Untitled live doc 2025-10-24

THREAT ID	COMPONENT NAME	THREAT NAME	STRIDE CATEGORY	WHY APPLICABLE	HOW MITIGATED	MITIGATION	LIKELIHOOD EXPLANATION	IMPACT EXPLANATION	RISK SEVERITY	**Comments**
0001	API Gateway	Attacker bypasses authentication by exploiting JWT token validation vulnerabilities to access core services	Spoofing	API Gateway is the single entry point for all customer and staff traffic. JWT tokens are used for authentication across microservices. Misconfigured validation or algorithm confusion (e.g., none algorithm) could allow unauthorized access.	Partially mitigated. ADR-01 mentions authentication (OAuth2/JWT) at API Gateway, but does not specify JWT library, algorithm enforcement (RS256 vs HS256), or token expiration policies.	Enforce strict JWT validation: use RS256 (asymmetric) not HS256 (symmetric), validate issuer/audience claims, implement short token expiration (15-30 min access tokens), use refresh token rotation, deploy WAF with JWT attack signatures, audit JWT library for	Medium. JWT vulnerabilities are well-documented and exploited in real-world attacks. However, likelihood depends on implementation quality. If using mature libraries with defaults, risk is lower.	High. Successful bypass grants attacker access to core services (booking, payment, user data), enabling data exfiltration, financial fraud, and service disruption.	High	Mitigation implemented. Additionally strict enforcement of JWT tokens using auth middleware plugins added to the gateway itself preventing unauthenticated endpoint exposure without proper procedure.

						known CVEs.				
0002	Vehicle Edge Device	Attacker extracts mTLS certificates from compromised vehicle hardware to impersonate legitimate vehicles and inject false telemetry	Spoofing	Vehicles use mTLS for authentication (ADR-11). Physical access to vehicles is possible (parked in public spaces). Attacker could extract private keys from embedded hardware via JTAG, firmware extraction, or side-channel attacks.	Partially mitigated. ADR-11 specifies mTLS for authentication, but does not mention hardware security modules (HSM), secure enclaves, or certificate rotation policies.	Use hardware-backed secure enclaves (ARM TrustZone, TPM) to store private keys, preventing extraction even with physical access. Implement short-lived certificates (24-hour validity) with automated rotation. Add device attestation (signed boot, secure firmware) to verify device integrity. Monitor for anomalous telemetry	Medium. Physical access to vehicles is easy, but extracting keys from secure hardware is difficult without sophisticated tools. Insider threat (malicious technicians) increases likelihood.	High. Attacker could inject false telemetry (fake GPS locations, battery status) to manipulate fleet operations, pricing, and maintenance. Could disable collision detection or exfiltrate real telemetry from legitimate vehicles.	High	Mitigation implemented. Additionally unique identifiers associated with each vehicle incase of duplicate/ irregular data. E.g multiple diff location or location jumps - alerts are triggered by the anomaly detection pipeline.

						patterns indicating compromised devices.				
0003	Kafka Event Bus	Unauthorized service reads sensitive events from Kafka topics containing customer PII and payment data	Information Disclosure	Kafka is the central event bus (ADR-06) with topics like bookings.created, customers.app_events, photos.uploaded. Without topic-level ACLs, any compromised microservice could read all events, including PII and payment data.	Not mitigated. ADR-06 does not mention Kafka ACLs, encryption, or authorization. Default Kafka installations have no access control.	Enable Kafka ACLs (topic-level read/write permissions), use SASL/SCRAM or mTLS for client authentication, encrypt data in transit (TLS) and at rest (encrypted volumes). Implement principle of least privilege: each service should only access required topics. Audit Kafka access logs regularly.	High. Default Kafka configurations lack access control. Compromising any microservice (via dependency vulnerability, RCE, etc.) grants access to all topics. This is a common misconfiguration.	Critical. Kafka contains PII (customer locations, behavior), payment data (authorization flows), and operational secrets. Exposure violates GDPR/DPDP compliance, enables identity theft, financial fraud, and competitive intelligence gathering.	Critical	Mitigation implemented. Additionally VPC network has clear subnet firewall separation to enforce separation of concerns.

0004	Multi-Provider AI Orchestrator	API keys for external LLM providers (OpenAI, Anthropic, Gemini, Azure) exposed in environment variables or config files	Information Disclosure	ADR-12 implements multi-provider AI with routing to OpenAI, Anthropic, Gemini, Azure. API keys likely stored in environment variables or config files. Exposed keys allow unauthorized usage, cost inflation, and data exfiltration via third-party LLMs.	Not mitigated. ADR-12 does not mention secrets management. Environment variables and config files are commonly checked into version control or logged.	Store API keys in dedicated secrets manager (AWS Secrets Manager, HashiCorp Vault, Azure Key Vault). Use IAM roles for cloud-native key retrieval. Rotate keys regularly (quarterly). Implement cost alerts and rate limiting per provider. Audit API usage logs for anomalies. Never commit secrets to version control (use pre-commit hooks, secret	High. Storing secrets in environment variables is a common practice, and accidental exposure via logs, error messages, or version control is frequent. Recent GitHub secret leaks demonstrate real-world prevalence.	High. Exposed API keys allow attacker to consume AI services at victim's expense (financial loss), send malicious prompts to extract training data or bypass safety filters, exfiltrate customer queries sent to LLMs (privacy violation), or deplete rate limits causing service outage.	High	Mitigation implemented. Additionally prefer going with OIDC/OAuth short lived tokens where supported rather than static api-key.

