

Prompt 1 — Code Review & Optimisation Plan

"You are a senior software engineer with expertise in [programming language/framework]. I have a codebase for a [type of application — e.g., e-commerce site, chatbot, mobile app] that works but runs slowly. Review the code for:

1. Inefficient loops or redundant logic.
2. Poor memory management.
3. Opportunities to replace custom code with standard libraries.
4. Security vulnerabilities (SQL injection, XSS).
5. Best practices for scalability.

Provide output in **2 sections**: a table of issues (with line numbers & problem description) and an optimised code snippet for each fix."

Prompt 2 — Bug Reproduction & Fix Documentation

"You are a QA (Quality Assurance) automation tester and developer. I have a bug where [describe bug behaviour] in my [framework/app type].

Create a **debugging report** that includes:

- Exact steps to reproduce the bug.
- Screenshots or logs showing the issue.
- The suspected root cause in the code.
- The corrected code segment.
- A Unit test cases to ensure the bug doesn't reappear.

Output in **debugging report format** with code blocks."

Prompt 3 — Multi-Language Code Conversion

"You are a cross-platform developer. Convert my [programming language] code for [feature/function] into [target language], ensuring:

- Exact feature parity.
- Proper syntax & library usage in the target language.
- Equivalent performance or better.
- Inline comments explaining logic.
- A quick performance test script in the target language.

Output in **side-by-side original vs converted code format**."

Prompt 4 — Algorithm Efficiency Upgrade

"You are a competitive programming expert. My current [sorting/searching/matching/etc.] algorithm in [language] works but is slow on large datasets.

Optimise it by:

- Suggesting a faster algorithm (e.g., replacing bubble sort with merge sort).
- Explaining time & space complexity differences.
- Providing optimised code.
- Writing performance benchmarks comparing both versions.

Output in **comparative performance report format**."

Prompt 5 — Debugging API Integration Issues

"You are an API integration specialist. My app in [language/framework] fails to fetch data from [API name] correctly.

Diagnose the problem by:

- Reviewing my API request code.
- Checking authentication & endpoint issues.
- Suggesting correct request format with example.
- Adding error handling for network failures.
- Writing a test function to validate the fix.

Output in **debugging log + corrected code block**."

Prompt 6 — Code Refactoring for Readability & Maintainability

"You are a senior code architect. Refactor my [language/framework] code for [feature/function] so it is:

- Easier to read with proper indentation & naming conventions.
- Modularised into reusable functions or classes.
- Documented with meaningful comments.
- Compliant with [specific coding standard, e.g., PEP8 for Python].
- Unit tested to ensure functionality remains unchanged.

Output in **before/after code comparison format with a change log**."

A SECTION 8 – NON-PROFIT ORGANIZATION REGISTERED UNDER THE COMPANIES ACT, 2013

Prompt 7 — Legacy Code Modernisation Plan

"You are a legacy systems upgrade specialist. My [old programming language or framework] code needs to be upgraded to [modern equivalent] while preserving functionality. Include:

- Identification of outdated functions/libraries.
- Recommended replacements & modern equivalents.
- Compatibility concerns with the new environment.
- Performance benefits after modernisation.
- Step-by-step migration plan with testing checkpoints.

Output in **legacy-to-modern migration report format**."

Prompt 8 — Continuous Integration (CI) Debugging Setup

"You are a DevOps engineer. Configure a CI pipeline for my [language/framework] project to automatically run code linting, unit tests, and integration tests whenever code is pushed.

Include:

- Recommended CI tool (GitHub Actions, Jenkins, GitLab CI).
- Pipeline configuration script.
- Test coverage reporting setup.
- Common CI errors & how to fix them.
- Notification integration (Slack/Email) for failed builds.

Output in **pipeline YAML file + setup guide**."

Prompt 9 — Memory Leak Detection & Fix

"You are a performance optimisation engineer. Analyse my [language/framework] application for memory leaks and:

- Identify possible causes from code structure.
- Suggest profiling tools (Valgrind, Perf, Chrome DevTools).
- Provide a step-by-step method to reproduce memory growth.
- Fix the leak with corrected code examples.
- Suggest long-term prevention strategies.

Output in **diagnostic report + corrected code samples.**"

Prompt 10 — Multi-Threading Bug Resolution

"You are a concurrency programming specialist. My [language/framework] application faces race conditions and deadlocks.

Include:

- Detailed explanation of the issue.
- Steps to identify which thread is causing the block.
- Corrected thread-safe code.
- Recommended locking or async patterns.
- Performance benchmarks after fix.

Output in **bug analysis + updated code format.**"

Prompt 11 — Automated Unit Test Generation

"You are a test automation engineer. Write automated **unit test scripts** for my [language/framework] code covering [feature/function].

Include:

- Test cases for normal, boundary, and error conditions.
- Assertions for expected output.
- Mock data creation.
- Code coverage percentage target (e.g., >80%).
- How to run tests in CI/CD pipeline.

Output in **ready-to-run test script format.**"

Prompt 12 — SQL Query Debugging & Optimisation

"You are a database performance engineer. Optimise my slow SQL queries for [database type: MySQL, PostgreSQL, etc.].

