

LGPD PARA DESENVOLVEDORES DE SISTEMAS

Privacidade, Segurança e Proteção de Dados na Engenharia de Software

Apostila didática ampliada e detalhada

Professor: Marcos Brandão

Versão: 2026

Objetivo da apostila: transformar os conceitos da LGPD em decisões concretas de análise, modelagem, programação, banco de dados, testes, implantação e manutenção de sistemas.

APRESENTAÇÃO

Vivemos em uma sociedade movida por dados. Cadastros, compras, acessos a aplicativos, registros escolares, transações financeiras e interações em redes sociais produzem informações capazes de identificar pessoas. Para quem desenvolve sistemas, isso significa que cada campo de formulário, tabela de banco de dados, log, cookie, API e rotina de backup pode envolver proteção de dados pessoais.

A Lei Geral de Proteção de Dados Pessoais - LGPD (Lei nº 13.709/2018) disciplina o tratamento de dados pessoais e busca proteger direitos fundamentais de liberdade, privacidade e o livre desenvolvimento da personalidade. Na prática de software, a lei não deve ser vista apenas como um texto jurídico: ela influencia requisitos, arquitetura, interfaces, permissões, retenção, segurança, auditoria e descarte de dados.

Esta versão amplia a apostila original, que já organizava o conteúdo em seis unidades, mas continha vários tópicos apresentados apenas como indicação de conteúdo, por exemplo “Explicação detalhada”, “Mostrar RBAC”, “Explicar cada etapa” e “Como evitar”. Aqui esses pontos são desenvolvidos em linguagem didática, com exemplos técnicos, situações reais, exercícios e laboratórios. A estrutura-base das seis unidades foi preservada.

Como estudar: Leia o conceito, observe o exemplo, tente prever o risco ou o comportamento do sistema, execute a atividade prática, altere o exemplo e finalize com os exercícios e desafios.

Sumário

- Unidade 1 - A LGPD sob a visão do Desenvolvedor
- Unidade 2 - Dados Pessoais, Sensíveis e Anonimização
- Unidade 3 - Princípios da LGPD aplicados ao Desenvolvimento
- Unidade 4 - Privacy by Design no Ciclo de Software
- Unidade 5 - Segurança da Informação para Desenvolvedores
- Unidade 6 - Banco de Dados e LGPD
- Projeto Integrador - Sistema Escolar aderente à LGPD
- Referências e glossário rápido

UNIDADE 1 - A LGPD SOB A VISÃO DO DESENVOLVEDOR

Objetivos: compreender por que a LGPD existe; relacionar programação e privacidade; reconhecer o papel do desenvolvedor; identificar riscos criados por decisões técnicas; distinguir os principais atores do tratamento.

1.1 O mundo é movido por dados

Dados são a matéria-prima de muitos sistemas. Um aplicativo de transporte precisa de localização; uma escola precisa identificar estudantes; um banco precisa autenticar clientes; uma loja precisa processar pedidos. O problema não está em usar dados, mas em coletá-los e tratá-los sem finalidade clara, sem proteção adequada ou além do necessário.

A LGPD usa a expressão tratamento de dados para abranger um ciclo muito maior do que “guardar no banco”. Tratamento inclui operações como coleta, recepção, classificação, uso, acesso, reprodução, transmissão, armazenamento, modificação, eliminação e outras operações realizadas com dados pessoais. Portanto, uma variável temporária, um arquivo de log, uma planilha exportada e um backup também podem fazer parte do tratamento.

Situação no sistema	Dado envolvido	Risco técnico possível
Login	usuário, senha, IP	senha em texto puro; log excessivo; brute force
Matrícula escolar	nome, CPF, endereço	coleta excessiva; acesso indevido
E-commerce	endereço, pedido, pagamento	exposição por API; retenção sem necessidade
App de saúde	sintomas, exames	tratamento de dado sensível sem controles reforçados

1.2 O papel do desenvolvedor

O controlador decide as finalidades e os elementos essenciais do tratamento; o operador realiza o tratamento em nome do controlador. O desenvolvedor pode trabalhar para um desses agentes e implementar tecnicamente suas decisões. Mesmo quando não decide juridicamente a finalidade, suas escolhas de implementação podem aumentar ou reduzir o risco.

- No formulário: define quais campos aparecem, quais são obrigatórios e quais validações serão aplicadas.
- No banco: define tabelas, relacionamentos, índices, permissões, criptografia e tempo de retenção.
- Na aplicação: define autenticação, autorização, sessões, logs, exportação e exclusão.
- Na integração: define quais dados uma API recebe, devolve e compartilha.
- Na manutenção: participa de correções, atualização de dependências, backup e resposta a incidentes.

Regra prática: antes de criar um campo, pergunte: por que precisamos deste dado, quem precisa acessá-lo, por quanto tempo e como ele será excluído?

