2025-10-30

Nabu

Risk Management

1. Risks

ID	Topic	Description	Consequence
1	**Authentication Data Integrity**	User credentials could be stored or transmitted insecurely, leading to unauthorized access.	Data breaches, loss of user trust, and legal consequences (GDPR violations).
2	**Server Downtime**	Server or hosting environment becomes unavailable due to crashes or maintenance failures.	Application unavailable, loss of productivity, potential data loss.
3	**Data Loss or Corruption**	Improper database handling or failed backups may lead to permanent data loss.	Loss of important user or system data, costly recovery operations.
4	**Security Vulnerabilities**	Exploits such as SQL injection, XSS, or insecure APIs remain unpatched.	System compromise, unauthorized data access, damaged reputation.
5	**Version Control Conflicts**	Multiple developers merge conflicting changes in Git without review.	Broken features, lost code, and unstable releases.
6	**Dependency or Library Failure**	External libraries or APIs become outdated, insecure, or unavailable.	System malfunction or blocked deployments.
7	**Insufficient Testing**	Missing unit or integration tests allow bugs into production.	System instability, user frustration, increased maintenance cost.
8	**Project Time Overrun**	Delays in development due to underestimated workload or unclear requirements.	Missed deadlines, reduced quality, or incomplete features.
9	**Compliance & Privacy Risk**	Failure to meet GDPR or other legal data protection requirements.	Legal penalties, loss of credibility, and forced project changes.
10	**Team Communication Issues**	Unclear responsibilities or poor communication in the team.	Redundant work, misunderstandings, slower progress.

2. Management

ID	Probability	Impact	Priority / Prevention Measure
1	**Medium**	**High**	Use HTTPS, bcrypt for passwords, secure tokens, and regular security audits.
2	**Medium**	**Medium**	Set up monitoring (UptimeRobot, Prometheus), redundant backups, and automatic restarts.
3	**Low**	**High**	Implement automated database backups and test recovery procedures regularly.
4	**Medium**	**High**	Use static code analysis, regular dependency updates, and penetration testing.
5	**High**	**Medium**	Use Git branching strategy (feature branches, pull requests) and code review process.
6	**Medium**	**Medium**	Regularly check dependency updates and maintain version compatibility matrix.
7	**Medium**	**High**	Automate testing with CI/CD (GitHub Actions, PHPUnit, Jest) and enforce test coverage.
8	**High**	**High**	Agile sprints, clear milestones, and regular team reviews to track progress.
9	**Low**	**High**	Review GDPR compliance, add consent banners, anonymize data, and restrict access logs.
10	**Medium**	**Medium**	Weekly stand-ups, shared task board (Jira/Trello), and clear documentation.