Picobytes: Execute and Test

Skills Acquired:

1. Container creation and management using Docker
2. Managing multiple containers using Kubernetes
3. Creating an isolated environment for running external code
4. Compiling, running and testing C code sent to our server
5. Collaboration with different teams
6. Effective system design

Lessons Learned:

1. More than expected amount of things need to be accounted for when integrating a front and back end
2. Don't wait until the end to actually time the responses from API if doing something intensive on the back end
3. AI can generate some really useless slop code that seems like it works until heavily utilized
4. There's always something that can be added or improved upon
5. Stay away from monolithic design choices

Goals Accomplished:

1. Isolated environment for running user submitted code
2. Compiling, executing and testing user submitted code
3. Valgrind analysis
4. API to communicate with our service
5. Customizable options in the api on how the code should run. Examples: custom time limits, custom tests, whether to run the tests or not
6. Blacklisting functions: functions that are not allowed to be used in the code
7. Whitelisted functions: only functions that can be used in the code
8. Resource limiting: amount of memory program can use, network access, time limits
9. Timing how long each step such as : compilations, execution took
10. Parsing the output to return a line by line analysis on the code
11. Parsing the valgrind output to be more readable by the user
12. Decoupling the API and computation allowing us to scale each part independently depending on the request load and load type
13. Autoscaling of resources based on the workload
14. Return a list of tests that failed
15. Automatic recovery in case any components fail
16. Clean up after each job is completed
17. Logging
18. Documentation

Goal Not Accomplished:

1. Request verification using API keys or other methods
2. Parallel execution of tests and other analysis tools such as valgrind
3. Different flags to change the behavior such as : treat warnings as error
4. Internally hardening containers through kubernetes
5. Deployment on a cloud server