Functional Document

Identification and Counteraction of Benign, DDOS, DOS, and MITM utilizing AI calculations project

1. **Introduction**

The objective is to create AI-driven models proficient in precisely recognising and mitigating diverse cyber dangers, such as innocuous activities, DDoS, DoS, and MitM assaults, while guaranteeing efficient real-time detection and response. Sprint 1 prioritises the development of a secure user authentication system, incorporating comprehensive registration, session management, and role-based access controls to provide a basis for AI-driven threat detection in later phases.

2. **Product Goal**

The primary goal of this sprint is to devise and execute a secure user authentication system that guarantees access to the network is restricted to authorised users, establishing a basis for further AI-driven threat identification and mitigation.

3. **Demography (Users, Location)**

Users

Target Users: IT Professionals, Business Executives, and Developers

User Characteristics: Real-time threat monitoring, Focus on compliance, Technical expertise

Location

Target Location: Worldwide - Global Enterprises, Critical Infrastructure Sectors

4. **Business Processes**

The key business processes include:

1) User Authentication and Access Management:

   Secure login, role-based access control, and session management.

2) Threat Detection and Incident Response:

   Monitoring for cyber threats (DDoS, DoS, MitM), real-time detection, and mitigation.

3) Data Security and Compliance:

   Ensuring adherence to security regulations (e.g., GDPR, HIPAA) and protecting sensitive information.

4) Network Monitoring and Traffic Analysis:

   Analyzing network traffic for anomalies and ensuring system performance.

5) System Maintenance and Updates:

   Regular updates to the cybersecurity infrastructure and AI algorithms to stay ahead of emerging threats.

6) Reporting and Audit Logging:

   Tracking security events, generating reports, and maintaining audit logs for accountability and compliance.

5. **Features**

This sprint will focus on implementing the following key features:

1) **User Registration and Login:**

   Secure registration process with strong password policies, email verification, and multi-factor authentication (MFA).

2) **Role-Based Access Control (RBAC):**

   Assigning different access levels based on user roles (e.g., admin, user, guest) to ensure security.

3) **Session Management:**

   Secure session handling with features like session timeouts, secure cookies, and session hijacking prevention.

4) **Password Security:**

   Implementation of password hashing, salting, and encryption to protect user credentials.

5) **Threat Detection for Authentication:**

   Monitoring login attempts for anomalies, detecting brute-force attacks, and logging failed login attempts.

6) **User-Friendly Interface:**

   Simple, intuitive UI for users to manage login and authentication seamlessly while maintaining security.

6. **Authorization Matrix**

Define the roles and their corresponding access levels:

Role	Access Level
Admin	Full access to all system functionalities
User	Access to operating the system through their account
Guest	Limited access to basic login capabilities
Moderator	oversee user interactions and manage content

7. **Assumptions**

Assumptions for the Development Environment and Infrastructure

1. **Cloud-Based Infrastructure**: It is assumed that the development environment will utilize a cloud-based infrastructure (e.g., AWS, Azure, Google Cloud) to ensure scalability, flexibility, and ease of access for all team members.
2. **Version Control System**: It is assumed that a version control system (e.g., Git) will be used to manage code changes, facilitating collaboration and tracking modifications over time.
3. **Development Frameworks and Languages**: It is assumed that the team will use modern development frameworks (e.g., React, Node.js, Django) and programming languages (e.g., JavaScript, Python) that are suited for building secure and scalable web applications.
4. **Testing and Staging Environments**: It is assumed that dedicated testing and staging environments will be established to validate features and ensure stability before deployment to the production environment.

5. **Database Management**: It is assumed that a robust database management system (e.g., PostgreSQL, MongoDB) will be used to securely store user data, with appropriate backup and recovery processes in place.
6. **Security Protocols**: It is assumed that security best practices will be implemented from the outset, including regular vulnerability assessments and adherence to OWASP guidelines.
7. **Integration with Third-Party Services**: It is assumed that the system will integrate with third-party services (e.g., email providers for verification, authentication APIs) and that these services will be reliable and secure.
8. **Development Team Expertise**: It is assumed that the development team possesses the necessary expertise in cybersecurity and application development to build a secure user authentication system.
9. **Continuous Integration/Continuous Deployment (CI/CD)**: It is assumed that CI/CD practices will be implemented to automate testing and deployment processes, ensuring rapid iteration and timely updates.
10. **Monitoring and Logging Tools**: It is assumed that appropriate monitoring and logging tools will be used to track application performance and user activity, aiding in the detection of anomalies and security threats.

These assumptions provide a framework for the development environment and infrastructure necessary to support the user authentication system, ensuring it is built on a solid foundation that prioritizes security, scalability, and efficiency.