Sample ERD Document — Simple Product (with intentional vulnerabilities for RAG detection)

Introduction:
This document describes a compact entity-relationship model for a simple multi-tenant web product: 'TaskBoard' (a task management app).
It includes an architecture overview, entity descriptions, data flows, and a deliberately small set of intentional vulnerabilities included for the purpose of testing RAG detection capabilities.

Architecture Overview:
- Relational database (Postgres) for tenants, users, projects, tasks, comments.
- Object storage (S3) for uploaded attachments.
- REST API (behind API Gateway/Lambda).
- DynamoDB for session cache and rate limiting.
- EventBridge for domain events.

Core Entities (conceptual):
- Tenant: id, name, plan_tier, created_at
- User: id, tenant_id, email, display_name, cognito_sub, is_active
- Project: id, tenant_id, name, description
- Task: id, project_id, title, description, status, assignee_id, created_at, due_date
- Comment: id, task_id, user_id, body, created_at
- Attachment: id, task_id, s3_key, filename, content_type, size_bytes
- AuditLog: id, tenant_id, actor_user_id, action, target_type, target_id, occurred_at

Data Flow Summary:
Clients authenticate via Cognito and call the REST API. API validates requests, performs RBAC checks, and persists records in Postgres.
Attachments are uploaded to S3 using presigned URLs. Events are published to EventBridge for downstream consumers.

Intentional Vulnerabilities (for RAG detection):
VULN-01: Hardcoded admin credentials in config file
- Detectable indicators: strings like 'ADMIN_PASSWORD=' or 'admin:Admin123' in repo/config; plaintext secrets in files.
- Mitigation (high-level): Remove secrets from code; use Secrets Manager; rotate credentials.

VULN-02: Public S3 bucket for attachments
- Detectable indicators: S3 policies with Principal '*' or public ACLs; missing Block Public Access.
- Mitigation (high-level): Enable Block Public Access; use presigned URLs; tighten bucket policies.

VULN-03: Missing input validation on task creation endpoint
- Detectable indicators: API code that inserts fields into DB without validation; lack of parameterized queries.
- Mitigation (high-level): Validate/sanitize inputs; use parameterized queries/ORM protections.

VULN-04: Exposed debug endpoint in production
- Detectable indicators: routes like '/debug' or ENV=development in deployed config.
- Mitigation (high-level): Remove debug endpoints in prod; restrict via auth and network controls.

VULN-05: Insufficient IAM permissions for lambda role
- Detectable indicators: IAM policies with 's3:*' or 'dynamodb:*' or use of '*' in actions.
- Mitigation (high-level): Use least-privilege IAM policies scoped to ARNs.

RAG Detection Hints (non-actionable):
- Index configuration files (.env, YAML, JSON) and flag secrets-like patterns.
- Scan IaC templates for permissive IAM and S3 configurations.

- Inspect OpenAPI specs for unvalidated endpoints.
- Code search for suspicious strings: 'admin', 'password', 'debug', 'allow *'.

Responsible Use Note:
This document is intended for defensive testing and detection training only. It contains high-level weaknesses to exercise RAG/analysis pipelines.
It does NOT provide exploit instructions. Perform testing only in authorized environments.

End of document.