THREAT MODEL AND MY OVERALL UNDERSTANDING OF WEB-SECURITY AND KEY TAKEAWAYS

THREAT MODEL

SCOPE

Threat model information

Application name: A Product Management System

Description: This platform allows customers to register and log in to easily manage their accounts. Once logged in, customers can create and manage their own products. Each product created is uniquely associated with the customer who owns it, enabling personalized management and ownership. The system ensures a seamless and secure experience for customers as they manage their products in one central location

Document Owner: Amo Samuel

Reviewer: Thomas Darko

External Dependencies

- Server type will be linux
- MySQL Database

Entry point

- Https ports
- entry point for both users and admins to log in to the system
- entry point for admin to get a list of all customers
- entry point for customers to create, delete, update and read their products they create

Exit Points

- User data returned in JSON format when fetching users.
- Product data returned in JSON format when retrieving one or more products from the database

Assets

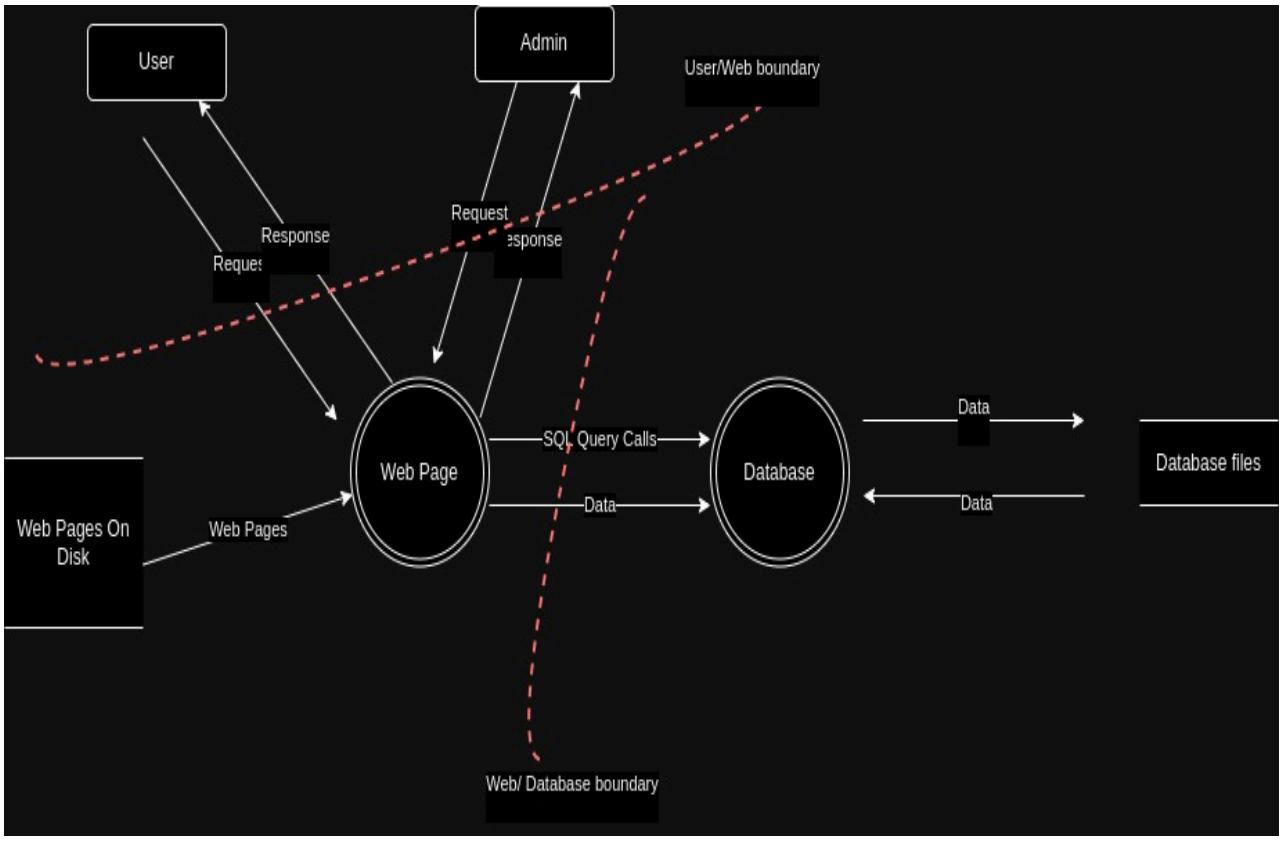
- User Data
- Products Data
- Database Server
- Availability of the web server
- Availability of the DB server

