**Assignment -AST Consulting**

**Name:-Prakhar Jain**

**Email:-prakharjain653@gmail.com**

**Enroll:-20103303**

Web app:-Netflix

Executive Summary:

• Brief overview of the assessment.

• High-level findings and recommendations.

• Target audience: non-technical stakeholders.

Introduction:

• Background information on the web application.

• Purpose and scope of the security assessment.

• Methods and tools used in the assessment.

Scope of Assessment:

• Clearly define the boundaries of the assessment.

• Specify the components, modules, or functionalities assessed.

Methodology:

• Explain the approach taken for the assessment.

• Detail tools and techniques used.

• Describe any limitations or constraints.

Authentication and Authorization:

• Findings related to how users are authenticated.

• Assess the strength of authorization mechanisms.

• Identify any vulnerabilities and potential exploits.

Input Validation:

• Evaluate how the application handles user inputs.

• Identify any lack of input validation.

• Provide examples of potential input-based attacks.

Session Management:

• Assess the handling of user sessions.

• Identify any session-related vulnerabilities.

• Recommend improvements for secure session management.

Cross-Site Scripting (XSS):

• Detail findings related to XSS vulnerabilities.

• Include examples of vulnerable code or areas.

• Suggest remediation measures.

SQL Injection:

• Report any SQL injection vulnerabilities.

• Provide examples of potential SQL injection points.

• Recommend strategies for preventing SQL injection.

Cross-Site Request Forgery (CSRF):

• Evaluate the susceptibility to CSRF attacks.

• Provide examples of potential CSRF vulnerabilities.

• Suggest countermeasures to mitigate CSRF risks.

Denial of Service (DoS) Attacks:

• Assess the resilience against DoS attacks.

• Identify any potential weaknesses.

• Recommend measures to mitigate DoS risks.

Findings Summary:

• Summarize key findings across all assessed areas.

• Prioritize findings based on severity and potential impact.

Recommendations:

• Propose specific actions to address identified vulnerabilities.

• Prioritize recommendations based on risk levels.

• Provide guidance on implementing suggested improvements.

Conclusion:

• Recap major findings and recommendations.

• Conclude with the overall security posture of the web application.

Appendix:

• Include any additional technical details, logs, or supporting documentation.

Ensure your report is clear, concise, and accessible to both technical and non-technical stakeholders. Use a professional tone and provide actionable insights for improving the web application's security.

SQL Ingestion

import requests

def check\_authentication(url):

pass

def check\_authorization(url):

pass

def check\_input\_validation(url):

pass

def check\_session\_management(url):

pass

def check\_xss(url):

pass

def check\_sql\_injection(url):

pass

def check\_csrf(url):

pass

def check\_dos(url):

pass

def main():

target\_url = "https://example.com"

check\_authentication(target\_url)

check\_authorization(target\_url)

check\_input\_validation(target\_url)

check\_session\_management(target\_url)

check\_xss(target\_url)

check\_sql\_injection(target\_url)

check\_csrf(target\_url)

check\_dos(target\_url)

if \_\_name\_\_ == "\_\_main\_\_":

from flask import Flask, render\_template, request

app = Flask(\_\_name\_\_)

@app.route('/')

def index():

return render\_template('index.html')

@app.route('/submit', methods=['POST'])

def submit():

user\_input = request.form.get('user\_input')

if not user\_input or not isinstance(user\_input, str):

return render\_template('error.html', message='Invalid input')

return render\_template('success.html', message='Data submitted successfully')

if \_\_name\_\_ == '\_\_main\_\_':

app.run(debug=True)