

SECURITY ANALYSIS REPORT

Generated By

Backstage Rookie

Provider: Backstage Rookie

contact - swarajdarekar9@gmail.com

Client: Swaraj Darekar

swarajdarekar10@gmail.com

Scan ID: c4c1b9b0-15a9-43b0-854d-0c579104b731

Date: 2026-02-07 03:50:34

Version: 1.0.0

CONFIDENTIAL

TABLE OF CONTENTS

1	Document Control
1.1	Team
1.2	List of Changes
2	Executive Summary
2.1	Overview
2.2	Identified Vulnerabilities
3	Methodology
3.1	Objective
3.2	Scope
3.3	User Accounts and Permissions
4	Findings
C1	Unsafe Hugging Face Hub Download
H1	Probable Insecure Temp File Usage
M1	Possible Binding to All Interfaces
M2	Unsafe Hugging Face Hub Download
M3	Missing DNSSEC
M4	Insecure Input Usage
L1	Sensitive Data Exposure in Error
L2	Sensitive Data Exposure in Error
L3	Sensitive Data Exposure in Error
L4	Sensitive Data Exposure in Error
L5	Sensitive Data Exposure in Error
5	Endpoint Security Analysis
	/api/auth/google/login
	/api/auth/google/callback
	/api/auth/google/session
	/api/change-plan
	/api/auth/logout
	/api/get-plan
	/api/analyze
	/api/generate-report
	/api/auth/me
	/healthz
	/api/models

6	Metrics Summary
7	Disclaimer
8	Appendix
8.1	Static Appendix Section
8.2	Tool Output

EXECUTIVE SUMMARY

This security assessment report highlights several critical, high, medium, and low-severity vulnerabilities identified in the provided repository. The findings emphasize the need for immediate attention to prevent potential security breaches.

Identified Vulnerabilities

ID	Title	CVSS	Page
C1	Unsafe Hugging Face Hub Download	9.0	
H1	Probable Insecure Temp File Usage	7.5	
M1	Possible Binding to All Interfaces	5.0	
M2	Unsafe Hugging Face Hub Download	5.0	
M3	Missing DNSSEC	4.0	
M4	Insecure Input Usage	4.0	
L1	Sensitive Data Exposure in Error	3.0	
L2	Sensitive Data Exposure in Error	3.0	
L3	Sensitive Data Exposure in Error	3.0	
L4	Sensitive Data Exposure in Error	3.0	
L5	Sensitive Data Exposure in Error	3.0	

METHODOLOGY

Introduction

This report details the results of a security assessment conducted on the specified repository. The analysis involved a multi-layered approach, combining automated static analysis tools with advanced, AI-driven verification and enrichment to identify potential security vulnerabilities.

Objective

The primary objective of this assessment was to identify security weaknesses, assess their potential impact, and provide actionable recommendations for remediation to improve the overall security posture of the application.

Scope

The assessment was performed on the source code of the repository cloned at the time of the scan. The analysis focused on common web application vulnerabilities, insecure coding practices, and dependency risks.

Systems in Scope

No systems explicitly defined.

User Accounts

As this was a static source code analysis, no user accounts were provisioned or tested.

FINDINGS

C1 – Unsafe Hugging Face Hub Download

Severity: Critical

CVSS Score: 9.0

CVSS Vector: CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:C/C:H/I:H/A:H

Target: Hugging Face model download functionality

Overview

The Hugging Face model is downloaded without specifying a revision, which could lead to the download of an unexpected model version.

Details

The download is performed using the `from_pretrained` method from the `transformers` library, with the `trust_remote_code=True` argument set, which can pose security risks if the model is not from a trusted source.

Evidence

- [analysis_engine/analyzers/llm_analyzer.py:60](#): Download of Hugging Face model without revision

References

- https://huggingface.co/docs/transformers/main_classes/model

Recommendation

- Always specify a revision when downloading models from the Hugging Face Hub. - Use the `from_pretrained` method with the `revision` argument set to a specific revision ID. - Consider implementing additional security measures, such as verifying the model's digital signature.

Prompt to Solve the Vulnerability:

To fix the issue, update the `from_pretrained` call to include a specific revision ID, such as `model = AutoModelForCausalLM.from_pretrained('model_path', revision='main')`. This ensures that the exact model version intended by the developers is downloaded.