						scanning).				
0005	PostgreSQL Database	SQL injection in booking or payment services allows attacker to exfiltrate customer PII and payment data	Tampering / Information Disclosure	Core services (Booking, Payment, User/KYC) use PostgreSQL (ADR-01). SQL injection is a persistent vulnerability if parameterized queries are not enforced. Successful injection allows data exfiltration, modification, or deletion.	Not mitigated. ADRs do not specify secure coding practices, parameterized queries, or SQL injection prevention.	Enforce parameterized queries (prepared statements) in all database interactions. Use ORM frameworks with built-in SQL injection protection. Implement least-privilege database user accounts (services should not use admin accounts). Deploy WAF with SQL injection detection rules. Conduct regular SAST/DAST scanning and penetrati	Medium. SQL injection is well-understood, and modern frameworks often prevent it by default. However, custom queries, legacy code, or developer mistakes still introduce vulnerabilities. OWASP Top 10 includes SQL injection due to continued real-world exploitation.	Critical. PostgreSQL stores customer PII (names, emails, payment details), booking history, and financial transactions. Exfiltration violates GDPR/DPDP, enables identity theft, financial fraud, and reputational damage. Data tampering could manipulate bookings, pricing, or payments.	Critical	Mitigation implemented. Additionally VPC network has clear subnet firewall separation to enforce separation of concerns.

						on testing.				
0006	S3 Data Lake	Misconfigured S3 bucket permissions allow public access to anonymized training datasets and telemetry archives	Information Disclosure	ADR-03, ADR-06, and ADR-08 specify S3 for storing telemetry archives, training datasets, and damage photos. Misconfigured bucket policies or ACLs could expose data publicly. Even anonymized data may be re-identifiable.	Not mitigated. ADRs do not mention S3 bucket policies, access logging, or public access blocks. AWS S3 defaults have improved, but misconfigurations remain common.	Enable S3 Block Public Access at account and bucket level. Use bucket policies with least privilege (IAM roles, not IAM users). Enable S3 access logging and CloudTrail for audit. Encrypt data at rest (S3-SSE or KMS). Implement lifecycle policies to auto-delete sensitive data per retention schedules. Use AWS Config rules to detect	Medium. S3 misconfigurations are a frequent cause of data breaches (e.g., Capital One, Facebook). AWS has improved defaults, but human error (overly permissive policies, public ACLs) persists. Automated scanners (Shodan, CloudSploit) detect exposed buckets.	High. S3 contains telemetry archives (GPS locations, customer behavior), training datasets (potentially re-identifiable), and damage photos (vehicle conditions). Exposure violates privacy regulations, enables competitive intelligence gathering, and may facilitate targeted attacks (stalking, theft).	High	Mitigation implemented. Additionally public access policy blocked at the org policy level. Have also use custom KMS to prevent PII disclosure.

						public buckets.				
0007	Conversational AI (MCP Integration)	User prompts sent to external LLMs (OpenAI, Anthropic, Gemini) include customer PII, violating GDPR and DPDP	Information Disclosure	ADR-13 describes conversational AI sending user queries to external LLM providers. User queries may contain PII (names, addresses, phone numbers, booking details). Sending PII to third-party LLMs without explicit consent violates GDPR Article 44 (international data transfers).	Partially mitigated. ADR-13 mentions "conversational memory limited to session scope for privacy" but does not address PII in prompts sent to external providers. ADR-14 covers data residency but does not explicitly restrict PII in LLM prompts.	Implement PII detection and redaction before sending prompts to external LLMs. Use named entity recognition (NER) models to identify and mask PII. Obtain explicit user consent for data processing by third-party AI providers. Use EU-hosted LLMs (OpenAI EU regions) for GDPR compliance. Audit and log all prompts sent to	High. Conversational AI systems commonly send user queries to external LLMs. Developers may not realize PII exposure risk. GDPR enforcement has increased (€20M fines), and regulators scrutinize third-party data transfers.	High. Sending PII to external LLMs violates GDPR Article 44 (international transfers) and DPDP Act (cross-border data flow). Regulatory fines (up to 4% annual revenue), reputational damage, and potential LLM provider data breaches (exposing customer PII) create significant risk.	High	Mitigation implemented.

						external providers. Consider fine-tuned on-premises LLMs for sensitive use cases.				
0008	Edge Damage Detection Model	Attacker deploys malicious OTA update to vehicle edge devices, disabling collision detection or injecting false damage classifications	Tampering	ADR-03 describes OTA updates for edge ML models. If update mechanism lacks integrity verification, attacker (malicious insider, compromised update server) could deploy malicious models that disable safety features or manipulate damage detection for fraud.	Partially mitigated. ADR-03 mentions "OTA updates enable rapid model reversion" but does not specify code signing, integrity verification, or secure boot.	Implement code signing for OTA updates (asymmetric cryptography). Vehicles verify update signatures before installation using embedded public keys. Use secure boot to prevent unauthorized firmware execution. Implement rollback protection (version pinning). Deploy canary	Low. Requires compromising OTA update infrastructure or insider access. Code signing and secure boot are standard best practices in automotive IoT. However, supply chain attacks (SolarWinds-style) increase risk.	Critical. Malicious OTA update could disable collision detection (safety risk, liability), manipulate damage classification (insurance fraud, customer disputes), or brick vehicles (service outage). Potential for mass-scale impact if deployed to entire fleet.	High	Mitigation implemented.