1.3 Como um erro de programação pode causar exposição

Considere uma API de perfil. O problema não é a consulta SQL isoladamente, mas o fato de a aplicação não restringir o resultado ao usuário autenticado nem filtrar os campos devolvidos.

```
// Exemplo perigoso
SELECT * FROM usuarios;

// Melhor abordagem conceitual
SELECT id, nome, email FROM usuarios WHERE id = ?;
```

O uso de WHERE reduz o conjunto de registros; a seleção explícita de colunas reduz os dados retornados; a autorização garante que o usuário só acesse o que lhe é permitido; e prepared statements ajudam a evitar SQL Injection. São camadas diferentes de proteção.

1.4 Atores importantes: titular, controlador, operador e encarregado

Termo	Explicação prática	Exemplo em uma escola
Titular	pessoa natural a quem os dados se referem	aluno, responsável ou servidor
Controlador	decide finalidade e elementos essenciais do tratamento	instituição de ensino, conforme o caso
Operador	trata dados em nome do controlador	empresa contratada para hospedar/processar o sistema, conforme o contrato e a operação
Encarregado	canal de comunicação e função prevista na LGPD/regulamentação	pessoa indicada para atividades relacionadas à proteção de dados

1.5 Estudo de caso: matrícula online

Um formulário solicita nome, CPF, RG, título de eleitor, passaporte, tipo sanguíneo, religião, cor favorita, peso e altura. O primeiro passo não é perguntar se o banco suporta esses campos; é verificar se cada informação é adequada à finalidade da matrícula. Quanto mais dados desnecessários, maior a superfície de risco, maior o custo de proteção e maior o impacto de um eventual incidente.

Análise: nome e identificadores podem ser necessários conforme o processo da instituição; religião, cor favorita, peso ou passaporte podem não ter relação com a finalidade. A necessidade concreta depende do contexto e da base legal aplicável, mas “coletar porque pode ser útil no futuro” é uma prática ruim.

Atividades da Unidade 1

1. Liste cinco sistemas que você utiliza e identifique pelo menos três dados pessoais tratados por cada um.

2. Escolha um formulário real e marque campos que parecem essenciais, opcionais ou excessivos para a finalidade informada.
3. Explique por que `SELECT *` pode ser inadequado em uma API mesmo quando não existe SQL Injection.
4. Descreva a diferença entre decidir a finalidade do tratamento e apenas implementar tecnicamente uma funcionalidade.

Laboratório 1 - Revisão de formulário

5. Crie um formulário fictício de cadastro de aluno com 12 campos.
6. Para cada campo, escreva a finalidade e quem precisa acessá-lo.
7. Remova pelo menos três campos que não sejam necessários.
8. Classifique os campos restantes como obrigatórios ou opcionais e justifique.

UNIDADE 2 - DADOS PESSOAIS, SENSÍVEIS E ANONIMIZAÇÃO

Objetivos: classificar informações; compreender identificadores diretos e indiretos; reconhecer dados sensíveis; diferenciar anonimização e pseudonimização; avaliar riscos de reidentificação.

2.1 O que é dado pessoal?

Dado pessoal é a informação relacionada a pessoa natural identificada ou identificável.

“Identificada” significa que a pessoa já pode ser reconhecida diretamente, como por nome e CPF.

“Identificável” significa que a identificação pode surgir pela combinação de informações, como localização, identificador de dispositivo, IP, histórico de navegação ou outros elementos, dependendo do contexto.

Campo	Classificação típica	Por quê?
Nome completo	Pessoal	identifica diretamente
CPF/RG	Pessoal	identificador individual
Telefone/e-mail	Pessoal	relaciona-se a uma pessoa e permite contato/identificação
Endereço IP	Pode ser pessoal	pode permitir associação a um usuário ou dispositivo no contexto
Foto	Pessoal	pode identificar a pessoa
Placa do veículo	Depende do contexto	pode ser associada ao proprietário/condutor
Cookie/ID de dispositivo	Pode ser pessoal	pode individualizar ou compor perfil de usuário

2.2 Dados pessoais sensíveis

A LGPD atribui proteção reforçada a categorias que podem gerar discriminação ou atingir aspectos íntimos da pessoa. A lei inclui origem racial ou étnica, convicção religiosa, opinião política, filiação a sindicato ou organização de caráter religioso, filosófico ou político, dados de saúde ou vida sexual e dados genéticos ou biométricos quando vinculados a uma pessoa natural.

Para o desenvolvedor, isso significa tratar esses campos com maior atenção na modelagem, no controle de acesso, nos logs, nas integrações e nos ambientes de teste. Um sistema não deve copiar uma base real de dados de saúde para um ambiente de desenvolvimento apenas por conveniência.

Exemplo: uma foto é dado pessoal. Se dela for extraído um template biométrico usado para autenticar uma pessoa, entra-se no campo de dado biométrico vinculado ao titular, que é sensível.