H1 – Probable Insecure Temp File Usage

Severity: High

CVSS Score: 7.5

CVSS Vector: CVSS:3.1/AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H

Target: Temporary file creation

Overview

The code uses a temporary file creation method that may be insecure.

Details

The `tempfile.mktemp` function is used to create temporary files, which can pose security risks if not properly handled.

Evidence

- [analysis_engine/analyzers/regex_analyzer.py:212](#): Usage of `tempfile.mktemp`

References

- <https://www.python.org/doc/security/>

Recommendation

- Use the `tempfile.TemporaryDirectory` context manager to create temporary directories.
- Avoid using `tempfile.mktemp` and instead use `tempfile.NamedTemporaryFile` with the `delete=True` argument.
- Ensure proper cleanup of temporary files to prevent information disclosure.

Prompt to Solve the Vulnerability:

Replace `tempfile.mktemp` with `tempfile.NamedTemporaryFile` and set `delete=True` to ensure the file is deleted after use. For example: `with tempfile.NamedTemporaryFile(delete=True) as tmp_file:`

M1 – Possible Binding to All Interfaces

Severity:	Medium
CVSS Score:	5.0
CVSS Vector:	CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:L/I:L/A:L
Target:	Application binding

Overview

The application may be binding to all available network interfaces.

Details

The `app.run` method is called with the `host='0.0.0.0'` argument, which can make the application accessible on all network interfaces.

Evidence

- **run.py:10:** Binding to all interfaces

References

- <https://docs.python.org/3/library/http.server.html>

Recommendation

- Bind the application to a specific IP address or hostname. - Use a reverse proxy or load balancer to control access to the application. - Consider implementing firewall rules to restrict access to the application.

Prompt to Solve the Vulnerability:

To fix the issue, update the `app.run` call to bind to a specific IP address, such as `app.run(host='127.0.0.1')`. This ensures that the application is only accessible on the localhost interface.

M2 – Unsafe Hugging Face Hub Download

Severity:	Medium
CVSS Score:	5.0
CVSS Vector:	CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:C/C:H/I:H/A:H
Target:	Hugging Face model download functionality

Overview

The Hugging Face model is downloaded without specifying a revision, which could lead to the download of an unexpected model version.

Details

The download is performed using the `from_pretrained` method from the `transformers` library, with the `trust_remote_code=True` argument set, which can pose security risks if the model is not from a trusted source.

Evidence

- [analysis_engine/utils/model_manager.py:50](#): Download of Hugging Face model without revision

References

- https://huggingface.co/docs/transformers/main_classes/model

Recommendation

- Always specify a revision when downloading models from the Hugging Face Hub. - Use the `from_pretrained` method with the `revision` argument set to a specific revision ID. - Consider implementing additional security measures, such as verifying the model's digital signature.

Prompt to Solve the Vulnerability:

To fix the issue, update the `from_pretrained` call to include a specific revision ID, such as `model = AutoModelForCausalLM.from_pretrained('model_path', revision='main')`. This ensures that the exact model version intended by the developers is downloaded.

M3 – Missing DNSSEC

Severity: Medium

CVSS Score: 4.0

CVSS Vector: CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:C/C:L/I:L/A:L

Target: DNS resolution

Overview

The application may be vulnerable to DNS spoofing attacks due to the lack of DNSSEC.

Details

The application does not use DNSSEC to validate the authenticity of DNS responses, which can make it vulnerable to DNS spoofing attacks.

Evidence

- `analysis_engine/analyzers/regex_analyzer.py:237`: Missing DNSSEC

References

- <https://dnssec.net/>

Recommendation

- Implement DNSSEC to validate the authenticity of DNS responses. - Use a DNS resolver that supports DNSSEC. - Consider implementing additional security measures, such as using a DNS proxy or VPN.

Prompt to Solve the Vulnerability:

To fix the issue, configure the application to use a DNS resolver that supports DNSSEC, such as `dns.resolver` from the `dnspython` library. For example: `import dns.resolver; resolver = dns.resolver.Resolver()`

M4 – Insecure Input Usage

Severity:	Medium
CVSS Score:	4.0
CVSS Vector:	CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:L/I:L/A:L
Target:	User input handling

Overview

The application may be vulnerable to input validation attacks due to insecure input usage.