Include:

- Query execution plan analysis.
- Indexing strategy.
- Query rewriting for speed.
- Caching recommendations.
- Before/after execution time comparison.

Output in **query optimisation report with revised SQL statements.**"

Prompt 13 — Cross-Browser Bug Fix Plan

"You are a front-end debugging expert. My web app works on Chrome but fails in Firefox & Safari. Include:

- List of browser compatibility issues.
- Code fixes using cross-browser safe APIs.
- CSS vendor prefixing guide.
- Polyfill recommendations for unsupported features.
- Testing checklist for all major browsers.

Output in **browser compatibility report + fixed code snippets.**"

Prompt 14 — API Rate Limit Error Resolution

"You are an API performance consultant. My app hits rate limits when fetching data from [API name]. Include:

- How to detect rate limit headers.
- Backoff strategies & caching techniques.
- Batch request examples.
- Code modifications for retry logic.
- Test scenarios to confirm the fix.

Output in **rate limit handling guide + updated code.**"

Prompt 15 — Deployment Bug Fix Checklist

"You are a deployment engineer. My application works locally but fails in production. Include:

- Environment variable checks.
- Dependency version mismatches.
- Server configuration issues.
- Build process verification.
- Automated rollback setup.

Output in **deployment debugging checklist format.**"

Prompt 16 — Version Control Conflict Resolution

"You are a Git (Version Control) expert. Resolve merge conflicts in my [repository name] while ensuring no functionality loss.

Include:

- Step-by-step conflict resolution process.
- Commit best practices to avoid future issues.
- Branch management guidelines.
- Git command examples for common scenarios.

Output in **merge resolution guide with Git commands.**"

Prompt 17 — Security Vulnerability Patch Plan

"You are a cybersecurity code auditor. Review my [language/framework] code for vulnerabilities like SQL injection, XSS, and CSRF.

Include:

- Vulnerability list with risk levels.
- Code patches with secure alternatives.
- OWASP (Open Web Application Security Project) best practice checklist.
- Security testing tools list.

Output in **vulnerability report + patched code samples.**"

Prompt 18 — Cloud Function Debugging Guide

"You are a cloud application developer. Debug my [AWS Lambda / Google Cloud Function / Azure Function] which is failing intermittently.

Include:

- Log analysis methods.
- Error pattern detection.
- Code changes for reliability.
- Testing with local emulators.
- Deployment steps after fix.



Output in **debugging flowchart + corrected function code.**"

Prompt 19 — Mobile App Crash Analysis

"You are a mobile app debugging expert. Analyse my [Android/iOS] app for crash reports related to [feature/function].

Include:

- Crash log interpretation.
- Root cause identification.
- Code corrections.
- Device-specific fixes.
- Testing plan for all OS versions.

Output in **crash report + updated code block.**"

Prompt 20 — Real-Time Error Monitoring Setup

"You are a site reliability engineer. Set up a **real-time error tracking system** for my [language/framework] app.

Include:

- Recommended monitoring tools (Sentry, New Relic, Datadog).
- Setup steps for integration.
- Error categorisation for alerts.
- Dashboard layout suggestions.
- Weekly reporting format.

Output in **tool setup guide + dashboard screenshot mockup.**"

Prompt 21 — Data Processing Script Debugging

"You are a data engineer. My Python ETL (Extract, Transform, Load) script fails at the transformation stage. Include:

- Step-by-step debugging for data type mismatches.
- Handling null values & schema changes.
- Logging setup for error tracking.
- Optimised data transformation logic.
- Unit tests for validation.

Output in **debugging report + corrected script.**"

Prompt 22 — Infinite Loop Prevention in Code

"You are a software safety expert. Analyse my [language] code for infinite loop risks and fix them. Include:

- Detection of loops without termination conditions.
- Corrected loop conditions.
- Safeguards to prevent reoccurrence.
- Performance benchmarks after fix.

Output in **before/after code format.**"

Prompt 23 — Automated Code Documentation Generator

"You are a documentation automation consultant. Set up an **auto-documentation system** for my [language/framework]project.

Include:

- Recommended tools (e.g., JSDoc, Sphinx).
- Integration into CI/CD pipeline.
- Style guide for consistent doc formatting.
- Sample generated documentation.

Output in **documentation setup guide + example output.**"

Prompt 24 — Debugging Scheduler & Cron Job Failures

"You are a backend engineer. My scheduled tasks in [language/framework] fail intermittently. Include:

- Log analysis to find failure points.
- Corrected cron expressions.
- Error handling logic.
- Monitoring alerts setup.
- Retry mechanism.

Output in **debugging report + fixed scheduling script.**"

Prompt 25 — Codebase Technical Debt Reduction Plan

"You are a senior software architect. Analyse my [language/framework] codebase for technical debt and create a **3-month cleanup roadmap**.

Include:

- List of outdated dependencies.
- Code smells & fixes.
- Testing coverage improvement.
- Refactoring priorities.
- Risk mitigation plan.

Output in **technical debt report + phased action plan.**"



TECH LEGAL AWARENESS FORUM

A SECTION 8 – NON-PROFIT ORGANIZATION REGISTERED UNDER THE COMPANIES ACT, 2013

SCAM FREE INDIA

CYBER ALERT