Trust Levels

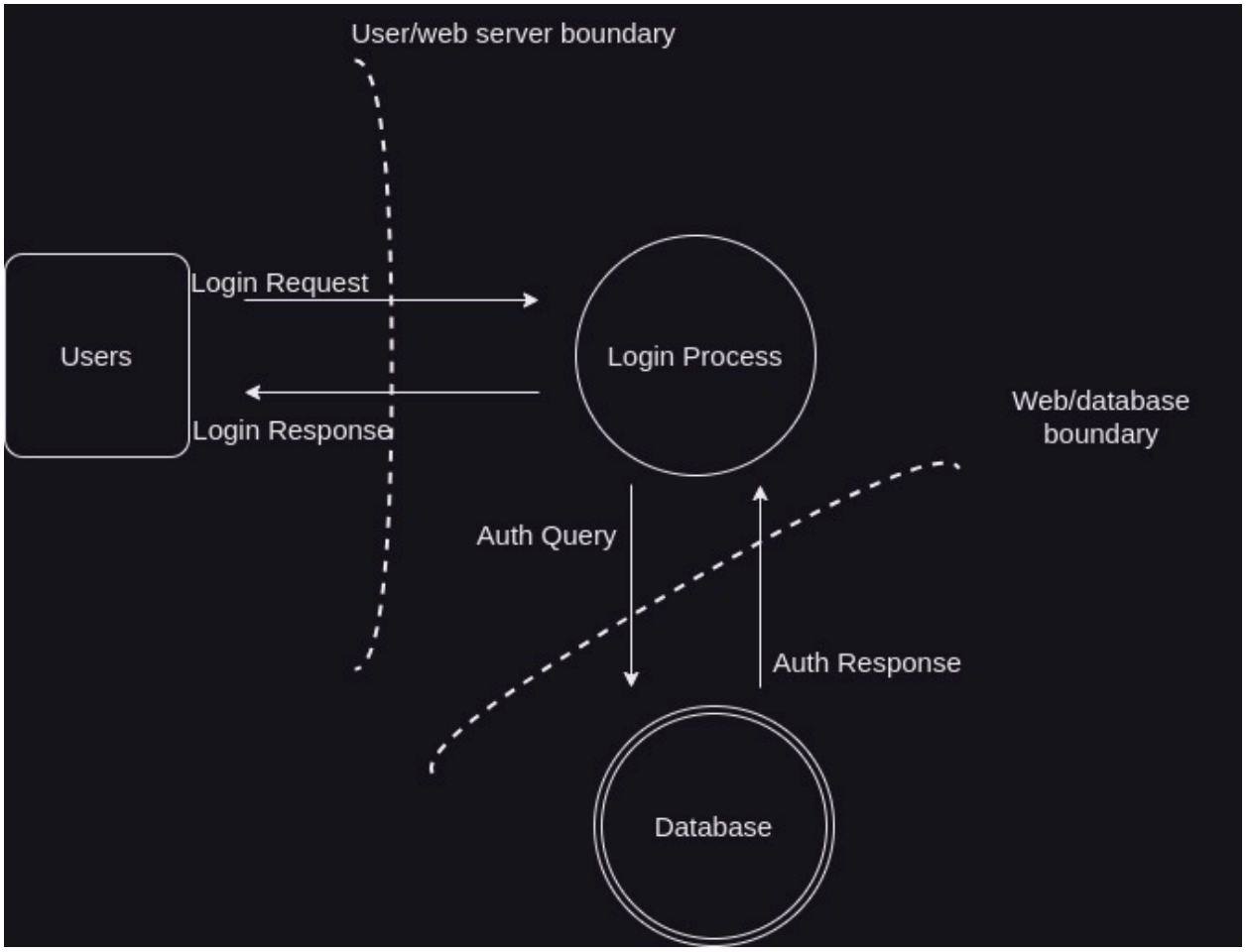
- Admin - Full access to manage users and view all products created by each user.
- Authenticated User - Access to create and manage their own products.
- Anonymous - A user who has connected to the application but has not provided valid credentials
- Database Read User - The database user account used to access the database for read access
- Database Read/Write User - The database user account used to access the database for read and write access

