    "median-of-two-sorted-arrays"
    difficulty: "Hard"
    ▶ cpp_sol:  """cpp\ndouble findMedia... return 0;\n}\n""`\n"
    ▶ java_sol:  """java\npublic double f... return 0;\n}\n""`\n"
    ▶ python_sol:  """python\ndef findMedia... \n    return 0\n""`\n"
    ▶ javascript_sol:  """javascript\nfunction ... return 0;\n}\n""`\n"
    ▶ explanation:  "1. Choose the smaller ar_ged array, even on odd."
    ▶ question:  "Given two sorted arrays _ould be `O(log (mn))`."
    ▶ examples:  [-]
    ▶ constraints:  "nums1.length == m`_s1[i], nums2[i] <= 10^6`"
    followup:  null
```



Model Configuration & Training

- **Base Model:** Initialized pre-trained LLMa model with Meta's `open_llama_3b_v2`
- **Low-Rank Adaption (LoRA):** Adapted the model using LoRa, a method designed to fine-tune models more effectively
 - Factorizes new parameters and reuses pre-trained ones
- **Optimized Loading:** Loaded in models in quantized format
 - Quantized model weights to reduce memory consumption
- **Regularization:** Set maximum token generation limit per query
 - Prevents overly verbose outputs and overfitting on certain query patterns



Model Querying


- Created CLI for continuous input queries for the model
- Implemented tokenizer to format user text
- Converted tokenized inputs to PyTorch tensors to CUDA
 - GPU acceleration in response generation
 - Produce tokenized response tensors



Future Work


- This week: finalize, optimize, and refine AI algorithms
- M5 (11/14): Create APIs, fullstack and frontend development
- M6 (12/12): Scaling and production deployment

Mockmate AI Powered Software Interview





Please write the code for FizzBuzz, and explain your thinking as you write.


19/25 cases passed.
View report.



Write a program to find the shortest path between point A and point B in a given graph.

So uhh... the first thing we uhh... want to do is uhh... use BFS





Pavlos is currently interviewing you.

```
// Basic Java example using FizzBuzz
//
import java.util.Random;

public class Example {
    public static void main(String[] args){
        // Generate a random number between 1-100. (See generateRandomNumber method.)
        int random = generateRandomNumber(100);

        // Output generated number.
        System.out.println("Generated number: " + random + "\n");

        // Loop between 1 and the number we just generated.
        for (int i=1; i<=random; i++){
            // If i is divisible by both 3 and 5, output "FizzBuzz".
            if (i % 3 == 0 && i % 5 == 0){
                System.out.println("FizzBuzz");
            }
            // If i is divisible by 3, output "Fizz"
            else if (i % 3 == 0){
                System.out.println("Fizz");
            }
            // If i is divisible by 5, output "Buzz".
            else if (i % 5 == 0){
                System.out.println("Buzz");
            }
        }
    }
}

src/components/Banner.test.js
Banner component
✓ renders the title (27 ms)
✓ renders the subtitle (4 ms)
✓ renders the image with the correct alt text (3 ms)
✓ hides the image when hidden prop is true (21 ms)

Test Suites: 1 passed, 1 total
Tests: 4 passed, 4 total
Snapshots: 0 total
Time: 2.103 s
Run all test suites matching /Banner.test.js/i.

Watch Usage: Press w to show more.
```





Harvard John A. Paulson
School of Engineering
and Applied Sciences

Thank you!