						rollouts with automated rollback on anomaly detection. Monitor edge model behavior for drift or malicious patterns.				
0009	Grafana Monitoring Dashboard	Unauthenticated or weak authentication on Grafana allows attacker to view sensitive operational metrics and business intelligence	Information Disclosure	ADR-10 and ADR-07 describe Grafana dashboards for monitoring. Dashboards may display real-time fleet locations, revenue metrics, customer activity, and AI model performance. Exposed dashboards leak competitive intelligence and operation	Not mitigated. ADRs do not specify Grafana authentication, RBAC, or network isolation. Default Grafana installations may have weak admin passwords or anonymous access enabled.	Enforce strong authentication (SSO with MFA, not default admin password). Implement RBAC to restrict dashboard access by role (operations staff, executives, engineers). Network isolate Grafana behind VPN or bastion host. Disable anonymo	Medium. Exposed Grafana dashboards are commonly discovered via Shodan scans. Default credentials (admin/admin) are often unchanged. However, modern deployments typically enforce authentication.	Medium. Leaked operational metrics (fleet locations, revenue, demand patterns) enable competitive intelligence gathering and targeted attacks (vehicle theft, surge pricing exploitation). Does not directly expose customer PII, but aids in planning	Medium	Mitigation implemented.

				al secrets.		us access. Use HTTPS with valid TLS certificates. Regularly audit user access and session logs.		broader attacks.		
0010	Airflow ML Training Pipeline	Attacker compromises Airflow scheduler or worker to poison ML training data, degrading model accuracy or injecting backdoors	Tampering	ADR-08 specifies Airflow for batch ML training pipelines (demand forecasting, predictive maintenance, vision models). Compromised Airflow deployment allows attacker to manipulate training data, inject poisoned samples, or steal model artifacts.	Not mitigated. ADR-08 does not mention Airflow authentication, DAG code review, or data integrity validation. Default Airflow installations may lack RBAC or secure secrets management.	Enable Airflow RBAC (role-based access control). Use Fernet encryption for secrets in metadata database. Validate training data integrity (checksums, anomaly detection). Implement DAG code review and approval workflows. Isolate	Medium. Airflow vulnerabilities (CVE-2020-11978 RCE) and misconfigurations are documented. Supply chain attacks on training data are emerging threats (e.g., poisoned datasets on public repositories).	High. Poisoned ML models produce incorrect predictions (demand forecasting errors, false maintenance alerts), causing operational disruptions, financial losses, and safety risks. Backdoored models could exfiltrate data or	High	Mitigation implemented.

						Airflow workers in dedicated VPC with no internet access. Monitor for unusual DAG executions or data access patterns. Use signed model artifacts.		provide attacker-controlled predictions. Model artifacts contain intellectual property.		
0011	Redis Cache	Unauthenticated Redis instance allows attacker to read cached customer sessions, payment tokens, and operational data	Information Disclosure	Redis is used for caching (ADR-04, ADR-06). Default Redis installations have no authentication. Cached data may include customer session tokens, payment authorization data, and operational state.	Not mitigated. ADRs do not mention Redis authentication, encryption, or network isolation. Default Redis configurations bind to 0.0.0.0 with no password.	Enable Redis authentication (requirepass). Use TLS for client connections. Network isolate Redis behind firewall (VPC with security groups). Implement short TTLs for sensitive cached data.	High. Unauthenticated Redis is a common misconfiguration, frequently exploited via internet-exposed instances (Shodan scans). Attackers use Redis as pivot point for lateral movement and data	High. Exposed Redis cache contains session tokens (account takeover), payment authorization data (fraud), and operational state (pricing, availability). Attacker could manipulate cached data to bypass	High	Mitigation implemented.

						Consider encrypting cached values at application layer. Use Redis Sentinel or Cluster for high availability with authentication.	exfiltration.	authorization, alter pricing, or cause service disruptions.		
0012	Payment Service	Attacker intercepts or replays payment authorization tokens to perform fraudulent transactions	Spoofing / Tampering	ADR-05 describes orchestrator for payment gateways. Payment authorization flows involve temporary tokens. Without proper anti-replay protections (nonces, short expiration), attacker could reuse intercepted tokens for fraud.	Partially mitigated. ADR-05 mentions orchestrator with retry and fallback logic, but does not specify anti-replay mechanisms or token validation. Payment gateway integration likely includes some protections, but application-layer validation is critical.	Implement nonce-based anti-replay (one-time tokens). Use short-lived payment authorization tokens (5-10 min expiration). Validate token binding (tie token to specific customer session, device fingerprint). Implement idempote	Low. Modern payment gateways (Stripe, Adyen) include anti-replay protections. However, application-layer vulnerabilities (poor token validation, long expiration) could still enable attacks. Requires attacker to intercept tokens	Critical. Successful payment fraud directly causes financial loss to company or customers. Violates PCI-DSS compliance, leading to fines and payment processor termination. Reputational damage and customer trust	High	Mitigation implemented.

