SECURE SAAS SOFTWARE DEVELOPMENT PLAN

GOALS

1. Emphasize speed while keeping core security controls in place.
2. Protect high-risk areas like cloud storage, sensitive data, and file management.
3. Ensure flexibility in the system to support future features.

Development Phases

1. **Requirements Gathering**
    1) Focus on speed and user experience.
    2) Collect security needs like multi-factor authentication and data encryption for sensitive areas.
    3) List acceptable risks and prioritize securing high-risk areas.

2. **Architecture Design**
    1) Design a flexible system with modular components.
    2) Define boundaries around critical areas like user data and project management to control access.
    3) Plan for secure connections between services to limit insider threats.

3. **Threat Modeling**
    1) Identify high-risk areas, especially data breaches, malware, and insider misuse.
    2) Use basic threat modeling to understand security weak spots.
    3) Focus on major threats using the STRIDE model to address common risks.

4. **Implementation**
    1) Add MFA for sensitive actions and access to important data.
    2) Use encryption to protect data in transit and at rest.
    3) Work in short development cycles to maintain agility, addressing key security risks as they arise.

5. **Testing**
    1) Use automated tests to quickly check security during development.
    2) Perform penetration testing on high-risk areas, like cloud data storage.
    3) Track and fix security issues quickly, prioritizing those affecting critical functions.

6. **Deployment**
    1) Set up continuous deployment to release updates quickly while ensuring security checks.
    2) Use secrets management for handling sensitive data like API keys.
    3) Enable activity monitoring to detect suspicious actions in real-time.

7. **Post-Deployment Maintenance**
    1) Implement incident response processes for quick action in case of data breaches or malware.
    2) Use continuous monitoring to track activity in critical areas.
    3) Regularly update software to patch vulnerabilities without slowing down innovation.