Prompt 1 — Automating Security Log Monitoring

Backstory: You're an IT administrator for a mid-sized company. Your security logs are massive, and manually scanning them for threats is impossible. Last year, you missed a brute-force attack because it got buried in the logs.

Goal: Create an automated pipeline that monitors security logs, flags suspicious activity, and sends real-time alerts.

Prompt:
"You are a **Cybersecurity Automation Engineer**. I want a script that scans server logs in real time and alerts me of suspicious activity such as failed login attempts, unusual IP addresses, or data spikes.

Your task:

1. Connect to the server log files via API or secure SSH.
2. Parse log entries and match against predefined threat patterns (failed logins >5 in 1 minute, foreign IP access, large file downloads).
3. Send an alert email/SMS if a threat is detected.
4. Store flagged events in a database for future analysis.
5. Generate a daily security summary report.

Output format: Security monitoring script + threat pattern list + alert notification system.

Input Files & Code Section:

- Path to log files or log API endpoint
- Threat detection rules CSV
- Email/SMS API credentials"

Prompt 2 — Automating Data Backup with Encryption

Backstory: You manage sensitive medical records for a clinic. If your system crashes or is hacked, you can't risk losing unencrypted patient data.

Goal: Automate daily backups to cloud storage with strong encryption.

Prompt:
"You are a **Data Security Engineer**. I want an automated backup system that encrypts files before uploading them to cloud storage (AWS S3, Google Drive).

Your task:

1. Identify sensitive folders for backup.
2. Compress and encrypt files using AES-256 encryption.
3. Upload encrypted backups to cloud storage via API.
4. Store encryption keys securely in a password vault.
5. Maintain a backup log with timestamps and checksum hashes.

Output format: Encrypted backup script + cloud upload integration + key storage instructions.

Input Files & Code Section:

- Encryption key file (secure vault reference)
- Cloud API credentials
- List of file/folder paths for backup"

Prompt 3 — Automating Phishing Email Detection

Backstory: Your employees keep falling for phishing emails, leading to security risks. Manual awareness training isn't enough.

Goal: Build an API-based system that scans incoming emails and flags potential phishing attempts.

Prompt:

"You are an **Email Security Automation Specialist**. I want to integrate Gmail API with an AI phishing detection model.

Your task:

1. Fetch incoming emails via Gmail API.
2. Scan sender domains, suspicious keywords, and link redirections.
3. Assign a risk score to each email.
4. Move high-risk emails to a "Quarantine" folder.
5. Send a weekly phishing report to the security team.

Output format: Email scanning script + risk scoring system + quarantine folder setup.

Input Files & Code Section:

- Gmail API credentials
- Keyword/risk pattern JSON file
- AI phishing detection model file"

Prompt 4 — Automating GDPR Data Deletion Requests

Backstory: Your European customers often request data deletion under GDPR laws. Handling requests manually is time-consuming and error-prone.

Goal: Automate GDPR "Right to be Forgotten" requests.

Prompt:

"You are a **Privacy Compliance Automation Expert**. I want a workflow that processes GDPR deletion requests automatically.

Your task:

1. Receive requests via a secure web form.
2. Authenticate the requester's identity via email verification.
3. Locate all user data across databases and APIs.
4. Delete or anonymise the data as per GDPR guidelines.
5. Send a confirmation email and store a compliance log.

Output format: Deletion automation script + compliance report template + GDPR checklist.

Input Files & Code Section:

- Database connection details
- Web form API endpoint
- Email verification script"

Prompt 5 — Automating Vulnerability Scans

Backstory: You’re a security analyst at a SaaS startup. You run vulnerability scans manually once a month, but threats change daily.

Goal: Schedule automated vulnerability scans for all production servers.

Prompt:
"You are a **Security Scan Automation Engineer**. I want to run daily vulnerability scans and generate reports automatically.

Your task:

1. Integrate with a vulnerability scanning tool API (e.g., Nessus, OpenVAS).
2. Schedule scans for all server IPs.
3. Export scan results to a PDF and store in a secure folder.
4. Send a daily email summary with high-risk vulnerabilities.
5. Track vulnerability history in a Google Sheet.