2.3 Dado público não significa “dado sem proteção”

O fato de uma informação estar publicamente acessível não autoriza automaticamente qualquer reutilização. O tratamento continua exigindo análise de finalidade, contexto, princípios e hipótese legal. Para o desenvolvedor, copiar grandes bases públicas e combiná-las com perfis internos pode criar um tratamento novo e mais invasivo.

2.4 Anonimização

Anonimizar significa aplicar meios técnicos razoáveis e disponíveis para que o dado perca a possibilidade de associação direta ou indireta a uma pessoa. Quando a reidentificação continua razoavelmente possível, não se deve tratar o conjunto como verdadeiramente anonimizado.

```
-- Antes: identificação direta
id | nome  | cpf    | nota
1  | Marcos | 111... | 8.5

-- Depois: remover nome/CPF não basta sempre
id_pesquisa | faixa_idade | bairro | nota
A001        | 45-54       | Centro | 8.5
```

No segundo conjunto, os identificadores diretos foram removidos, mas combinações raras podem permitir reidentificação. Por isso anonimização é processo baseado em risco, não simples troca de nome por código.

2.5 Pseudonimização

Na pseudonimização, identificadores são substituídos por códigos ou tokens, mas existe informação adicional que permite reconstruir a associação. Essa chave deve ser mantida separadamente e protegida. Como a reidentificação continua possível, os dados pseudonimizados permanecem sujeitos à proteção de dados pessoais.

```
tabela_pesquisa: token_usuario = 'U8F2A1'
tabela_chaves (acesso restrito): U8F2A1 -> usuario_id 153
```

2.6 Dados em produção, teste e desenvolvimento

Uma prática comum e perigosa é clonar o banco de produção para testar uma nova versão. O ambiente de desenvolvimento costuma ter mais usuários, ferramentas de debug e controles menos rígidos. Prefira dados sintéticos ou conjuntos anonimizados adequadamente. Se dados reais forem indispensáveis, limite o conjunto, aplique controles e documente a necessidade.

Atividades da Unidade 2

9. Classifique: nome, CPF, religião, biometria, CEP, IP, e-mail, diagnóstico médico, matrícula escolar e cookie.
10. Explique por que substituir o nome por “Usuário 001” pode ser apenas pseudonimização.
11. Dê um exemplo de combinação de três dados aparentemente comuns que poderia reidentificar alguém.
12. Proponha uma estratégia para criar dados de teste sem copiar a base de produção.

Laboratório 2 - Classificação de banco

Monte uma planilha com 30 a 50 campos de um sistema escolar. Use as colunas: campo, exemplo, finalidade, pessoal?, sensível?, obrigatório?, quem acessa?, retenção e observações de segurança. O objetivo é transformar classificação jurídica em inventário técnico.

UNIDADE 3 - PRINCÍPIOS DA LGPD APLICADOS AO DESENVOLVIMENTO

Objetivos: entender os princípios do art. 6º e convertê-los em requisitos verificáveis de software.

3.1 Finalidade

O tratamento deve ter propósitos legítimos, específicos, explícitos e informados. Em software, a finalidade deve aparecer no requisito e não apenas na política de privacidade. Se um campo é coletado para entregar uma compra, ele não deve ser reutilizado silenciosamente para outra finalidade incompatível.

Aplicação prática: Requisito: “O sistema coletará o endereço para entrega do pedido e o exibirá apenas às funções responsáveis pela logística.”

3.2 Adequação

O tratamento precisa ser compatível com a finalidade informada e com o contexto. Um sistema escolar pode precisar de dados acadêmicos para gestão pedagógica; usar os mesmos dados para uma finalidade estranha ao serviço exige nova análise.

Aplicação prática: Teste: comparar cada uso do dado com a finalidade documentada no inventário.

3.3 Necessidade

É a minimização: limitar o tratamento ao mínimo necessário. Para programadores, afeta campos, payloads de API, logs, backups, relatórios e permissões.

Aplicação prática: Newsletter: normalmente nome e e-mail podem bastar; pedir RG, CNH e tipo sanguíneo seria excessivo.

3.4 Livre acesso

O titular deve poder obter informações sobre forma e duração do tratamento e acessar seus dados nos termos da lei. O sistema pode apoiar isso com uma área “Meus Dados”, mecanismos de consulta e processos de exportação.

Aplicação prática: Funcionalidade: tela de perfil com dados cadastrados e canal para solicitações.

3.5 Qualidade dos dados

Os dados devem ser exatos, claros, relevantes e atualizados conforme a necessidade. Cadastros duplicados e informações desatualizadas podem causar decisões erradas.

Aplicação prática: Implementação: validação, confirmação de e-mail, normalização e rotinas de atualização.

3.6 Transparência

Informações sobre o tratamento devem ser claras, precisas e facilmente acessíveis, respeitados segredos comercial e industrial. Interfaces obscuras e textos genéricos prejudicam a compreensão.