Details

The application does not properly validate user input, which can make it vulnerable to input validation attacks.

Evidence

- [analysis_engine/analyzers/regex_analyzer.py:240](#): Insecure input usage

References

- <https://www.python.org/doc/security/>

Recommendation

- Implement input validation to ensure that user input conforms to expected formats. - Use a library or framework that provides input validation features. - Consider implementing additional security measures, such as using a web application firewall (WAF).

Prompt to Solve the Vulnerability:

To fix the issue, update the code to use a library or framework that provides input validation features, such as `wtforms` from the `Flask-WTF` library. For example: `from flask_wtf import FlaskForm; class MyForm(FlaskForm):`

L1 – Sensitive Data Exposure in Error

Severity:	Low
CVSS Score:	3.0
CVSS Vector:	CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:L/I:L/A:L
Target:	Error handling

Overview

The application may be vulnerable to sensitive data exposure in error messages.

Details

The application does not properly handle error messages, which can expose sensitive data to attackers.

Evidence

- `app/routes/main.py:75`: Sensitive data exposure in error

References

- <https://www.python.org/doc/security/>

Recommendation

- Implement proper error handling to prevent sensitive data exposure. - Use a library or framework that provides error handling features. - Consider implementing additional security measures, such as using a logging framework.

Prompt to Solve the Vulnerability:

To fix the issue, update the code to use a library or framework that provides error handling features, such as `logging` from the `python-logging` library. For example: `import logging; logging.basicConfig()`

L2 – Sensitive Data Exposure in Error

Severity:	Low
CVSS Score:	3.0
CVSS Vector:	CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:L/I:L/A:L
Target:	Error handling

Overview

The application may be vulnerable to sensitive data exposure in error messages.

Details

The application does not properly handle error messages, which can expose sensitive data to attackers.

Evidence

- `app/routes/main.py:166`: Sensitive data exposure in error

References

- <https://www.python.org/doc/security/>

Recommendation

- Implement proper error handling to prevent sensitive data exposure. - Use a library or framework that provides error handling features. - Consider implementing additional security measures, such as using a logging framework.

Prompt to Solve the Vulnerability:

To fix the issue, update the code to use a library or framework that provides error handling features, such as `logging` from the `python-logging` library. For example: `import logging; logging.basicConfig()`

L3 – Sensitive Data Exposure in Error

Severity:	Low
CVSS Score:	3.0
CVSS Vector:	CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:L/I:L/A:L
Target:	Error handling

Overview

The application may be vulnerable to sensitive data exposure in error messages.

Details

The application does not properly handle error messages, which can expose sensitive data to attackers.

Evidence

- `app/routes/main.py:195`: Sensitive data exposure in error

References

- <https://www.python.org/doc/security/>

Recommendation

- Implement proper error handling to prevent sensitive data exposure. - Use a library or framework that provides error handling features. - Consider implementing additional security measures, such as using a logging framework.

Prompt to Solve the Vulnerability:

To fix the issue, update the code to use a library or framework that provides error handling features, such as `logging` from the `python-logging` library. For example: `import logging; logging.basicConfig()`

L4 – Sensitive Data Exposure in Error

Severity:	Low
CVSS Score:	3.0
CVSS Vector:	CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:L/I:L/A:L
Target:	Error handling

Overview

The application may be vulnerable to sensitive data exposure in error messages.

Details

The application does not properly handle error messages, which can expose sensitive data to attackers.

Evidence

- [app/services/flaskFastApi_info_service.py:347](#): Sensitive data exposure in error

References

- <https://www.python.org/doc/security/>

Recommendation

- Implement proper error handling to prevent sensitive data exposure. - Use a library or framework that provides error handling features. - Consider implementing additional security measures, such as using a logging framework.