Output format: Scheduled scan automation + PDF reporting script + risk tracking spreadsheet.

Input Files & Code Section:

- Vulnerability scanner API credentials
- Server IP list CSV
- Google Sheets API credentials"

Prompt 6 — Automating Two-Factor Authentication (2FA) Setup for All Users

Backstory: You’re a system admin at a mid-sized company. Many employees still use only passwords to log in, making the company vulnerable to credential theft.

Goal: Enforce and automate 2FA setup across all employee accounts using APIs.

Prompt:
"You are an **Identity & Access Management Automation Specialist**. I want to roll out mandatory Two-Factor Authentication for all company accounts in Google Workspace.

Your task:

1. Use Google Admin SDK API to identify accounts without 2FA enabled.
2. Send automated emails prompting users to enable 2FA.
3. Provide a one-click link to the 2FA setup page.

4. Disable accounts not compliant after 7 days.
5. Generate a compliance report for management.

Output format: 2FA enforcement script + email template + compliance report spreadsheet.

Input Files & Code Section:

- Google Admin SDK API credentials
- Email template file

Compliance tracking CSV"

Prompt 7 — Automating Data Breach Monitoring with Dark Web Scans

Backstory: A client's credentials were leaked on the dark web, and you only found out weeks later. You want to monitor this proactively.

Goal: Create an automation that scans the dark web for stolen company credentials.

Prompt:

"You are a **Threat Intelligence Automation Engineer**. I want to integrate HaveIBeenPwned API and a dark web monitoring API to scan for leaked email/password combinations.

Your task:

1. Fetch employee email list from HR database.
2. Query APIs for data breaches linked to these emails.
3. Flag and notify affected employees to reset passwords.
4. The store results in an encrypted database.
5. Send a monthly summary to the security team.

Output format: Breach monitoring script + notification template + encrypted breach database.

Input Files & Code Section:

- API keys for HaveIBeenPwned and dark web monitoring tool
- Employee email list CSV

Email SMTP settings for alerts"

Prompt 8 — Automating Role-Based Access Control (RBAC) Updates

Backstory: Employees change departments often, but their access permissions stay the same, leaving old data vulnerable.

Goal: Automate RBAC updates based on HR records.

Prompt:

"You are an **Access Control Automation Expert**. I want an integration between our HR system and internal application APIs to update user permissions automatically.

Your task:

1. Fetch updated employee roles from HR system API.
2. Compare current permissions in application API.
3. Add/remove access rights based on role changes.
4. Log all changes with timestamps.
5. Notify IT admin for high-privilege changes.

Output format: RBAC sync script + permissions change log + alert email template.

Input Files & Code Section:

- HR system API credentials
- Application API credentials

Role-to-permission mapping JSON"

Prompt 9 — Automating Security Awareness Quizzes
Backstory: Employees forget cybersecurity best practices unless reminded regularly.

Goal: Send automated monthly security quizzes to employees via email.

Prompt:
"You are a **Security Training Automation Specialist**. I want a system that emails a short quiz to employees each month and records their scores.

Your task:

1. Store quiz questions in a Google Sheet or database.
2. Send quiz links via Gmail API.
3. Collect responses via Google Forms API.
4. Calculate scores and store in a results sheet.
5. Flag employees who score below 70% for follow-up training.

Output format: Quiz automation script + question bank file + results dashboard.

Input Files & Code Section:

- Google Sheets API credentials
- Gmail API credentials

Quiz question CSV or database file"

Prompt 10 — Automating SSL Certificate Expiry Alerts
Backstory: One of your client websites went down because the SSL certificate expired — and nobody noticed in time.

Goal: Monitor SSL expiry dates and send alerts before expiration.

Prompt:

"You are a **Web Security Automation Engineer**. I want a system that checks SSL certificate expiry dates for a list of domains.

Your task:

1. Fetch SSL certificate details for each domain.
2. Identify expiry dates within the next 30 days.
3. Send alert emails with renewal instructions.
4. Update a tracking sheet with expiry status.
5. Repeat the check daily.

