General Approach to Updating the Benchmark Test (Conceptual):

Assuming the applications:benchmark:run task is based on a benchmarking framework (like JMH, or a custom script), you would typically:

1. Locate the Benchmark Test File: Find the relevant benchmark test file within the applications/benchmark/ directory of the Email Verifier codebase. This file would likely be written in Java, Kotlin, or Groovy, depending on the Gradle project's setup.
2. Identify Throughput Metrics: The benchmark test's results would need to include a metric for throughput (e.g., operations per second, requests per second). You'd identify how this metric is reported.
3. Add a Threshold Check:
   * In Code: If it's a code-based benchmark, you would add an assertion or a conditional check at the end of the test.
     + Retrieve the measured throughput.
     + Compare it against your target (50 registrations per second).
     + If the measured throughput is less than 50, you would programmatically throw an exception, fail an assertion, or log an error message and return a non-zero exit code to indicate a build failure.
   * In Gradle Script: Sometimes, thresholds can be configured directly in the build.gradle file for specific benchmark plugins. You would look for options to set a minimum throughput or maximum latency and configure the build to fail if these thresholds are not met.
4. Error Message: Ensure the failure message is clear, indicating *why*the test failed (e.g., "Throughput requirement of 50 registrations/sec not met. Actual: X registrations/sec").

Conceptual Example (if using a hypothetical custom benchmark runner in Java/Groovy):



Generated code may be subject to license restrictions not shown here. Use code with care. [Learn more](https://cloud.google.com/duet-ai/docs/discover/code-generation-source-citation#how_helps_with_coding)

You would then run ./gradlew applications:benchmark:run as specified, and it would likely fail due to the throughput not meeting the 50 registrations/sec requirement, as intended by the problem statement.