Prompt to Solve the Vulnerability:

To fix the issue, update the code to use a library or framework that provides error handling features, such as `logging` from the `python-logging` library. For example: `import logging; logging.basicConfig()`

L5 – Sensitive Data Exposure in Error

Severity:	Low
CVSS Score:	3.0
CVSS Vector:	CVSS:3.1/AV:N/AC:L/PR:L/UI:N/S:U/C:L/I:L/A:L
Target:	Error handling

Overview

The application may be vulnerable to sensitive data exposure in error messages.

Details

The application does not properly handle error messages, which can expose sensitive data to attackers.

Evidence

- `app/routes/main.py:259`: Sensitive data exposure in error

References

- <https://www.python.org/doc/security/>

Recommendation

- Implement proper error handling to prevent sensitive data exposure. - Use a library or framework that provides error handling features. - Consider implementing additional security measures, such as using a logging framework.

Prompt to Solve the Vulnerability:

To fix the issue, update the code to use a library or framework that provides error handling features, such as `logging` from the `python-logging` library. For example: `import logging; logging.basicConfig()`

ENDPOINT SECURITY ANALYSIS

Endpoint Path /api/auth/google/login

Endpoint Path	/api/auth/google/login
HTTP Methods	GET
Source Location	..\routes\GoogleIntegra.py - 23
Authentication Required	No
Risk Severity	Low
CVSS Score	0.0

Authentication Analysis

This endpoint does not enforce authentication, which may expose it to unauthorized access depending on its functionality and the sensitivity of data processed.

Request Analysis

Content Type: unknown

No request fields were identified for this endpoint.

Response Analysis

Content Type: unknown

Status Codes:

Contains Sensitive Data: No

Identified Security Risks

No direct security risks were identified for this endpoint.

Compliance Impact

Regulation	Applicable	Risk Level	Reason
SOC 2	No	low	This endpoint appears to be for initiating a Google login flow, which typically redirects to Google and doesn't handle sensitive data directly within this endpoint.
ISO/IEC 27001	No	low	This endpoint initiates an external authentication flow and does not appear to directly handle sensitive information or system configurations.
CSA STAR	No	low	This endpoint is part of an authentication mechanism and does not directly involve data processing or storage in a way that is typically addressed by CSA STAR.

Security Assessment Notes

CVSS Vector: string

References

Endpoint Path /api/auth/google/callback

Endpoint Path	/api/auth/google/callback
HTTP Methods	GET
Source Location	..\routes\GoogleIntegra.py - 46
Authentication Required	No
Risk Severity	Low
CVSS Score	0.0

Authentication Analysis

This endpoint does not enforce authentication, which may expose it to unauthorized access depending on its functionality and the sensitivity of data processed.

Request Analysis

Content Type: unknown

No request fields were identified for this endpoint.

Response Analysis

Content Type: unknown

Status Codes:

Contains Sensitive Data: No

Identified Security Risks

No direct security risks were identified for this endpoint.

Compliance Impact

Regulation	Applicable	Risk Level	Reason
SOC 2	No	low	This endpoint is a callback for Google authentication. While it processes authentication tokens, the sensitive data handling aspects are not clearly visible from this snippet.
ISO/IEC 27001	No	low	This endpoint handles authentication callbacks. Further analysis of data processing within the callback logic would be needed for a definitive ISO 27001 assessment.
CSA STAR	No	low	This endpoint is part of an OAuth flow. Its direct impact on cloud security controls requires understanding the data exchanged and processed.

Security Assessment Notes

CVSS Vector: string

References

Endpoint Path /api/auth/google/session

Endpoint Path	/api/auth/google/session
HTTP Methods	GET
Source Location	..\routes\GoogleIntegra.py - 82
Authentication Required	No
Risk Severity	Low
CVSS Score	0.0

Authentication Analysis

This endpoint does not enforce authentication, which may expose it to unauthorized access depending on its functionality and the sensitivity of data processed.

Request Analysis

Content Type: unknown

No request fields were identified for this endpoint.

Response Analysis

Content Type: unknown

Status Codes:

Contains Sensitive Data: No

Identified Security Risks

No direct security risks were identified for this endpoint.

Compliance Impact

Regulation	Applicable	Risk Level	Reason
SOC 2	No	low	This endpoint retrieves a Google session. Without visibility into what session information is returned, a definitive risk assessment is difficult.
ISO/IEC 27001	No	low	This endpoint retrieves session information. The compliance applicability depends on the sensitivity of the session data and how it's managed.
CSA STAR	No	low	This endpoint relates to session management. Its compliance implications depend on the nature of the session data and its protection.