						ncy keys for payment requests. Monitor for duplicate transaction attempts. Use PCI-DSS compliant payment tokenization (avoid storing raw card data).	via MITM or XSS.	erosion. Chargebacks and dispute resolution costs.		
0013	TimescaleDB (Telemetry Database)	Attacker exploits database vulnerability or compromised credentials to exfiltrate historical telemetry data (GPS, customer locations)	Information Disclosure	ADR-03 and ADR-06 specify TimescaleDB for storing telemetry (GPS, battery). Database contains 90 days of hot data with 2-year cold storage (S3). Exposed database allows exfiltration of customer	Not mitigated. ADRs do not specify database authentication, network isolation, or encryption at rest. TimescaleDB is PostgreSQL-based, inheriting PostgreSQL security considerations.	Use strong database credentials (complex passwords, not default). Implement network isolation (VPC, security groups, no public internet access). Enable encryption at rest (LUKS, AWS EBS encryption). Use	Medium. Database breaches via SQL injection, weak credentials, or unpatched vulnerabilities are common. However, network isolation and modern cloud security reduce exposure. Insider threat (malicious	High. Telemetry database contains 90-day history of customer GPS locations (stalking, privacy violations), vehicle usage patterns (competitive intelligence), and operational state (fleet availability). Violates GDPR	High	Mitigation implemented.

				location history.		least-privilege database users (read-only for analytics, write-only for ingestion). Enable database audit logging. Implement column-level encryption for sensitive GPS coordinates. Regular vulnerability scanning and patching.	employee) increases likelihood.	Article 5 (data minimization, purpose limitation). Enables targeted physical attacks.		
0014	External Weather/ Events APIs	Compromised or malicious external API (OpenWeatherMap, PredictHQ) injects false data to manipulate demand	Tampering	ADR-04 integrates external APIs for weather and events data. Demand forecasting models (ADR-02) rely on this data. If API is	Partially mitigated. ADR-04 mentions caching and fallback logic, but does not specify data validation or integrity checks.	Validate external API responses (schema validation, range checks, anomaly detection). Cross-reference multiple data	Low. Requires compromising external API provider or MITM attack on API communication. Major providers (OpenWe	Medium. False weather/ event data causes incorrect demand forecasts, leading to suboptimal pricing (revenue loss or customer	Medium	Mitigation implemented via cross reference & signed response.

		forecasting and dynamic pricing		compromised or provides malicious data, ML models produce incorrect predictions, causing operational and financial impact.	Circuit breaker pattern mitigates availability issues but not data tampering.	sources (primary and fallback providers). Implement outlier detection in ML pipelines to flag suspicious data. Use signed API responses if available. Monitor API response patterns for sudden changes. Cache known-good data as fallback.	atherMap) have strong security. However, supply chain attacks are increasing (e.g., npm, SolarWinds).	dissatisfaction), poor fleet positioning (unavailability), and incorrect relocation incentives. Financial impact is indirect and bounded by pricing caps (2-2.5x surge).		
0015	Operations Dashboard	Insufficient authorization checks allow staff with limited privileges to access or modify critical fleet	Elevation of Privilege	ADR-01 describes Operations Dashboard for staff. Without granular RBAC, low-privilege staff	Not mitigated. ADRs do not specify RBAC implementation, authorization checks, or audit logging	Implement fine-grained RBAC (role-based access control) with least privilege. Define roles: Field	Medium. Insufficient authorization is a common vulnerability (OWASP A01:2021 Broken Access Control).	High. Unauthorized staff could manipulate fleet operations (disable vehicles, alter task assignments),	High	Mitigation implemented. Additionally strict enforcement of JWT tokens using auth middleware plugins

		operations data		(e.g., field technicians) could access sensitive data (revenue, customer details) or perform unauthorized actions (disable vehicles, modify pricing).	for staff actions. Microservices architecture (ADR-01) requires consistent authorization enforcement across services.	Technician (limited to task execution), Operations Manager (fleet management), Admin (full access). Enforce authorization at API Gateway and service layer. Audit all staff actions with immutable logs. Implement MFA for privileged operations. Regular access reviews and deprovisioning.	Microservices architectures increase risk due to distributed authorization logic. Insider threat (malicious or negligent staff) elevates likelihood.	access customer PII (privacy violations), modify pricing (revenue loss or fraud), or sabotage systems (service outage). Insider threats are difficult to detect and cause significant damage.		added to the gateway itself preventing unauthenticated endpoint exposure without proper procedure
0016	Kafka Event Bus	Attacker injects malicious events into Kafka	Tampering	ADR-06 describes event-driven architecture with	Partially mitigated. ADR-06 mentions Avro schemas	Implement producer authentication (SASL/SC	Medium. Requires compromising a microservice or	High. Injected events could bypass business	High	Mitigations Implemented

		topics to trigger unintended system behavior or bypass business logic		Kafka. If event producers are not authenticated or messages lack integrity verification, attacker could inject false events (e.g., fake bookings, fraudulent payment completions, manipulated telemetry).	with Schema Registry for schema evolution, which provides some validation. However, does not mention message signing, producer authentication, or integrity verification.	RAM or mTLS). Use message signing (HMAC or digital signatures) to verify event integrity. Validate events against business rules before processing (e.g., booking must reference existing customer and vehicle). Enable Kafka ACLs to restrict write access per topic. Monitor for anomalous event patterns (rate limiting, outlier detection).	event producer. Kafka without authentication allows any client to publish events. Insider threat or compromised service account increases likelihood.	logic (fraudulent bookings without payment, fake vehicle relocations, manipulated pricing updates), cause operational disruptions (false collision alerts, bogus maintenance tasks), or exfiltrate data (trigger unauthorized data exports).		