Output format: SSL monitoring script + expiry alert template + tracking spreadsheet.

Input Files & Code Section:

- Domain list CSV
- Email SMTP settings
- Google Sheets API credentials"

Prompt 11 — Automating Endpoint Device Compliance Checks

Backstory: Your company has a Bring Your Own Device (BYOD) policy, but many employees connect with outdated or unpatched devices, creating security gaps.

Goal: Automatically check if employee devices meet compliance requirements before allowing network access.

Prompt:

"You are an **Endpoint Security Automation Engineer**. I want a system that verifies employee device compliance (OS version, antivirus status, firewall enabled) every time they connect to the company network.

Your task:

1. Integrate with an endpoint management API (e.g., Microsoft Intune, Jamf).
2. Collect device compliance data in real time.
3. Block network access if the device fails checks.
4. Notify the employee with steps to fix compliance issues.
5. Log all non-compliant devices for security audits.

Output format: Compliance check script + access control API integration + remediation email template.

Input Files & Code Section:

- Endpoint management API credentials
- Compliance rule configuration file (JSON)
- Network access control API credentials"

Prompt 11 — Automating Endpoint Device Compliance Checks

Backstory: Your company has a Bring Your Own Device (BYOD) policy, but many employees connect with outdated or unpatched devices, creating security gaps.

Goal: Automatically check if employee devices meet compliance requirements before allowing network access.

Prompt:
"You are an **Endpoint Security Automation Engineer**. I want a system that verifies employee device compliance (OS version, antivirus status, firewall enabled) every time they connect to the company network.

Your task:

1. Integrate with an endpoint management API (e.g., Microsoft Intune, Jamf).
2. Collect device compliance data in real time.
3. Block network access if the device fails checks.
4. Notify the employee with steps to fix compliance issues.
5. Log all non-compliant devices for security audits.

Output format: Compliance check script + access control API integration + remediation email template.

Input Files & Code Section:

- Endpoint management API credentials
- Compliance rule configuration file (JSON)
- Network access control API credentials"

Prompt 12 — Automating Sensitive File Access Alerts

Backstory: You store financial reports in a shared drive, and last quarter a contractor downloaded files they weren't supposed to access.

Goal: Set up real-time alerts for access to sensitive files.

Prompt:
"You are a **File Access Monitoring Specialist**. I want an automation that detects and alerts whenever certain high-security files are accessed.

Your task:

1. Connect to Google Drive API or internal file server API.
2. Monitor access logs for the target file/folder.
3. Trigger an alert when access is detected outside approved user list.
4. Record details: user ID, timestamp, IP address.
5. Send an incident report to the security team.

Output format: File access monitoring script + alerting system + incident report format.

Input Files & Code Section:

- File/folder ID list CSV
- Approved user list CSV

Email/SMS API credentials"

Prompt 13 — Automating Database Security Audits

Backstory: Your customer database holds personal information, but monthly manual security audits take too much time and miss critical misconfigurations.

Goal: Automate periodic database security audits and reporting.

Prompt:
"You are a **Database Security Automation Expert**. I want a script that scans for vulnerabilities in our MySQL/PostgreSQL databases and generates a security report.

Your task:

1. Connect to the database securely.
2. Check for weak passwords, outdated versions, excessive user privileges.
3. Identify unused accounts and revoke access.
4. Generate a PDF report with recommendations.
5. Email the report to the database administrator.

Output format: Database audit script + PDF report template + email delivery function.

Input Files & Code Section:

- Database connection credentials (secured)
- Vulnerability scan checklist JSON
- Email SMTP settings"

Prompt 14 — Automating USB Device Restrictions

Backstory: An employee once copied sensitive data onto a personal USB drive without permission.

Goal: Automatically detect and block unapproved USB devices.

Prompt:
"You are an **Endpoint Device Control Automation Specialist**. I want a system that blocks all USB devices except approved company drives.