Aplicação prática: UI: explicar junto ao campo por que o dado é solicitado quando isso ajuda o usuário.

3.7 Segurança

Exige medidas técnicas e administrativas aptas a proteger dados contra acessos não autorizados e situações acidentais ou ilícitas.

Aplicação prática: Controles: autenticação, autorização, HTTPS, hash de senha, atualização, backup, monitoramento.

3.8 Prevenção

A organização deve agir antes do incidente. No desenvolvimento, isso significa threat modeling, revisão de código, testes, atualização de bibliotecas e configuração segura.

Aplicação prática: Prática: checklist de segurança antes de publicar uma nova versão.

3.9 Não discriminação

Dados não devem ser tratados para fins discriminatórios ilícitos ou abusivos. Algoritmos e regras de negócio também precisam ser avaliados.

Aplicação prática: Exemplo: um sistema de seleção não deve usar atributos protegidos para excluir candidatos de forma discriminatória.

3.10 Responsabilização e prestação de contas

Não basta afirmar conformidade: é necessário demonstrar medidas eficazes. Logs adequados, documentação, inventário, registros de decisão e controles de acesso ajudam a produzir evidências.

Aplicação prática: Evidência: registro de alteração de permissão com usuário administrador, data e motivo.

3.11 Transformando princípios em critérios de aceite

Princípio	Pergunta de revisão	Critério técnico possível
Finalidade	Por que este campo existe?	cada campo possui finalidade documentada
Necessidade	Podemos fazer com menos dados?	API retorna apenas colunas necessárias
Livre acesso	Como o titular consulta?	área de dados + processo de solicitação
Qualidade	Como corrigir dados?	edição controlada e validação
Segurança	Quem pode acessar?	RBAC + autenticação + auditoria
Prestação de contas	Como provar o controle?	logs, documentação e evidências de teste

Atividades da Unidade 3

13. Escolha uma tela do seu sistema e associe pelo menos cinco princípios a requisitos concretos.

14. Reescreva um cadastro excessivo aplicando o princípio da necessidade.
15. Defina três critérios de aceite para transparência em uma tela de consentimento ou informação ao usuário.
16. Explique por que “temos firewall” não é suficiente para demonstrar o princípio da segurança.

UNIDADE 4 - PRIVACY BY DESIGN NO CICLO DE SOFTWARE

Ideia central: privacidade deve ser considerada desde a concepção e ao longo de todo o ciclo de vida do produto, não adicionada somente depois que o sistema está pronto.

4.1 O que é Privacy by Design

Privacy by Design é uma abordagem de engenharia e governança que incorpora privacidade à arquitetura, aos processos e às escolhas padrão do sistema. A expressão ficou associada ao trabalho de Ann Cavoukian e aos sete princípios fundacionais. Para uma equipe de software, a ideia mais importante é antecipar riscos e construir proteção como requisito de projeto.

4.2 Os sete princípios fundacionais - leitura para desenvolvedores

Proativo, não reativo

identificar riscos antes de incidentes; fazer revisão de requisitos e ameaças antes da implementação.

Privacidade como padrão

configurações iniciais devem favorecer a proteção; não depender de o usuário descobrir opções escondidas.

Privacidade incorporada ao design

proteção faz parte da arquitetura, não é um “plugin” posterior.

Funcionalidade total

buscar soluções que conciliem objetivo do negócio e privacidade, evitando o falso dilema “ou funciona ou protege”.

Segurança de ponta a ponta

proteger dados da coleta ao descarte, incluindo transmissão, armazenamento, backup e eliminação.

Visibilidade e transparência

processos e controles devem ser verificáveis e explicáveis.

Respeito à privacidade do usuário

interfaces e fluxos devem colocar o titular no centro, com informações compreensíveis e controles adequados.

4.3 Aplicação no ciclo de desenvolvimento

Etapa	Perguntas de privacidade	Entregável técnico
Levantamento	quais dados? finalidade? base legal? titulares? compartilhamentos?	inventário inicial e requisitos de privacidade
Modelagem	onde armazenar? relacionamentos? segregação?	modelo de dados minimizado e matriz de acesso

Programação	como autenticar, autorizar, validar e registrar?	código seguro e tratamento de erros
Testes	há exposição em logs, APIs, mensagens ou massas de teste?	casos de teste de privacidade e segurança
Implantação	HTTPS? segredos? permissões? backups?	checklist de hardening e configuração
Operação/Auditoria	quem acessou? retenção? incidentes?	monitoramento, logs e revisão periódica
Descarte	como eliminar cópias e backups conforme regras?	rotina documentada de retenção e eliminação

4.4 Perguntas para o levantamento de requisitos

- Qual problema de negócio exige o dado?
- O mesmo resultado pode ser obtido com menos dados?
- O dado é pessoal ou sensível?
- Quem são os titulares?
- Quem terá acesso e por qual papel?
- Há compartilhamento com terceiros?
- Qual é a hipótese legal definida pelo responsável pelo tratamento?
- Quanto tempo o dado precisa permanecer?
- Como o titular exercerá direitos aplicáveis?
- Como ocorrerão correção, exportação, bloqueio ou exclusão quando cabíveis?
- Quais riscos existem em caso de vazamento ou uso indevido?