0017	Vehicle NFC/Bluetooth Unlock	Attacker uses relay attack or cloned NFC/Bluetooth credentials to unlock vehicles without authorization	Spoofing	ADR-03 describes NFC tap or QR scan for vehicle unlock. NFC and Bluetooth Low Energy (BLE) are vulnerable to relay attacks (attacker proxies communication between legitimate customer phone and vehicle) or credential cloning.	Partially mitigated. ADR-03 does not specify relay attack protections or distance bounding. NFC/BLE unlock mechanisms may include basic authentication but often lack anti-relay measures.	Implement distance bounding protocols (measure round-trip time to detect relay attacks). Use challenge-response authentication (dynamic tokens, not static credentials). Implement geofencing (verify customer is at vehicle location via GPS before unlock). Enable BLE pairing with out-of-band confirmation (e.g., display code on vehicle screen). Monitor	Low. NFC relay attacks are technically feasible but require proximity to both customer phone and vehicle. BLE relay attacks are more complex. Real-world exploits primarily target luxury cars and high-value assets. MobilityCorp vehicles (bikes, scooters) are lower-value targets, reducing attacker motivation.	Medium. Unauthorized vehicle unlocks enable theft (bikes, scooters, even cars/vans), joyriding, vandalism, or using vehicles without payment. Financial loss from stolen vehicles, insurance claims, and customer dissatisfaction. Does not directly expose customer data.	Medium	Have applied mitigations (addressed with hardware security & firmware updates)

						for unusual unlock patterns (multiple rapid attempts, distant locations).				
0018	OpenTelemetry Logging	Distributed tracing logs inadvertently capture sensitive customer data (PII, payment details), exposing it to monitoring infrastructure	Information Disclosure	ADR-07 specifies OpenTelemetry for tracing and logging. Distributed tracing spans may capture request payloads, headers, and database queries containing PII or payment data. Logs stored in centralized monitoring systems may be accessible to wide audience.	Partially mitigated. ADR-07 mentions OpenTelemetry adoption but does not address log sanitization or sensitive data redaction. Document notes "Logs may contain sensitive data (requires sanitization)" in DF14 but no implementation details.	Implement automatic PII redaction in tracing instrumentation (mask credit card numbers, names, emails). Configure OpenTelemetry SDK with data sanitization hooks. Use structured logging with explicit allow-lists (log only necessary fields). Enable RBAC on monitoring	High. Logging sensitive data is a common developer mistake, especially in distributed tracing where entire request contexts are captured. Centralized log aggregation increases exposure surface. Compliance audits (GDPR, PCI-DSS) frequently identify unsanitized logs.	Medium. Exposed logs reveal customer PII (names, emails, locations), payment details (card numbers, authorization tokens), and business logic (pricing algorithms, operational patterns). Violates GDPR/PCI-DSS, leading to regulatory fines. However, requires attacker to	Medium	Mitigations applied

						systems (Grafana, VictoriaMetrics) to restrict log access. Encrypt logs at rest and in transit. Regular log audits for sensitive data leakage.		compromise monitoring infrastructure or insider access.		
0019	Model Registry (MLflow)	Unauthorized access to MLflow model registry allows attacker to steal proprietary ML models or deploy malicious models	Tampering / Information Disclosure	ADR-08 and ADR-12 describe MLflow for model versioning and registry. Model registry contains trained model artifacts (demand forecasting, predictive maintenance, vision models), which represent intellectual property.	Not mitigated. ADRs do not mention MLflow authentication, access control, or artifact signing. Default MLflow deployments may lack authentication or use weak credentials.	Enable MLflow authentication (built-in auth or SSO integration). Implement RBAC (data scientists can view, only ML engineers can deploy to production). Sign model artifacts (digital signatures) to verify integrity. Network	Medium. MLflow security is often overlooked in rapid ML development. Exposed MLflow servers are discoverable via internet scans. Insider threat (disgruntled data scientist) increases likelihood.	High. Stolen ML models represent intellectual property loss (competitive disadvantage). Deployed malicious models cause operational disruptions, financial losses, and safety risks (poisoned demand	High	Mitigations applied

				Unauthorized access allows model theft or poisoning.		isolate MLflow server behind VPN or bastion host. Encrypt model artifacts at rest. Audit all model deployments and access logs.		forecasting, manipulated maintenance predictions). Model theft enables adversarial attacks (reverse engineering, evasion).		
0020	Dynamic Pricing Engine	Attacker manipulates demand forecast inputs (weather, events) to artificially inflate or deflate prices for financial gain	Tampering	ADR-02 describes AI-driven dynamic pricing based on demand forecasts. Attacker who can manipulate forecast inputs (weather API data, event data, or telemetry) could cause price manipulation for personal benefit (e.g., deflate	Partially mitigated. ADR-04 describes caching and fallback logic for external APIs, but does not prevent tampering of internal telemetry or forecast inputs. Price caps (2-2.5x surge) limit financial impact but do not	Validate all forecast inputs (schema validation, range checks, cross-referencing multiple sources). Implement anomaly detection in pricing engine (flag sudden price changes, unusual demand patterns). Use tamper-	Low. Requires compromising multiple input sources (weather API, telemetry, event data) or internal forecasting service. Price caps (2-2.5x) limit manipulation range. Detection via monitoring reduces success likelihood.	Medium. Successful price manipulation causes revenue loss (artificially low prices) or customer dissatisfaction (excessive surge pricing). Price caps limit maximum impact per transaction. Widespread manipulation could	Medium	Mitigations applied