Your task:

1. Detect when a USB device is connected.
2. Compare its serial number against the approved list.
3. Block access if not approved.
4. Send an alert to the IT security team.
5. Log all USB connection attempts.

Output format: USB restriction script + approved device list + alert and log system.

Input Files & Code Section:

- Approved USB device serial number list CSV
- Endpoint monitoring API credentials"

Prompt 15 — Automating Password Expiry Reminders

Backstory: Employees often forget to change passwords on time, leading to expired accounts and downtime.

Goal: Send automated password change reminders.

Prompt:
"You are an **Account Security Automation Specialist**. I want to integrate with Active Directory (AD) to send reminders before password expiry.

Your task:

1. Connect to AD via API or LDAP.
2. Fetch users whose passwords expire within 10 days.
3. Send email reminders at 10, 5, and 2 days before expiry.
4. Track who changes passwords after reminders.
5. Generate a monthly compliance report.

Output format: Reminder script + email template + compliance tracking sheet.

Input Files & Code Section:

- AD connection details
- Email SMTP credentials"

Prompt 16 — Automating Encrypted File Sharing

Backstory: Your legal team frequently shares confidential documents with clients, but sending via regular email is risky.

Goal: Create a secure encrypted file-sharing automation.

Prompt:
"You are a **Secure File Transfer Automation Expert**. I want to encrypt files and send them via a secure download link that expires after 48 hours.

Your task:

1. Accept file upload from the legal team.
2. Encrypt the file using AES-256.
3. Upload to secure cloud storage.
4. Generate a time-limited download link.
5. Email the link to the client with a decryption key sent separately.

Output format: Secure sharing script + encryption guide + link expiration setup.

Input Files & Code Section:

- Encryption key management file
- Cloud storage API credentials"

Prompt 17 — Automating Insider Threat Detection

Backstory: A recently resigned employee downloaded large volumes of data before leaving.

Goal: Detect unusual internal data access patterns.

Prompt:

"You are an **Insider Threat Monitoring Specialist**. I want a system that flags abnormal access activity by employees.

Your task:

1. Collect access logs from file servers, databases, and cloud storage.
2. Identify sudden spikes in file downloads or sensitive data access.
3. Compare against the user's historical activity patterns.
4. Flag anomalies and send alerts to security admins.
5. Log all incidents for investigation.

Output format: Threat detection script + anomaly detection rules + alert system.

Input Files & Code Section:

- Access log API endpoints
- User activity baseline data CSV
- Email/SMS API credentials"

Prompt 18 — Automating Compliance Document Management

Backstory: You must submit ISO 27001 compliance reports annually, but collecting required documents is messy.

Goal: Automate collection and organisation of compliance evidence.

Prompt:

"You are a **Compliance Automation Specialist**. I want a system that fetches logs, audit reports, and policy documents from multiple systems into one folder.

Your task:

1. Connect to APIs for security tools, HR systems, and monitoring platforms.
2. Download latest compliance-related files.
3. Store them in a structured folder by category and date.
4. Generate a manifest listing all collected documents.
5. Zip and archive the folder.

Output format: Document collection script + manifest file + folder structure template.

Input Files & Code Section:

- API credentials for each system

Compliance checklist JSON"

Prompt 19 — Automating Ransomware Simulation Drills

Backstory: You want your IT team prepared for ransomware attacks, but training is irregular.

Goal: Automate simulated ransomware drills.

Prompt:

"You are a **Cybersecurity Training Automation Engineer**. I want a script that simulates a ransomware infection without actually encrypting files.

Your task:

1. Rename and lock sample files in a sandbox environment.
2. Display a mock ransom note.
3. Test the IT team's incident response process.
4. Record time taken to respond.
5. Generate a performance report.

Output format: Simulation script + ransom note template + performance report.

Input Files & Code Section:

- Sandbox environment setup file
- Sample file set"

Prompt 20 — Automating API Security Testing

Backstory: Your company's APIs are public-facing, and you want to test them regularly for vulnerabilities.

Goal: Create an automated API penetration testing tool.

Prompt:

"You are an **API Security Testing Specialist**. I want a script that runs OWASP API Security Top 10 checks on all company APIs.