4.5 Privacy by Default

Configurações padrão importam porque muitos usuários nunca as alteram. Um sistema que publica o perfil por padrão e exige que o usuário encontre uma opção escondida para torná-lo privado cria mais exposição. Em aplicações corporativas, o equivalente é conceder permissões amplas por padrão. O padrão recomendado é o menor privilégio necessário.

4.6 Projeto prático: sistema escolar

Modele um sistema com alunos, turmas, professores, atividades e notas. Antes de criar tabelas, produza um mapa de dados. Para cada entidade, defina finalidade, classificação, papéis autorizados, retenção e operações permitidas. Depois desenhe o modelo lógico.

Desafio: compare duas arquiteturas: (A) todos os funcionários consultam todos os dados; (B) cada papel acessa apenas o necessário. Explique como a segunda reduz risco e facilita auditoria.

Atividades da Unidade 4

17. Crie cinco requisitos funcionais e cinco requisitos não funcionais de privacidade para um AVA.
18. Desenhe o fluxo de vida de um dado desde o formulário até a exclusão.
19. Escolha uma configuração do sistema e defina qual deveria ser o padrão mais protetivo.

20. Faça uma matriz simples de papéis x dados para administrador, professor e aluno.

UNIDADE 5 - SEGURANÇA DA INFORMAÇÃO PARA DESENVOLVEDORES

Objetivos: entender controles essenciais de segurança e sua relação com proteção de dados; aplicar hash, criptografia, HTTPS, autenticação, autorização, logs e backups de forma adequada.

5.1 Segurança não é um único recurso

Segurança funciona em camadas. HTTPS não corrige senha fraca; hash de senha não impede usuário comum de acessar uma rota administrativa mal protegida; backup não evita SQL Injection. Um sistema seguro combina prevenção, detecção, resposta e recuperação.

5.2 Hash de senhas

Hash é uma função de mão única: transforma uma entrada em uma saída de tamanho definido e não foi projetada para ser revertida. Para senhas, devem ser usados algoritmos próprios para armazenamento de credenciais, lentos e resistentes a ataques de força bruta, como bcrypt ou Argon2, com salt. MD5 e SHA-1 não são adequados para armazenamento moderno de senhas.

```
// PHP - exemplo recomendado
$hash = password_hash($senha, PASSWORD_DEFAULT);

if (password_verify($senhaDigitada, $hash)) {
    // autenticação válida
}
```

O salt é um valor único associado ao cálculo do hash e dificulta o uso de tabelas pré-computadas. Em PHP, password_hash gerencia os detalhes necessários conforme o algoritmo escolhido. Nunca armazene a senha original para “lembrar” ao usuário; em recuperação, use fluxo de redefinição com token temporário.

5.3 Hash x criptografia

Recurso	Pode ser revertido?	Uso típico
Hash de senha	não, por projeto	verificar credenciais sem guardar a senha
Criptografia simétrica (ex.: AES)	sim, com a chave	proteger dados que precisam ser recuperados
Criptografia assimétrica (ex.: RSA/ECC)	sim, com chave apropriada	troca de chaves, assinatura e cenários criptográficos específicos

Criptografar um dado só é útil se a chave for protegida. Guardar a chave no mesmo arquivo público do código ou no repositório elimina grande parte do benefício. Segredos devem ser separados do código e ter acesso restrito.

5.4 HTTPS, TLS e certificados

HTTPS é HTTP protegido por TLS. Ele ajuda a fornecer confidencialidade e integridade durante o transporte e autentica o servidor por certificado. Sem HTTPS, credenciais e dados enviados entre navegador e servidor podem ficar expostos na rede. Certificados gratuitos podem ser obtidos por autoridades como Let's Encrypt, conforme a infraestrutura utilizada.

5.5 Autenticação, sessão e cookies

Autenticação responde “quem é você?”. Após o login, aplicações web costumam manter estado por sessão ou tokens. Cookies de sessão devem usar atributos adequados, como Secure quando há HTTPS, HttpOnly para reduzir acesso por scripts e SameSite conforme o fluxo da aplicação. IDs de sessão devem ser imprevisíveis e renovados após autenticação para reduzir riscos como session fixation.

```
// Exemplo conceitual PHP
session_start();
session_regenerate_id(true);
$_SESSION['usuario_id'] = $usuario['id'];
$_SESSION['papel'] = $usuario['papel'];
```

5.6 Autorização e RBAC

Autorização responde “o que você pode fazer?”. RBAC (Role-Based Access Control) organiza permissões por papéis. Esconder um botão no HTML não é autorização; o servidor deve verificar a permissão em cada operação sensível.

