Spam Detector Toolkit Document

1. Overview of the Chosen Technology

This project uses **HTML**, **CSS**, and **JavaScript** to build a simple, browser-based Spam Detection tool. The spam detector checks text messages for known spam keywords and displays whether the message is likely to be spam.

Why These Technologies?

- **HTML** provides the structure of the web page.
- **CSS** styles the interface.
- **JavaScript** adds functionality and performs the spam detection logic.
- No backend, databases, or external APIs are required.
- Beginner-friendly and runs directly in any web browser.

How the Spam Detection Works

The JavaScript script searches the input message for a list of predefined spam keywords such as *"win money", "free", "lottery", "click here"*. If any of these appear, the tool marks the message as spam and calculates a spam score.

---

2. Setup Instructions

Follow these steps to set up the project locally:

Step 1: Create Project Folder

Create a folder named **spam-detector**.

Step 2: Add Required Files

Inside the folder, create: - `index.html` - `style.css` - `script.js`

Step 3: Copy the Code

Paste the provided HTML, CSS, and JavaScript code into the corresponding files.

Step 4: Run the Project

- Open the **index.html** file using Chrome, Edge, or Firefox.
- The spam detector will load instantly, no installation required.

---

3. Minimal Working Example (Hello World)

Before building the spam detector, here is a minimal JavaScript "Hello World" example:

```html
<!DOCTYPE html>
<html>
  <body>
    <h1>Hello World Test</h1>
    <button onclick="alert('Hello, World!')">Click Me</button>
  </body>
</html>
```

Opening this file in a browser will show a button that displays "Hello, World!" when clicked.

---

4. Minimal Working Example of Spam Detector

Once the setup is complete, this simple version demonstrates the core functionality:

```javascript
const message = "Congratulations, you won a free prize!";
const spamWords = ["free", "prize", "win", "click here", "lottery"];

let isSpam = false;
spamWords.forEach(word => {
  if (message.toLowerCase().includes(word)) {
    isSpam = true;
  }
});

console.log(isSpam ? "Spam Detected" : "Not Spam");
```

This verifies that the keyword-matching logic works.

---

5. AI Prompts Used

A cleaned and organized list of AI prompts used during development (ChatGPT conversation references): - "Create a simple HTML + JavaScript spam detector project." - "Guide me step-by-step as a beginner." - "Add spam score percentage." - "Add detected spam words list." - "Prepare toolkit documentation for submission."

(Additional prompts can be added after final review.)

---

6. Learning Reflections

This project provided hands-on experience with:

✔️Basic Web Development

I learned how HTML structures a page, CSS makes it visually appealing, and JavaScript makes a webpage interactive.

✔️Understanding Conditional Logic

The spam detection is based on checking if words exist in a message. This helped reinforce `if` statements, loops, arrays, and string methods.

✔️Debugging Skills

I learned to test code, inspect browser console errors, and fix bugs like missing IDs or broken links.

✔️Building End-to-End Projects

This was my first project combining multiple files into a complete functional tool.

---

7. Common Errors & How to Resolve Them

1. Button Not Working

Cause: JavaScript not linked correctly.
Fix: Ensure this line exists before the closing `body` tag:

```
<script src="script.js"></script>
```

2. CSS Not Applying

Cause: Wrong filename or missing link tag.
Fix:

```
<link rel="stylesheet" href="style.css">
```

3. spamWords Not Detected

Cause: Words not matched due to uppercase letters.
Fix: Convert message to lowercase:

```
message.toLowerCase()
```

4. Nothing Shows on Screen

Cause: HTML structure incomplete.
Fix: Ensure `index.html` has correct tags and is saved in the same folder.

5. Browser Shows Old Code

Cause: Browser cached files.
Fix: Press **CTRL + SHIFT + R** to hard refresh.

---

8. Reference Resources

Authoritative and helpful links used during development: - MDN Web Docs – JavaScript: https://developer.mozilla.org - MDN HTML Guide - MDN CSS Guide - W3Schools (Beginner tutorials): https://www.w3schools.com - GitHub Docs: Repository setup and README formatting

---

9. Final Notes

This toolkit document supports the spam detector project by explaining the technologies, showing how to run the project, and providing troubleshooting guidance. It is suitable for submission as part of a beginner-level technical coursework project.