				prices for self, inflate prices to harm competitors or customers).	prevent manipulation.	evident logging for pricing decisions (audit trail). Enforce separation of duties (different teams manage forecasting vs. pricing). Monitor for correlated patterns (same user benefiting from price drops, geographic clustering of anomalies). Implement pricing approval workflows for extreme values.	However, insider threat (malicious data scientist) increases risk.	cause significant aggregate losses. Reputational damage if customers detect unfair pricing. Regulatory scrutiny for discriminatory pricing.		
0021	Temporal Workflow Engine	Compromised Temporal workers	Tampering	ADR-08 specifies Temporal for event-	Not mitigated. ADR-08 does not	Implement Temporal namespa	Medium. Temporal security depends	High. Malicious workflows could	High	Mitigations implemented

		execute malicious workflows, triggering unauthorized retraining of ML models or operational disruptions		driven workflows (retraining triggers, real-time inference orchestration). Compromised Temporal workers could execute malicious workflows to poison models, exfiltrate data, or disrupt operations.	mention Temporal authentication, workflow validation, or worker isolation. Temporal workflows are code that runs with service account permissions, creating execution risks.	ce isolation with authentication. Use mTLS for worker-to-server communication. Validate workflow definitions before execution (code review, static analysis). Run workers in isolated environments (containers, sandboxes) with least-privilege service accounts. Monitor workflow execution patterns for anomalies. Implement workflow approval	on workflow code quality and deployment security. Compromising CI/CD pipeline or worker infrastructure allows malicious workflow injection. Insider threat (malicious engineer) elevated risk.	trigger unauthorized model retraining (data poisoning), exfiltrate training data or model artifacts, manipulate operational workflows (fake maintenance alerts, fraudulent relocations), or cause service outages (resource exhaustion, infinite loops). Wide blast radius due to Temporal's orchestration role.		

						gates for critical operations (production model deployment). Enable workflow versioning and rollback.				
0022	Customer Mobile App	Reverse engineering of mobile app exposes API endpoints, authentication flows, and business logic for exploitation	Information Disclosure	Customer mobile app (React Native) communicates with backend APIs. Reverse engineering APK/IPA files can reveal API endpoints, authentication mechanisms, encryption keys, and business logic, enabling targeted attacks.	Not mitigated. ADRs do not mention mobile app hardening, obfuscation, or certificate pinning. React Native apps are particularly vulnerable to reverse engineering (JavaScript bundle extraction).	Implement mobile app obfuscation (ProGuard for Android, symbol stripping for iOS). Use certificate pinning to prevent MITM attacks. Store sensitive keys in platform secure storage (Android Keystore, iOS Keychain). Implement root/jailbr	High. Mobile app reverse engineering is straightforward with publicly available tools (apktool, Frida, Hopper). React Native apps are easier to reverse than native apps. Attackers routinely reverse engineer apps to find vulnerabilities.	Medium. Exposed API endpoints enable automated scraping, abuse, and targeted attacks. Revealed authentication flows may expose weaknesses. Extracted business logic (pricing algorithms, incentive calculations) provides competitive	Medium	Mitigations implemented

						eak detection. Use runtime application self-protection (RASP). Avoid hardcoding secrets in app code. Implement server-side business logic validation (never trust client). Regular security assessments and penetration testing of mobile apps.		intelligence. However, backend validation and rate limiting mitigate impact. Does not directly expose customer data unless combined with other vulnerabilities.		
0023	Vehicle Collision Detection	False positive collision alerts due to edge model errors or adversarial inputs cause operational disruptions and customer	Denial of Service	ADR-03 describes edge-based collision detection using IMU sensors and ML model. False positives (incorrectly flagged	Partially mitigated. ADR-03 includes rule-based fallback (g-force threshold) and specifies 99.5% recall, 90% precision	Implement sensor fusion (combine IMU, wheel slip, ABS, camera) for robust detection. Use ensemble models (multiple algorithm	Medium. Edge ML models will inevitably produce false positives (90% precision means 10% false positive rate). Adversari	Medium. False positives cause operational disruption (unnecessary staff dispatch, vehicle downtime), customer frustratio	Medium	Mitigations implemented

		dissatisfaction		collisions) trigger emergency responses, disable vehicles, and alert operations team, causing disruptions.	targets. However, adversarial inputs (intentional sensor manipulation) are not addressed.	s voting) to reduce false positives. Deploy anomaly detection to identify adversarial inputs (sensor spoofing, vibration attacks). Implement confidence thresholds with graduated response (low confidence → log only, high confidence → emergency action). Regular model retraining with real-world false positive data. A/B test model updates in shadow	al attacks (intentionally triggering sensors via bumps, vibrations) require physical access but are feasible. High traffic volume increases absolute false positive count.	n (service interruption), and increased costs (wasted manual interventions). Does not directly cause safety risk or data exposure. Multiple false positives could train customers to ignore alerts (cry wolf effect), reducing safety efficacy.		