Papel	Pode ver própria nota	Pode lançar nota	Pode gerenciar usuários
Aluno	Sim	Não	Não
Professor	Conforme turma/disciplina	Sim, conforme vínculo	Não
Administrador	Conforme necessidade administrativa	Conforme regra	Sim

```
if (!isset($_SESSION['papel']) || $_SESSION['papel'] !== 'admin') {
    http_response_code(403);
    exit('Acesso negado');
}
```

5.7 JWT e OAuth - onde entram

JWT é um formato de token assinado que pode transportar claims. Ele não é “login seguro automaticamente”: é preciso validar assinatura, expiração, emissor, audiência e estratégia de revogação quando aplicável. OAuth 2.0 é um framework de autorização usado para conceder acesso delegado; OpenID Connect adiciona uma camada de identidade sobre OAuth 2.0. Em sistemas simples, sessões tradicionais podem ser mais fáceis de administrar corretamente.

5.8 Logs: o que registrar e o que evitar

Logs são importantes para auditoria e investigação, mas também podem virar uma nova base de dados pessoais. Registre eventos necessários: login bem-sucedido ou falho, alterações de permissão, operações administrativas, erros relevantes e identificadores técnicos. Evite registrar senha, token completo, número de cartão, conteúdo sensível desnecessário ou payloads inteiros sem avaliação.

Exemplo ruim: ERRO login senha=MinhaSenha123 token=eyJ... CPF=...

Exemplo melhor: 2026-09-14 18:10 login_falhou usuario_id=153 ip=... motivo=credencial_invalida

5.9 Backups

Backup protege disponibilidade, mas cria cópias adicionais. Defina frequência, retenção, criptografia quando apropriada, controle de acesso e testes de restauração. Um backup nunca testado pode falhar justamente no incidente. A política de retenção deve considerar também as cópias de segurança e os procedimentos de expurgo.

5.10 Resposta a incidentes

Um incidente pode envolver acesso não autorizado, perda, alteração, divulgação ou indisponibilidade de dados. A equipe técnica deve saber preservar evidências, conter a falha, trocar credenciais quando necessário, corrigir a causa e fornecer informações para a avaliação jurídica/regulatória. A comunicação de incidentes à ANPD e aos titulares segue a LGPD e regulamentação específica; não cabe ao desenvolvedor improvisar sozinho a decisão de notificar.

Atividades da Unidade 5

21. Implemente um cadastro que armazene senha com `password_hash` e valide com `password_verify`.
22. Crie três rotas e aplique autorização diferente para aluno, professor e administrador.
23. Analise um arquivo de log fictício e marque informações que não deveriam estar registradas.
24. Desenhe uma política simples de backup com frequência, retenção, responsável e teste de restauração.
25. Explique por que criptografar o banco e deixar a chave pública no repositório é uma falsa sensação de segurança.

UNIDADE 6 - BANCO DE DADOS E LGPD

Objetivos: modelar dados com minimização, separar finalidades, controlar privilégios, usar consultas seguras, planejar retenção, exclusão e auditoria.

6.1 Modelagem orientada à finalidade

Uma boa modelagem começa pelo requisito, não pela vontade de guardar “tudo”. Cada coluna deve ter finalidade. Campos opcionais devem ser realmente opcionais. Dados sensíveis podem exigir tabelas ou controles separados quando isso melhora a segregação e o acesso. O modelo também precisa representar consentimentos quando essa for a hipótese legal utilizada, sem tratar consentimento como base universal para qualquer operação.

6.2 Exemplo de estrutura para sistema escolar

```
CREATE TABLE usuarios (  
  id INT PRIMARY KEY AUTO_INCREMENT,  
  nome VARCHAR(120) NOT NULL,  
  email VARCHAR(160) NOT NULL UNIQUE,  
  senha_hash VARCHAR(255) NOT NULL,  
  papel ENUM('aluno','professor','admin') NOT NULL,  
  criado_em DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP  
);  
  
CREATE TABLE alunos (  
  usuario_id INT PRIMARY KEY,  
  matricula VARCHAR(30) NOT NULL UNIQUE,  
  FOREIGN KEY (usuario_id) REFERENCES usuarios(id)  
);
```

Observe que senha é armazenada como hash; e-mail e matrícula possuem unicidade; e o papel permite controle de autorização. O modelo real deve ser adaptado aos requisitos e às regras da instituição.