Your task:

1. Load list of API endpoints from a file.
2. Run tests for authentication flaws, excessive data exposure, and injection attacks.
3. Record findings in a structured report.
4. Notify developers of high-risk vulnerabilities.
5. Store results for trend analysis.

Output format: API security scan script + vulnerability report + notification system.

Input Files & Code Section:

- API endpoint list CSV
- API security rules JSON"

Prompt 21 — Automating Cloud Security Policy Enforcement

Backstory: Your cloud storage contains sensitive client contracts, but some employees make files public by mistake.

Goal: Automatically detect and fix misconfigured cloud permissions.

Prompt:
"You are a **Cloud Security Automation Specialist**. I want a system that scans all files in Google Drive/AWS S3 for public access and restricts them to approved users only.

Your task:

1. Connect to the cloud storage API.
2. Identify files/folders with public sharing enabled.
3. Automatically remove public access.
4. Notify the file owner about the change.
5. Log all permission changes for audits.

Output format: Permission scan script + remediation log + owner notification email template.

Input Files & Code Section:

- Cloud storage API credentials
- Approved user list CSV
- Notification email template"

Prompt 22 — Automating Incident Response Playbook Execution

Backstory: In case of a security breach, your IT team follows a manual checklist, which delays containment.

Goal: Automate the first 10 minutes of incident response.

Prompt:
"You are a **Security Incident Automation Engineer**. I want a system that executes predefined playbook actions when a breach is detected.

Your task:

1. Receive breach alert from SIEM (Security Information and Event Management) tool.
2. Isolate affected servers from the network.
3. Collect logs and system snapshots.
4. Notify the security team and management.
5. Update the incident tracking system.

Output format: Incident response script + action log + notification system.

Input Files & Code Section:

- SIEM API credentials
- Server isolation commands script
- Incident tracking tool API key"

Prompt 23 — Automating Encrypted Chat for Sensitive Communications

Backstory: Your legal team sometimes needs to chat securely with external lawyers, but regular Slack channels are not safe enough.

Goal: Provide an automated, temporary encrypted chat channel.

Prompt:
"You are a **Secure Communications Automation Specialist**. I want a tool that creates an encrypted chat room that expires after 24 hours.

Your task:

1. Generate a secure chat room via API (e.g., Matrix, Rocket.Chat).
2. Require password-protected entry.
3. Enable end-to-end encryption for all messages.
4. Automatically delete the chat room and logs after expiry.
5. Email participants the join link and password.

Output format: Chat room creation script + deletion automation + participant email template.

Input Files & Code Section:

- Secure chat API credentials

Participant email list CSV"

Prompt 24 — Automating Malware File Scanning for Uploads

Backstory: Your website allows file uploads for client documents, but there's a risk of malicious files being uploaded.

Goal: Automatically scan all uploaded files for malware.

Prompt:
"You are a **File Security Automation Engineer**. I want an integration that scans each uploaded file with a malware detection API before it's stored.

Your task:

1. Intercept file uploads via the web application backend.
2. Scan files using VirusTotal or ClamAV API.
3. Reject and quarantine suspicious files.
4. Notify the uploader about rejection.
5. Log scan results for audits.

Output format: File scanning script + quarantine folder setup + log file format.

Input Files & Code Section:

- Malware scanning API credentials

- File storage path configuration
- Notification email template"

Prompt 25 — Automating Security Patch Deployment

Backstory: A zero-day vulnerability was discovered last month, but patching all systems took your team over a week.

Goal: Deploy security patches automatically across all servers.

Prompt:

"You are a **Patch Management Automation Specialist**. I want a script that applies critical security patches across all systems as soon as they're available.

Your task:

1. Check vendor API or repositories for new patches.
2. Download and install patches automatically.
3. Reboot systems if required, during off-hours.
4. Notify admins of patch completion.
5. Maintain a patch history log.

Output format: Patch automation script + reboot schedule + patch log.

Input Files & Code Section:

- Server list CSV
- Patch repository URLs
- Admin email list"