Security Assessment Notes

CVSS Vector: string

References

Endpoint Path /api/change-plan

Endpoint Path	/api/change-plan
HTTP Methods	POST
Source Location	..\routes\main.py - 55
Authentication Required	No
Risk Severity	High
CVSS Score	8.8

Authentication Analysis

This endpoint does not enforce authentication, which may expose it to unauthorized access depending on its functionality and the sensitivity of data processed.

Request Analysis

Content Type: unknown

No request fields were identified for this endpoint.

Response Analysis

Content Type: unknown

Status Codes:

Contains Sensitive Data: No

Identified Security Risks

- **AUTH_MISSING** (high): Authentication is required for this endpoint as it modifies user data (changing a plan).

Potential Attack Scenario:

An attacker could change plans for any user without authorization.

Compliance Impact

Regulation	Applicable	Risk Level	Reason
SOC 2	Yes	high	Missing authentication on an endpoint that changes user plans violates controls related to access control and data integrity.
ISO/IEC 27001	Yes	high	Lack of authentication on a modification endpoint poses a significant risk to access control (A.9) and data integrity (A.8).
CSA STAR	Yes	high	This endpoint's lack of authentication poses a direct risk to the security of cloud-based data and services, impacting integrity and access controls.

Security Assessment Notes

CVSS Vector: CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H

References

- <https://nvd.nist.gov/vuln-metrics/cvss/v3-calculator>
- https://owasp.org/www-community/vulnerabilities/Unprotected_TotheWeb_API

Endpoint Path /api/auth/logout

Endpoint Path	/api/auth/logout
HTTP Methods	POST
Source Location	..\routes\main.py - 79
Authentication Required	No
Risk Severity	Medium
CVSS Score	5.3

Authentication Analysis

This endpoint does not enforce authentication, which may expose it to unauthorized access depending on its functionality and the sensitivity of data processed.

Request Analysis

Content Type: unknown

No request fields were identified for this endpoint.

Response Analysis

Content Type: unknown

Status Codes:

Contains Sensitive Data: No

Identified Security Risks

- AUTH_MISSING** (medium): Logout functionality typically requires authentication to ensure the correct user session is terminated.

Potential Attack Scenario:

An attacker could potentially terminate other users' sessions if session management is weak and logout is not properly authenticated.

Compliance Impact

Regulation	Applicable	Risk Level	Reason
SOC 2	Yes	medium	While not directly exposing data, unprotected logout can imply weak session management controls, impacting SOC 2 requirements.
ISO/IEC 27001	Yes	medium	Weak session termination controls can be a risk to access control (A.9) and potentially lead to unauthorized access if sessions are not properly invalidated.
CSA STAR	Yes	medium	Improper session management, even for logout, can impact the overall security posture of cloud services.

Security Assessment Notes

CVSS Vector: CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:N/I:L/A:N

References

- <https://nvd.nist.gov/vuln-metrics/cvss/v3-calculator>
- <https://owasp.org/www-project-api-security/>

Endpoint Path /api/get-plan

Endpoint Path	/api/get-plan
HTTP Methods	GET
Source Location	..\routes\main.py - 87
Authentication Required	Yes (recommended)
Risk Severity	Low
CVSS Score	0.0

Authentication Analysis

This endpoint enforces authentication using a recommended-based mechanism. Authentication checks were detected at the following code locations: 86, 87, 86.

Request Analysis

Content Type: unknown

No request fields were identified for this endpoint.

Response Analysis

Content Type: unknown

Status Codes:

Contains Sensitive Data: No

Identified Security Risks

No direct security risks were identified for this endpoint.

Compliance Impact

Regulation	Applicable	Risk Level	Reason
SOC 2	Yes	low	This endpoint appears to retrieve user plan information. Authentication is indicated, which aligns with access control requirements. The sensitivity of plan data needs further evaluation.
ISO/IEC 27001	Yes	low	Retrieving user plan information, if authenticated, generally aligns with access control policies (A.9). Data classification is key for a full assessment.
CSA STAR	Yes	low	Retrieving user plan data, assuming proper authentication, is usually compliant with cloud security best practices. Sensitivity of the data is a factor.