6.3 Consentimentos e registros de preferência

Quando consentimento for a hipótese legal apropriada, o sistema deve registrar evidências suficientes para demonstrar o que foi consentido, em que momento e para qual finalidade, além de suportar revogação nos termos aplicáveis. Evite uma coluna genérica consentiu=1 para finalidades diferentes.

```
CREATE TABLE consentimentos (  
  id BIGINT PRIMARY KEY AUTO_INCREMENT,  
  usuario_id INT NOT NULL,  
  finalidade VARCHAR(120) NOT NULL,  
  versao_texto VARCHAR(20) NOT NULL,  
  concedido_em DATETIME NOT NULL,  
  revogado_em DATETIME NULL  
);
```

6.4 Exclusão lógica x exclusão física

Exclusão lógica marca o registro como inativo, por exemplo `deleted_at`, mas mantém os dados. Ela é útil para certas necessidades operacionais e auditoria, porém não equivale à eliminação física. Exclusão física remove o registro do banco, embora cópias possam permanecer em backups até o ciclo de retenção. A escolha depende da finalidade, das obrigações de retenção e das regras aplicáveis.

Cuidado: não use “soft delete” como desculpa para conservar indefinidamente todos os dados. Retenção precisa ter justificativa e prazo/regras definidos.

6.5 Auditoria e trilhas de alteração

Auditoria deve registrar eventos relevantes sem copiar dados sensíveis desnecessariamente. Para alterações críticas, pode ser útil registrar quem alterou, quando, qual entidade foi afetada e o tipo de mudança. Triggers podem ajudar em cenários específicos, mas aumentam complexidade e não substituem uma estratégia de auditoria bem projetada.

```
CREATE TABLE auditoria (  
  id BIGINT PRIMARY KEY AUTO_INCREMENT,  
  usuario_id INT NULL,  
  evento VARCHAR(80) NOT NULL,  
  entidade VARCHAR(80) NOT NULL,  
  entidade_id BIGINT NULL,  
  criado_em DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP  
);
```

6.6 Permissões SQL e menor privilégio

A aplicação não precisa se conectar ao MariaDB como root. Crie um usuário de banco com apenas as permissões necessárias. Separar contas administrativas das contas da aplicação reduz o impacto de uma falha.

```
-- Exemplo conceitual; ajuste ao seu ambiente  
CREATE USER 'app_ava'@'localhost' IDENTIFIED BY 'senha-forte-e-unica';  
GRANT SELECT, INSERT, UPDATE, DELETE ON ava.* TO 'app_ava'@'localhost';  
FLUSH PRIVILEGES;
```

6.7 SQL Injection

SQL Injection ocorre quando dados fornecidos pelo usuário são incorporados de forma insegura ao comando SQL, permitindo alterar a estrutura lógica da consulta. Escapar manualmente strings é uma abordagem frágil. O padrão recomendado é usar prepared statements e parâmetros.

```
// PERIGOSO  
$sql = "SELECT * FROM usuarios WHERE email = '$email'";  
  
// MELHOR - PDO  
$stmt = $pdo->prepare("SELECT id, nome, senha_hash, papel FROM usuarios WHERE email = ?");  
$stmt->execute([$email]);  
$usuario = $stmt->fetch();
```

6.8 Criptografia de colunas

Criptografar determinadas colunas pode reduzir impacto em caso de acesso indevido ao arquivo do banco, mas exige gestão de chaves, rotação, backup das chaves e análise de como pesquisar/ordenar os dados. Nem toda coluna deve ser criptografada na aplicação; primeiro avalie ameaça, sensibilidade e necessidade. Senhas, por exemplo, devem usar hash próprio para credenciais, não criptografia reversível.

6.9 Retenção e descarte

Crie regras de retenção por categoria, não um prazo único para todo o banco. Alguns registros podem precisar ser mantidos por obrigações legais ou contratuais; outros podem ser eliminados após a finalidade. A equipe técnica deve receber essas regras de quem responde pela governança/jurídico e implementá-las em rotinas verificáveis.

Categoria	Regra técnica a definir	Pergunta
Conta ativa	manter enquanto necessária	quais dados são essenciais à conta?
Logs	retenção limitada e protegida	quanto tempo é útil para segurança/auditoria?
Tokens de recuperação	expiração curta	o token é invalidado após uso?
Backups	ciclo de retenção	quando cópias antigas são destruídas?
Conta encerrada	eliminação/retensão conforme justificativa	o que deve ser apagado e o que deve permanecer por obrigação?

6.10 Backup, replicação e ambientes

O inventário de dados deve incluir réplicas, dumps, arquivos CSV exportados e backups. Apagar uma linha do banco principal não elimina automaticamente todas as cópias. O projeto deve documentar onde os dados circulam e como as cópias envelhecem até o descarte.

Atividades da Unidade 6

26. Crie um modelo MariaDB com usuários, alunos, turmas, atividades, notas e auditoria.
27. Defina quais campos são obrigatórios, opcionais e desnecessários e justifique.
28. Implemente login com prepared statement e password_verify.
29. Crie uma conta de banco para a aplicação sem privilégios administrativos globais.
30. Escreva uma rotina conceitual para expirar tokens de recuperação e remover registros antigos conforme uma política definida.