Data Flow Diagrams

Data flow diagram for the product application



Data flow diagram for login process



THREATS

Users

Spoofing: An attacker could pretend to be a legitimate customer by stealing or guessing login credentials

Repudiation:

- The customer may deny performing certain actions, if there's no logging or audit mechanism.
- A user with lower privileges may attempt to exploit vulnerabilities to escalate their privileges to an admin level

Web Page

Tampering: The website may inadvertently leak sensitive customer data, either through improper authentication or insecure API endpoints.

Information Disclosure: Attackers might overwhelm the website with requests, causing it to become slow or unresponsive

Denial of Service: Without logging mechanisms, malicious actions might go untracked, and attackers or users could deny performing specific activities

Database

Tamperinge: Attackers may attempt to modify or delete data in the database through SQL injection or other techniques

Information Disclosure: Improper database configurations could allow unauthorized access to sensitive customer information.

Denial of service: A flood of queries or poorly optimized operations could slow down or crash the database.

MITIGATION TECHNIQUES

Users

Spoofing: Implement strong authentication mechanisms, such as multi-factor authentication (MFA) and complex passwords. Ensure secure storage of passwords using hashing and salting techniques

Repudiation: Implement role-based access control (RBAC) to ensure that users have only the privileges they need. Regularly audit and review permissions for potential misuse

Web Page

Tampering: Use Content Security Policy (CSP) to prevent the injection of malicious scripts. Validate and sanitize all user inputs, and implement integrity checks for web page content.

Information Disclosure: Ensure all sensitive data is encrypted both in transit (using TLS/SSL) and at rest. Limit the exposure of sensitive data through secure API design and access controls

Denial of Service: Implement rate limiting, IP blacklisting, and CAPTCHA challenges to mitigate DoS attacks.

Database

Tampering: Implement strong access controls and use encryption to protect data integrity. Regularly backup the database and monitor for unauthorized access or changes.

Information Disclosure: Encrypt sensitive data stored in the database and ensure that only authorized users have access. Use parameterized queries to prevent SQL injection attacks.

Denial of service: Implement database-specific rate limiting and use connection pooling to manage load. Monitor database performance and set up alerts for unusual activity.

MY OVERALL UNDERSTANDING AND TAKEAWAY OF WEB SECURITY FUNDAMENTAL CONCEPTS

Below are key areas I've focused on to strengthen my understanding of web security

OWASP top 10 risks: understood the most common web application security risk and vulnerabilities like sql injection, xss and security misconfigurations and the extent to which they can affect a working application.

Threat Modeling: The threat modeling process and the steps you need to take to prepare a threat model which include,

- understanding the scope of work,
- identifying threats,
- determining countermeasures and
- assessing whether you have done a good job

Security testing: utilizing various methods, such as static analysis with tools like Semgrep, to identify and address vulnerabilities in web applications early in the development process.

Restful api security: following best practices to secure api endpoints through authentication, input validation, and rate limiting.

Secure coding practices: applying safe coding techniques such as data validation and error handling to prevent vulnerabilities.

Authentication and access control: implementing strong authentication methods and role-based access control for protecting sensitive data.