**Threat Modelling for Hospital Management App**

**Aim**

The aim of this threat modelling lab is to systematically identify, assess, and mitigate potential security threats within the Hospital Management App. By analysing the app’s architecture, data flows, and functionalities, we can pinpoint vulnerabilities and design effective security measures to protect against threats such as data breaches, unauthorized access, and attacks on sensitive medical information. ####

1. Understand the basics of threat modelling, including its purpose and importance in application security.

2. Identify key assets in the Hospital Management App, such as user credentials, patient data, doctor data, and appointment records.

3. Map out the app’s architecture and data flow, highlighting points of data entry, storage, and communication between entities.

4. Identify potential threats and attack vectors, including SQL injection, authentication bypass, and data breaches.

5. Classify the identified threats using a structured methodology like STRIDE (Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service, and Elevation of Privilege).

6. Evaluate the risks posed by these threats by considering their impact and likelihood.

7. Propose mitigation strategies for each identified threat to ensure the app is protected against common vulnerabilities.

8. Document the threat model, including identified threats, risk levels, and mitigation strategies, to be reviewed and referenced throughout the development process.

1.

- User Credentials: This includes usernames and passwords for authenticating different users such as hospital staff, doctors, and patients.

- Patient Data: Sensitive information including personal details, medical history, and treatment records.

- Doctor Data: Information related to healthcare providers, including their qualifications, specialties, and schedules.

- Appointment Records: Details about patient appointments, including times, dates, and related medical information.

- Surgical Records: Information pertaining to surgeries performed, including patient consent and surgical history.

- Audit Logs: Records of user actions within the application, essential for tracking access and changes to sensitive data.

2.

- Spoofing: - Attackers may attempt to impersonate legitimate users to gain unauthorized access to the app.

- Tampering: - Unauthorized modifications to patient data or surgical records could occur, compromising data integrity.

- Repudiation:

- Users may deny their actions within the app, such as accessing or modifying sensitive data.

- Information Disclosure:

- Sensitive patient information could be exposed through vulnerabilities such as SQL injection or improper access controls.

- Denial of Service (DoS):

- Attackers may attempt to overload the system to make it unavailable to legitimate users.

- Elevation of Privilege:

- Users may exploit vulnerabilities to gain higher access levels than authorized, allowing them to access restricted data.

Critical Risks: Tampering, Information Disclosure, and Elevation of Privilege.

- High Risks: Spoofing and Denial of Service.

- Medium Risks: Repudiation.

4. Propose Mitigation Strategies For each identified high-priority threat, the following mitigation strategies are proposed:

- Spoofing: - Implement strong user authentication methods, including multi-factor authentication (MFA).

- Regularly conduct user access reviews to ensure appropriate access levels.

- Tampering:

- Utilize cryptographic techniques to ensure data integrity (e.g., hashing). - Implement audit logging to track modifications and access to sensitive records.

- Information Disclosure:

- Employ input validation and prepared statements to prevent SQL injection attacks.

- Enforce strict access control policies, ensuring that users can only access data relevant to their roles.

- Denial of Service (DoS):

- Use rate limiting and throttling to manage excessive requests. - Implement redundancy and failover mechanisms to maintain availability.

-Elevation of Privilege:

- Conduct regular security assessments and penetration testing to identify and fix vulnerabilities.

- Apply the principle of least privilege (PoLP) to user roles and permissions.

- Repudiation:

- Ensure comprehensive logging of user activities with timestamps and user identifiers.

- Implement non-repudiation