Smart Email Guardian – Deployment & Verification Guide

Objective

This document provides instructions for deploying and verifying your AI-powered spam/phishing detection toolkit. The goal is to ensure your application is accessible, functional, and secure—without requiring paid services.

What You Must Submit

- A public GitHub repository with the full project
- A working CLI tool (`email_guard.py`) that accepts input and returns results
- A deployed web or mobile interface
- A hosted backend API with a /scan endpoint
- A demo video if mobile app cannot be hosted
- A `README.md` and `reflection.md` with documentation and what you learned

Free Hosting Platforms

You can use these platforms to deploy your application: - **Frontend**: Vercel – great for React, Flask UI, or Streamlit - **Backend**: Render, Railway, or Replit - **Mobile**: Use Expo and share the live preview or a video recording

What We'll Check (Review Checklist)

- A live frontend app (via Vercel or demo video)
- A working backend with `/scan` endpoint
- Secure communication (HTTPS hosting or localhost tunnel)
- Clear `README.md` for setup and testing
- Proper folder structure and clean code
- Documented security practices in 'security_not

Hosting Your Project (Frontend Example)

To deploy a frontend on Vercel: 1. Push your frontend code to GitHub 2. Go to https://vercel.com and sign in 3. Import your GitHub project 4. Vercel will auto-deploy and give you a public link 5. Share this link in your `README.md`

Hosting Your Backend (Example Using Render)

1. Push your backend code (Flask/FastAPI) to GitHub
2. Go to https://render.com and sign in

3. Create a new web service → link your GitHub repo
4. Set start command (e.g., `python app.py`) and expose port 10000 or 5000
5. Render will give you a public URL
6. Add this to your frontend and documentation