						mode before production deployment.				
0024	Data Residency Compliance	Accidental cross-region data transfer (PII from EU to US) violates GDPR, causing regulatory fines and legal liability	Compliance Violation	ADR-14 and ADR-09 establish strict data residency requirements (EU PII must stay in eu-west-1, no cross-border transfer). Misconfigured replication, backup, or analytics pipelines could transfer PII across regions.	Partially mitigated. ADR-14 defines data classification and residency policy (PII restricted to origin region). However, implementation details (technical controls, automated validation) are not specified. Human error in configuration remains risk.	Implement technical controls for data residency: network-level restrictions (VPC peering only within region), encryption key management per region (separate KMS keys), automated data classification tagging. Deploy data loss prevention (DLP) tools to detect cross-region PII transfers. Impleme	Medium. Data residency violations are common in complex multi-region architectures due to misconfiguration, inadequate testing, or developer errors. Cloud service defaults (global replication, cross-region backup) increase risk. GDPR enforcement has increased (€20M+ fines).	Critical. GDPR violations result in regulatory fines (up to 4% annual global revenue or €20M, whichever higher), mandatory breach notifications, and reputational damage. Legal liability for customer harm. Potential for individual lawsuits (GDPR Article 82). Loss of customer trust in EU market. Business continuity	Critical	Mitigations implemented

						nt pre-deployment validation (infrastructure-as-code checks, policy-as-code). Regular compliance audits and data flow mapping. Enable AWS SCPs (Service Control Policies) or Azure Policies to block cross-region resource creation.		risk if data processing banned.		
0025	Predictive Maintenance Service	Incorrect failure predictions (false negatives) cause unexpected vehicle breakdowns, safety incidents, and customer harm	Safety Risk	ADR-03 describes predictive maintenance using telemetry data (battery voltage, vibration) to forecast failures 7-14 days in advance.	Partially mitigated. ADR-03 specifies 85% true positive rate target, acknowledging 15% miss rate. Model monitoring for drift and	Implement conservative prediction thresholds (prioritize recall over precision). Use ensemble models (multiple	Medium. ML models will inevitably have false negatives (85% true positive = 15% miss rate). Model performance degrades	High. Unexpected vehicle breakdowns during customer use cause safety incidents (mid-trip failures on roads), customer	High	Mitigations implemented

				False negatives (missed failures) result in unexpected breakdowns during customer use.	retraining on real-world data reduces risk over time. However, initial deployment and edge cases remain vulnerable.	algorithms, voting) for robust predictions. Combine predictive maintenance with scheduled preventive maintenance (defense in depth). Enable fallback to rule-based alerts (absolute thresholds: battery voltage <X, vibration >Y). Monitor prediction accuracy continuously and trigger emergency retraining on accuracy drops. Implement rapid manual	with drift (new vehicle types, environmental changes). Real-world complexity (unusual usage patterns, sensor failures) increases miss rate.	harm (injury risk, stranded users), service disruptions (towing, refunds), and reputational damage. Liability for injuries or accidents caused by maintenance failures. Regulatory scrutiny if pattern of safety incidents emerges.			

						intervention for high-risk vehicles (age, usage patterns).				
0026	Feature Store (Feast/Tecton)	Unauthorized modification of feature store data corrupts ML model inputs, causing prediction failures and operational disruptions	Tampering	Architecture diagram describes Feature Store (Feast/Tecton) ingesting data from Kafka and providing features to ML models. Compromised feature store allows attacker to poison model inputs in real-time, affecting all models simultaneously.	Not mitigated. Architecture mentions Feature Store but ADRs do not specify implementation, authentication, or integrity validation. Feature stores are often overlooked in security reviews.	Implement authentication and RBAC for feature store access. Use immutable feature versioning (write-once, append-only). Validate feature values against expected distributions (anomaly detection). Sign feature batches (cryptographic hashes) to detect tampering. Implement feature store audit	Low. Requires compromising feature store infrastructure or CI/CD pipeline. Feature stores are internal systems, reducing exposure. However, insider threat or lateral movement after initial compromise increases risk.	High. Poisoned features affect all ML models simultaneously (demand forecasting, predictive maintenance, pricing), causing widespread prediction failures. Operational disruptions include incorrect pricing, false maintenance alerts, and poor fleet positioning. Financial losses and safety	Medium	Mitigations implemented

						logging. Network isolate feature store behind firewall. Monitor for unusual feature update patterns or values.		risks. Difficult to detect as features appear legitimate.		
0027	Vision AI Service (Damage Detection)	Privacy-preserving design fails, and raw vehicle/customer photos are inadvertently uploaded to cloud, violating GDPR	Information Disclosure	ADR-03 describes privacy-preserving vision AI with edge processing (blurred thumbnails, embeddings uploaded, not raw images). Implementation errors or fallback logic could bypass privacy protections and upload raw photos containing faces,	Partially mitigated. ADR-03 specifies privacy-first design (edge processing, blurred uploads, 7-day local retention). However, implementation bugs, error handling, or user opt-in for dispute resolution could leak raw images.	Implement strict upload policies enforced at edge (allow-list for metadata/embeddings, deny raw images by default). Use content-aware validation (detect face/license plate in images before upload, reject if found). Encrypt local image storage	Medium. Implementation bugs in edge devices or upload logic could bypass privacy protections. Error conditions (low storage, connectivity issues) may trigger fallback to cloud processing with raw images. User disputes requiring raw	High. Raw image uploads violate GDPR Article 5 (data minimization) and Article 9 (biometric data). Photos may contain faces (biometric data), license plates (personal data), or sensitive location context (home addresses). Regulatory fines,	High	Mitigations implemented