Security Assessment Notes

CVSS Vector: string

References

Endpoint Path /api/analyze

Endpoint Path	/api/analyze
HTTP Methods	POST
Source Location	..\routes\main.py - 98
Authentication Required	Yes (recommended)
Risk Severity	Low
CVSS Score	0.0

Authentication Analysis

This endpoint enforces authentication using a recommended-based mechanism. Authentication checks were detected at the following code locations: 97, 98, 97.

Request Analysis

Content Type: unknown

No request fields were identified for this endpoint.

Response Analysis

Content Type: unknown

Status Codes:

Contains Sensitive Data: No

Identified Security Risks

No direct security risks were identified for this endpoint.

Compliance Impact

Regulation	Applicable	Risk Level	Reason
SOC 2	Yes	low	This endpoint analyzes repositories. Authentication is indicated. The compliance depends on whether sensitive code or data is processed and stored.
ISO/IEC 27001	Yes	low	Repository analysis implies handling of code, which can be sensitive. Authentication is present, aligning with access controls (A.9). Data protection (A.8) is crucial.
CSA STAR	Yes	low	Analyzing repositories falls under data processing. Authentication is important. The security of data processed and stored in the cloud must be considered.

Security Assessment Notes

CVSS Vector: string

References

Endpoint Path /api/generate-report

Endpoint Path	/api/generate-report
HTTP Methods	POST
Source Location	..\routes\main.py - 176
Authentication Required	Yes (recommended)
Risk Severity	Low
CVSS Score	0.0

Authentication Analysis

This endpoint enforces authentication using a recommended-based mechanism. Authentication checks were detected at the following code locations: 175, 176, 175.

Request Analysis

Content Type: unknown

No request fields were identified for this endpoint.

Response Analysis

Content Type: unknown

Status Codes:

Contains Sensitive Data: No

Identified Security Risks

No direct security risks were identified for this endpoint.

Compliance Impact

Regulation	Applicable	Risk Level	Reason
SOC 2	Yes	low	Generating reports often involves sensitive data. Authentication is present, which is good for access control (CC7.1). The sensitivity of report content is key.
ISO/IEC 27001	Yes	low	Report generation requires authentication (A.9). If reports contain sensitive information, data protection (A.8) and access controls are critical.
CSA STAR	Yes	low	Generating reports can involve sensitive data. Proper authentication and access control are essential for cloud security.

Security Assessment Notes

CVSS Vector: string

References

Endpoint Path /api/auth/me

Endpoint Path	/api/auth/me
HTTP Methods	GET
Source Location	..\routes\main.py - 200
Authentication Required	Yes (recommended)
Risk Severity	Low
CVSS Score	0.0

Authentication Analysis

This endpoint enforces authentication using a recommended-based mechanism. Authentication checks were detected at the following code locations: 199, 200, 200, 207, 199, 208, 208, 208.

Request Analysis

Content Type: unknown

No request fields were identified for this endpoint.

Response Analysis

Content Type: unknown

Status Codes:

Contains Sensitive Data: No

Identified Security Risks

No direct security risks were identified for this endpoint.

Compliance Impact

Regulation	Applicable	Risk Level	Reason
SOC 2	Yes	low	Retrieving user profile information requires authentication, which is present, aligning with access controls (CC7.1). Sensitivity of profile data is a consideration.
ISO/IEC 27001	Yes	low	Accessing user profile data necessitates strong authentication (A.9). If PII is involved, data protection (A.8) is paramount.
CSA STAR	Yes	low	Retrieving user profile data requires secure authentication. Protecting PII in cloud environments is a key aspect.

Security Assessment Notes

CVSS Vector: string

References

Endpoint Path /healthz

Endpoint Path	/healthz
HTTP Methods	GET
Source Location	..\routes\main.py - 264
Authentication Required	No
Risk Severity	Low
CVSS Score	0.0

Authentication Analysis

This endpoint does not enforce authentication, which may expose it to unauthorized access depending on its functionality and the sensitivity of data processed.

Request Analysis

Content Type: unknown

No request fields were identified for this endpoint.

