README (Updated)

Project: Randomness Testing System (Upgraded Flask + Bootstrap 5)

Overview  
This project provides a web-based suite for generating random numbers from multiple sources and evaluating their statistical quality using several tests. The upgraded version includes:

* Full English UI with Bootstrap 5 (LTR)
* Dedicated pages: direct generator and full statistical tests
* Background test execution with live progress and status polling
* Structured logging

Main Features

* Random generators:
  + Python Generator (import random)
  + Java Threads Generator (Thread scheduling entropy)
  + Time Nano Generator (time-based)
  + Sound Generator (microphone-based; requires hardware)
* Statistical tests:
  + Frequency (Monobit), Runs, Chi-Square (bytes)
  + Serial tests (pairs, triplets)
  + Autocorrelation (lag=1, lag=2)
  + Poker test (group sizes 4, 5)
  + Maurer’s Universal Test (L=7)
* Async tests with status endpoint and UI progress
* Error logging for both Flask app and generators
* Pytest-based automated unit tests

Repository Structure

* app\_updated\_en.py — Flask server (English UI, endpoints, background tasks)
* [generators.py](http://generators.py) — Random generators implementation + factory
* tests\_module.py — Statistical tests
* test\_generators.py — Pytest for generators and factory
* test\_tests\_module.py — Pytest for tests\_module
* templates/
  + base.html — Base layout (Bootstrap 5)
  + direct.html — Direct generator page
  + full\_tests.html — Statistical tests page (with Restart)
  + 404.html — Not Found template
  + 500.html — Server error template
* static/js/
  + app\_en.js — Client-side logic (validation, status polling, toasts)
* logs
  + generator\_errors.log — Generator-related errors
  + flask\_app\_errors.log — Server errors
* MyRandomProject.java — Java-based generator (threads race)
* README.docx (this file; updated)

What’s New in the Upgraded Version

* English LTR UI with Bootstrap 5
* Improved progress reporting:
  + Frequent status updates for smoother UX
* Robust handling of stop requests and background task cleanup
* Better validations, tooltips, and user feedback via toasts

Installation & Setup

Prerequisites

* Python 3.8+
* pip packages: flask, numpy, scipy, pyaudio, pytest
* Java (for Java generator)
* Microphone (to use Sound generator)
* Windows/macOS/Linux support

Install Python dependencies  
pip install flask numpy scipy pyaudio pytest

Project paths  
Make sure PROJECT\_DIR and log/result file paths in:

* app\_updated\_en.py
* [generators.py](http://generators.py)
* tests\_module.py  
  point to the correct local project directory.

Run the server  
python app\_updated\_en.py

Open the UI

* Direct generator page: <http://localhost:5000/>
* Statistical tests page: <http://localhost:5000/tests>

Using the Application

Direct Generator (Home)

* Select generator and upper bound
* Submit to generate an instant random value
* Sound generator requires functional microphone access
* Java generator requires Java installed and on PATH

Statistical Tests

* Go to <http://localhost:5000/tests>
* Select a generator, test type, upper bound, and samples
* Start test to run in the background
* Live progress bar with status (reaches 100% upon completion)
* Test results displayed with generator name and formatted details
* Stop Test button to request cancellation

API Endpoints

GET/POST /

* Main direct generator page

GET /tests

* Full statistical tests UI

POST /start\_test

* Start a background randomness test
* Form fields: generator, test\_type, upper\_bound, samples
* Returns: task\_id

POST /stop\_test

* Stop a running test task
* Form field: task\_id

GET /status/<task\_id>

* Get current status and final result for a specific test task
* Returns: status, done, result (and handled internally with generator name)

Statistical Tests Details

* frequency: Monobit frequency balance of 0/1
* runs: Number of consecutive identical bits and deviation
* chi\_squared\_full\_test (bytes): Distribution uniformity across byte values
* serial\_test (pairs/triplets): Pattern frequency uniformity
* autocorrelation\_test (lags 1/2): Dependency across positions
* poker\_test: Group distribution uniformity (4/5-bit groups)
* maurer\_universal\_test (L=7): Compressibility/unpredictability (requires long sequences)

Automated Testing (Pytest)

* Run generator tests:  
  pytest test\_generators.py
* Run statistical tests:  
  pytest test\_tests\_module.py

Troubleshooting

Templates not found (direct.html/404.html/500.html)

* Ensure templates are under templates/ with exact filenames
* Verify Flask app is running from the project root

Static JS not found (static/js/app\_en.js)

* Ensure the JS file exists at static/js/app\_en.js
* Confirm the script tag in templates references the correct path

Java generator errors

* Ensure Java is installed and accessible in PATH
* Ensure MyRandomProject.java is compiled or callable by the subprocess
* Check generator\_errors.log for the command output/return code

Sound generator issues

* Requires microphone and PyAudio
* Ensure the microphone device is accessible and not blocked by OS permissions
* Check generator\_errors.log for initialization or read errors

Progress not reaching 100%

* Upgraded logic ensures 100% on completion; if persisting:
  + Check that /status/<task\_id> returns done: true
  + Confirm that front-end JS sets progress to 100% when done is true
  + Review flask\_app\_errors.log for exceptions during finalization

Stop test not stopping immediately

* Stop is cooperative; the task checks for stop signal between sample iterations
* For long tests with high samples, allow a short delay to observe “Stopped by user” status

Extending the Project

Add a new generator

* Implement class in [generators.py](http://generators.py)
* Add to generator\_factory mapping
* Update UI selects in full\_tests.html and direct.html

Add a new test

* Implement function in tests\_module.py (return dict with stats, p-value, passed)
* Add routing/rendering in app\_updated\_en.py
* Update UI selects and result formatting as needed

Notes on Logging

* Server errors: flask\_app\_errors.log
* Generator errors (Java/Sound/Time/Python): generator\_errors.log
* Statistical test exceptions: test\_errors.log (as configured inside tests\_module.py)

Credits

* Author: Saar Weinberg (סער וינברג)
* Email: [saarwai@gmail.com](mailto:saarwai@gmail.com)

Version

* Last upgrade: English UI + Bootstrap 5, Restart button, accurate 100% progress, generator name in results, improved validation and UX.

1. app\_en.js
2. MyRandomProject.java

1. [script.py](http://script.py)
2. script\_1.py
3. test\_generators.py
4. test\_tests\_module.py
5. tests\_module.py
6. README.docx
7. 404.html
8. 500.html
9. base.html
10. direct.html
11. full\_tests.html
12. app\_updated\_en.py
13. generator\_errors.log

1. [generators.py](http://generators.py)