				license plates, or sensitive backgrounds.		with automatic deletion after 7 days. Audit all image uploads for compliance. Implement differential privacy techniques for embeddings. Regular penetration testing of image upload pipeline. Enable user consent flows with explicit warnings for raw image uploads (dispute resolution only).	images increase upload frequency.	reputational damage, and customer trust erosion. Images stored in cloud are vulnerable to breaches.		
0028	CI/CD Pipeline	Compromised CI/CD pipeline injects malicious code into	Tampering	Microservices architecture (ADR-01) and ML pipelines	Not mitigated. ADRs do not mention CI/CD security,	Implement pipeline hardening: use dedicated service	Medium. CI/CD compromise is an emerging attack vector	Critical. Compromised CI/CD allows attacker to inject	Critical	Mitigations implemented

		microservices or ML models, affecting production systems		(ADR-08) require CI/CD for deployment. Compromised pipeline (vulnerable Jenkins/GitLab, stolen credentials, supply chain attack) allows attacker to inject backdoors, steal secrets, or deploy malicious code.	pipeline hardening, or deployment controls. CI/CD is critical infrastructure but often under-secured.	accounts with least privilege, enable MFA for pipeline access, sign commits and artifacts (GPG), scan code for vulnerabilities (SAST/DAST), implement approval gates for production deployments. Use immutable build environments (containers, ephemeral runners). Enable pipeline audit logging. Store secrets in dedicated vaults (not environment	(SolarWinds, Codecov). Vulnerabilities in pipeline tools (Jenkins CVEs) and stolen credentials (phishing, credential stuffing) are common. Supply chain attacks (malicious npm packages) increasing.	code into all microservices and ML models, creating persistent backdoors. Full system compromise: data exfiltration, service disruption, malware deployment. Difficult to detect (appears as legitimate deployment). Wide blast radius affecting all customers. Incident response and remediation extremely costly.		

						variables). Implement supply chain security (verify dependency integrity, use private registries, scan for known vulnerabilities). Regular pipeline security audits.				
0029	VictoriaMetrics (Metrics Database)	Metrics database exposed without authentication allows attacker to exfiltrate operational intelligence and manipulate monitoring	Information Disclosure / Tampering	ADR-10 specifies VictoriaMetrics for metrics storage. Exposed metrics database reveals operational patterns (traffic volumes, error rates, resource usage), business intelligence (revenue, usage	Not mitigated. ADR-10 does not mention VictoriaMetrics authentication, network isolation, or access control. Default VictoriaMetrics installations may lack authentication.	Enable VictoriaMetrics authentication (basic auth or external auth proxy). Network isolate behind firewall (VPC, security groups). Use HTTPS for client connections. Implement RBAC	Medium. Exposed time-series databases are discoverable via internet scans. VictoriaMetrics security features are less mature than enterprise solutions. Default configurations prioritize	Medium. Leaked metrics provide competitive intelligence (fleet size, revenue trends, customer activity patterns), operational intelligence (system vulnerabilities, error patterns), and	Medium	Mitigations implemented

				trends), and system vulnerabilities (unpatched services, misconfigurations).		via reverse proxy (query-level access control). Sanitize metric labels to avoid leaking sensitive data (customer IDs, PII). Enable audit logging for query access. Regular security assessments of monitoring infrastructure.	ease of use over security.	attack planning data (traffic patterns for DDoS, rate limits for abuse). Metric manipulation could blind monitoring (hide attacks, trigger false alerts). Does not directly expose customer PII.		
0030	Schema Registry (Avro)	Unauthorized schema modifications in Kafka Schema Registry break event processing and cause service outages	Denial of Service	ADR-06 describes Avro schemas with Schema Registry for Kafka event validation. Unauthorized schema changes	Partially mitigated. ADR-06 mentions Avro schemas and Schema Registry for schema evolution, providing basic validation	Enable Schema Registry authentication and authorization. Implement schema compatibility checks (backward,	Low. Requires compromising Schema Registry or engineer credentials. Schema Registry is internal infrastructure with	Medium. Incompatible schema changes break event processing across all consumers (booking service, telemetry	Medium	Mitigations implemented

				(incompatible modifications, deletions) break event consumers, causing processing failures and service outages.	. However, does not specify access control or change approval workflows.	forward, full compatibility). Use schema versioning with immutable history. Implement change approval workflows (peer review, testing in staging). Network isolate Schema Registry behind firewall. Enable audit logging for schema changes. Deploy pre-production schema validation (test consumers with new schemas). Monitor for schema evolution errors in	limited exposure. However, accidental incompatible changes by legitimate developers are more likely than malicious attacks.	ingestion, pricing engine), causing widespread service outages. Data loss if events cannot be parsed. Operational disruption until schemas rolled back. Does not directly expose data or cause financial fraud.		

						production.				

```

```