Response Analysis

Content Type: unknown

Status Codes:

Contains Sensitive Data: No

Identified Security Risks

No direct security risks were identified for this endpoint.

Compliance Impact

Regulation	Applicable	Risk Level	Reason
SOC 2	No	low	Health check endpoints are typically unauthenticated and do not expose sensitive data, therefore less directly relevant to SOC 2 controls.
ISO/IEC 27001	No	low	Health check endpoints are operational and do not typically involve sensitive information or system configurations relevant to ISO 27001.
CSA STAR	No	low	Health check endpoints are for monitoring service availability and do not usually pose a direct risk to cloud data security.

Security Assessment Notes

CVSS Vector: string

References

Endpoint Path /api/models

Endpoint Path	/api/models
----------------------	-------------

HTTP Methods	GET
Source Location	..\routes\main.py - 271
Authentication Required	Yes (recommended)
Risk Severity	Low
CVSS Score	0.0

Authentication Analysis

This endpoint enforces authentication using a recommended-based mechanism. Authentication checks were detected at the following code locations: 270, 271, 270.

Request Analysis

Content Type: unknown

No request fields were identified for this endpoint.

Response Analysis

Content Type: unknown

Status Codes:

Contains Sensitive Data: No

Identified Security Risks

No direct security risks were identified for this endpoint.

Compliance Impact

Regulation	Applicable	Risk Level	Reason
SOC 2	Yes	low	Listing models, if they represent sensitive configurations or data, should be authenticated, which is indicated. Access controls are relevant (CC7.1).
ISO/IEC 27001	Yes	low	Accessing model information, especially if they relate to data processing or sensitive system components, should be authenticated (A.9).
CSA STAR	Yes	low	Access to models used in cloud services should be properly controlled through authentication to ensure cloud service integrity.

Security Assessment Notes

CVSS Vector: string

References

METRICS SUMMARY

Total Findings: 11

Severity	Count
CRITICAL	1
HIGH	1
MEDIUM	4
LOW	5

BUSINESS RISK ADVICE

Based on a comprehensive security analysis, here are the prioritized recommendations to enhance the security posture and mitigate identified risks.

Remediate Critical RCE, TOCTOU, and Unauthorized Access Vulnerabilities (Priority: Immediate)

Description: Address critical supply chain risks from unsafe Hugging Face model downloads, prevent race conditions from insecure temporary file usage, and secure the `/api/change-plan` endpoint against unauthorized modifications.

Why it Matters: These vulnerabilities present immediate and severe risks, including remote code execution, sensitive data compromise, and unauthorized alteration of critical user data, directly impacting the confidentiality, integrity, and availability of the application. This is a critical violation of SOC 2, ISO/IEC 27001, and CSA STAR access control and data integrity principles.

Recommended Actions:

- For `analysis_engine/analyzers/llm_analyzer.py:60` and `analysis_engine/utils/model_manager.py:50`: Enforce specific revision IDs (e.g., commit hash or 'main') for all `from_pretrained` calls and consider model signature verification.
- For `analysis_engine/analyzers/regex_analyzer.py:212`: Replace `tempfile.mktemp` with `tempfile.NamedTemporaryFile(delete=True)` or `tempfile.TemporaryDirectory` for secure temporary resource management.
- For `/api/change-plan` (app/routes/main.py:55): Implement mandatory authentication and granular authorization checks to ensure only authenticated and authorized users can access and modify plan information.

Expected Outcome: Elimination of critical remote code execution and temporary file exploitation risks, prevention of unauthorized plan changes, and immediate improvement in data integrity and access control posture.

Harden API Endpoints and Enhance Input Validation (Priority: Short-Term)

Description: Restrict the application's network binding to specific interfaces, secure the logout functionality, and implement comprehensive input validation across the application.

Why it Matters: Binding to all interfaces increases the attack surface. An unauthenticated logout endpoint can be abused for session termination attacks. Insecure input usage is a root cause for many common web vulnerabilities (e.g., injection attacks) that can lead to data breaches or system compromise, impacting all core security principles and compliance requirements.