PROJETO INTEGRADOR - SISTEMA ESCOLAR ADERENTE À LGPD

O projeto final reúne as seis unidades. O objetivo não é declarar “conformidade jurídica” apenas porque o sistema possui senha e HTTPS. O aluno deverá demonstrar como requisitos, dados, acessos, segurança, transparência e retenção foram pensados.

Etapas do projeto

31. Levantamento: descrever finalidade do sistema, titulares, funcionalidades e dados necessários.
32. Inventário: classificar cada campo como pessoal, sensível quando aplicável, obrigatório/opcional, finalidade e retenção.
33. Modelagem: criar MER e esquema relacional minimizado.
34. Controle de acesso: definir papéis aluno, professor e administrador e suas permissões.
35. Segurança: senha com hash, prepared statements, HTTPS no ambiente publicado, sessão segura e validação de entrada.
36. Transparência: criar tela “Meus Dados” e texto simples explicando o uso das informações.
37. Auditoria: registrar eventos administrativos relevantes sem expor segredos.
38. Retenção: documentar o que acontece quando uma conta é encerrada.
39. Testes: executar casos de acesso indevido, SQL Injection, sessão e autorização.
40. Documentação: entregar matriz de dados, matriz de acesso, diagrama, evidências de testes e conclusão.

Rubrica sugerida

Critério	Peso
Minimização e classificação de dados	20%
Modelagem e banco	20%
Autenticação/autorização e consultas seguras	20%
Privacy by Design e transparência	15%
Logs, backup e retenção	15%
Documentação e apresentação	10%

CHECKLIST DO DESENVOLVEDOR

- ☐ Sei a finalidade de cada dado coletado?
- ☐ Coleta somente o necessário?
- ☐ Dados sensíveis possuem controles reforçados?
- ☐ A aplicação usa HTTPS?
- ☐ Senhas usam algoritmo adequado de hash?
- ☐ Consultas usam prepared statements?
- ☐ A autorização é verificada no servidor?
- ☐ A conta do banco segue menor privilégio?

- ☐ Logs evitam senhas, tokens e dados excessivos?
- ☐ Backups são protegidos e testados?
- ☐ Existe regra de retenção e descarte?
- ☐ O titular consegue obter informações e exercer direitos por processo definido?
- ☐ Ambientes de teste evitam dados reais desnecessários?
- ☐ Há evidências de testes e decisões de segurança?

GLOSSÁRIO RÁPIDO

Dado pessoal: informação relacionada a pessoa natural identificada ou identificável.

Dado pessoal sensível: categoria de dado pessoal com proteção reforçada definida pela LGPD.

Titular: pessoa natural a quem se referem os dados.

Tratamento: operações realizadas com dados pessoais ao longo de seu ciclo de vida.

Controlador: agente a quem competem decisões referentes ao tratamento.

Operador: agente que realiza tratamento em nome do controlador.

Anonimização: processo que busca retirar a possibilidade de associação do dado a um indivíduo considerando meios técnicos razoáveis e disponíveis.

Pseudonimização: separação/substituição de identificadores mantendo informação adicional capaz de restabelecer a associação.

RBAC: controle de acesso baseado em papéis.

Hash: transformação unidirecional usada, entre outros fins, no armazenamento seguro de senhas com algoritmos apropriados.

Criptografia: transformação reversível mediante chave apropriada para proteger confidencialidade.

Prepared statement: consulta parametrizada que separa dados da estrutura SQL e ajuda a prevenir SQL Injection.

Retenção: regras sobre por quanto tempo dados e cópias permanecem armazenados.

REFERÊNCIAS PARA APROFUNDAMENTO

- BRASIL. Lei nº 13.709, de 14 de agosto de 2018. Lei Geral de Proteção de Dados Pessoais (LGPD), texto compilado. Presidência da República / Planalto.
- AGÊNCIA NACIONAL DE PROTEÇÃO DE DADOS (ANPD). Materiais Educativos e Publicações: guias orientativos, glossário e documentos técnicos.
- ANPD. Guia Orientativo sobre Segurança da Informação para Agentes de Tratamento de Pequeno Porte.
- ANPD. Glossário de Proteção de Dados Pessoais e da Privacidade.
- ANPD. Estudos técnicos sobre anonimização de dados na LGPD.
- OWASP Foundation. OWASP Top 10 e materiais de segurança de aplicações web, como referência técnica complementar.
- CAVOUKIAN, Ann. Privacy by Design: princípios fundacionais, como referência conceitual histórica da abordagem.

Nota didática: esta apostila apresenta conceitos técnicos e educacionais e não substitui análise jurídica específica. A hipótese legal, os prazos de retenção e as obrigações aplicáveis dependem do contexto concreto do tratamento.