Recommended Actions:

- For `run.py:10`: Change `app.run(host='0.0.0.0')` to `app.run(host='127.0.0.1')` or a specific internal IP, and enforce external access via a properly configured reverse proxy.
- For `/api/auth/logout` (app/routes/main.py:79): Require authentication for this endpoint to ensure that only the legitimate user can invalidate their own session.
- For `analysis_engine/analyzers/regex_analyzer.py:240`: Enforce strict input validation for all user-supplied data using secure libraries (e.g., Flask-WTF for Flask) to sanitize and validate against expected formats and types.

Expected Outcome: Reduced network exposure, prevention of unauthorized session invalidation, and mitigation of common input-based attacks such as SQL injection, XSS, and command injection.

Standardize Secure Error Handling and Logging (Priority: Short-Term)

Description: Implement a centralized and secure mechanism for handling application errors to prevent sensitive data leakage through verbose error messages.

Why it Matters: Exposing detailed error information (stack traces, internal paths) provides attackers with valuable reconnaissance, making it easier to identify and exploit other vulnerabilities. This is a direct violation of data confidentiality requirements under SOC 2, ISO/IEC 27001, and CSA STAR.

Recommended Actions:

- For `app/routes/main.py` (lines 75, 166, 195, 259) and `app/services/flaskFastApi_info_service.py:347`: Implement a global exception handler that catches all unhandled exceptions.
- Configure the application to present generic error messages to end-users (e.g., 'An internal server error occurred') for all production environments.
- Integrate a robust logging framework (e.g., Python's `logging`) to capture detailed error information, stack traces, and relevant context securely to internal logs, ensuring no sensitive data is written to publicly accessible logs.

Expected Outcome: Prevention of sensitive information exposure through error messages, improved user experience during application failures, and enhanced internal observability for debugging and incident response.

Implement DNS Security Extensions (DNSSEC) (Priority: Mid-Term)

Description: Protect the application against DNS spoofing and cache poisoning by ensuring DNS resolution processes validate DNSSEC records.

Why it Matters: DNS-based attacks can redirect legitimate user traffic to malicious sites, leading to phishing, data interception, and compromised application integrity. DNSSEC is crucial for ensuring the authenticity and integrity of DNS queries and responses, aligning with ISO/IEC 27001 (A.13.1.2) and CSA STAR identity and access management controls.

Recommended Actions:

- For `analysis_engine/analyzers/regex_analyzer.py:237`: Configure the underlying infrastructure and application's DNS clients to use DNS resolvers that actively support and validate DNSSEC.
- Ensure all critical domain names associated with the application are DNSSEC-signed and properly maintained.
- Where direct DNS lookups are performed within the application, integrate libraries that support DNSSEC validation (e.g., `dnspython`).

Expected Outcome: Protection against DNS spoofing, ensuring users and the application connect to legitimate services and resources, thereby upholding communication integrity.

Establish a Comprehensive Secure Development Lifecycle (SDLC) (Priority: Long-Term)

Description: Integrate security practices and tools throughout the entire software development lifecycle to proactively identify and mitigate vulnerabilities, moving beyond reactive fixes.

Why it Matters: A mature SDLC significantly reduces the introduction of new vulnerabilities, improves the overall security posture, and reduces the cost of remediation. This strategic approach is fundamental for continuous compliance with SOC 2, ISO/IEC 27001, and CSA STAR frameworks.

Recommended Actions:

- Mandate regular, role-specific security training for all development, QA, and operations teams covering secure coding, threat modeling, and privacy-by-design principles.
- Implement and automate Static Application Security Testing (SAST) and Software Composition Analysis (SCA) into the CI/CD pipeline to detect code-level and dependency vulnerabilities early.
- Establish a formal threat modeling process for all new features and major architectural changes to identify and mitigate design-level security flaws before implementation.
- Conduct regular manual security code reviews and penetration testing by independent security teams or third-party experts.
- Develop and enforce secure coding guidelines and standards specific to Python, Flask/FastAPI, and any other technologies used, integrating them into code review checklists.

Expected Outcome: A security-aware development culture, significant reduction in recurring vulnerabilities, faster and more efficient security issue resolution, and demonstrable adherence to industry security best practices and compliance standards.

DISCLAIMER

This report is generated by an automated security analysis tool.