Rethinking “Offline ERP” in Mining: A Case for True Offline-First Systems

Author

Harsh Sharma Independent Researcher, IT Systems & ERP Integration

GitHub: connect

LinkedIn: connect

Research Focus: Offline-first architectures, ERP-OT integration, mining communication systems, and industrial IoT resilience.
Contribution: Concept design, architecture development, integration strategy for mining ERP, and manuscript preparation.

Abstract

In most mining IT literature, “offline ERP” is treated as simply hosting the ERP server on-site, disconnected from the public internet. While this avoids reliance on cloud connectivity, it still requires a functioning local network at all times. In underground mining, where communication links can fail unpredictably, this approach still causes downtime and data loss.

In my work, I’ve been exploring a different angle — a *true offline-first ERP* where data is stored right on the device in the field. Instead of stopping when the network goes down, these devices keep logging operations, safety checks, and production data. When any connection becomes available — whether through Wi-Fi at the surface, a leaky feeder cable, or a mesh link — the data syncs automatically to a central server and ERP. This paper outlines the communication environment in mines, explains why existing ERP models fall short, and proposes a device-level offline-first architecture built using PouchDB and CouchDB.

---

1. Introduction

Anyone who’s worked with IT systems in a mining context knows that connectivity is fragile [1]. Rock faces, dust, and equipment constantly get in the way of radio signals. Underground, miners rely on systems like leaky feeders [4], mesh networks [1], and RFID checkpoints [1] just to keep basic communication alive.

In ERP research, “offline” usually just means “on-premise” — meaning the server is in the mine’s own server room instead of in the cloud [2]. This is better than needing the internet all the time, but there’s a catch: if the local LAN goes down, you can’t use the ERP at all. There’s no buffering of data, no way to capture information while disconnected.

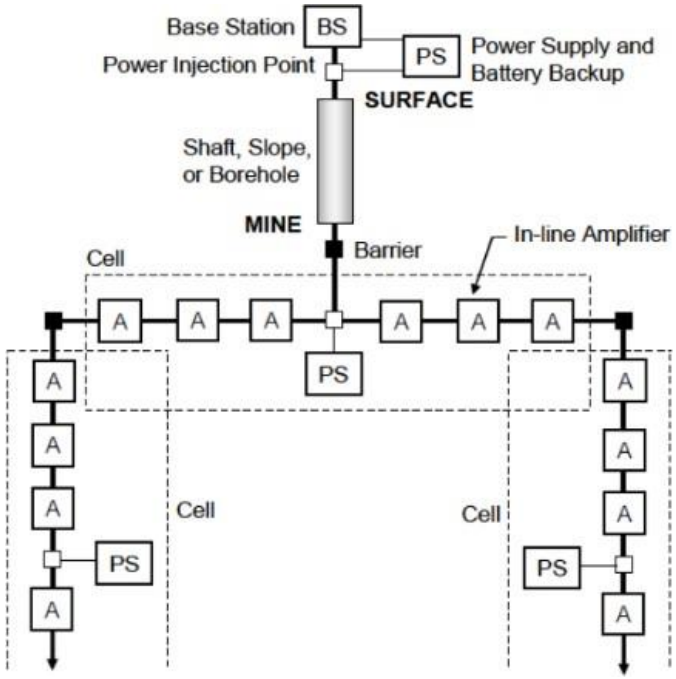
What I’m proposing is different. Imagine if every tablet, scanner, or sensor could keep working without any network — storing its own data — and then, whenever it got even a brief connection, quietly sync with the central ERP. That’s the gap I think we can fill.

---

2. A Quick Look at Underground Communication Systems

2.1 Leaky Feeder Cables

A leaky feeder is basically a coaxial cable with intentional gaps in its shielding so radio signals can “leak” in and out along its length. It acts like a very long antenna. Amplifiers are placed every few hundred meters to keep the signal strong. They’re reliable for voice and basic data, but not for heavy data like high-res images.



The image shows a Basic Leaky Feeder Layout for underground mine communications. This is a commonly used system that enables radio communications in environments where normal radio signals cannot travel due to rock, depth, or interference. Let's break down the components and terms used in the diagram (based on Novak, 2010):

BS – Base Station Located at the surface, it's the main communication hub that connects the underground network to the external world.

Typically includes radios, servers, and routing equipment to handle data and voice.

PS – Power Supply Provides electrical power to different parts of the leaky feeder system. May include battery backup in case of power failure. You'll notice multiple PS units underground — these ensure the system keeps working throughout the network.

Power Injection Point – where power is injected into the coaxial cable line to run the in-line amplifiers and other electronic components. Typically co-located with the Base Station.

Shaft, Slope, or Borehole – This is the vertical or inclined tunnel that connects the surface with the underground mine.

The leaky feeder cable runs through this structure to extend communication underground.

Barrier – An intrinsically safe barrier or protection device that ensures electrical equipment used underground does not cause sparks or explosions (important for explosive atmospheres like coal mines).

Separates safe (surface) and hazardous (mine) areas electrically.

A – In-Line Amplifier These boost the signal strength along the leaky feeder cable to compensate for losses over distance.

Ensures clear and continuous communication over long stretches.

They are placed periodically (in both horizontal and vertical directions) in the system.

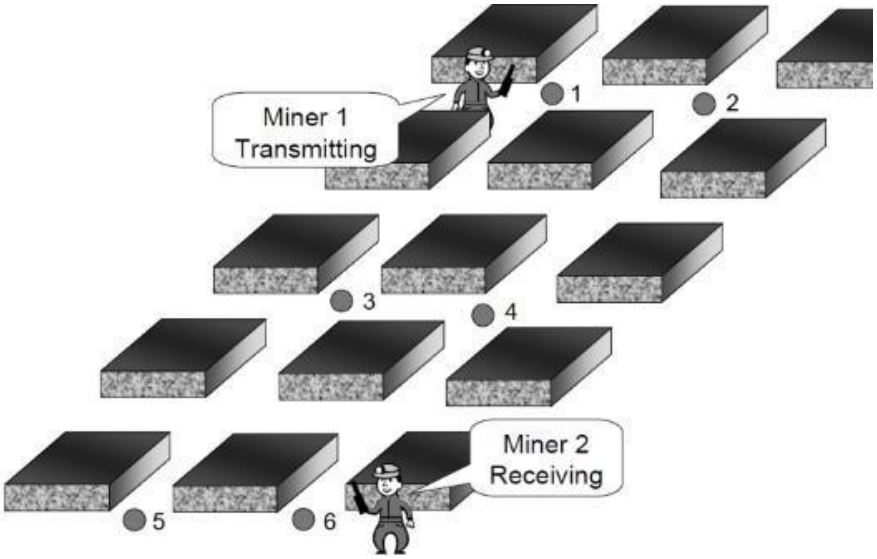
Leaky Feeder Cable – A coaxial cable that is specially designed to "leak" radio signals out and let them back in. Acts like a long antenna – enabling radios to communicate with each other underground. Provides radio coverage in tunnels, even where regular radios would fail.

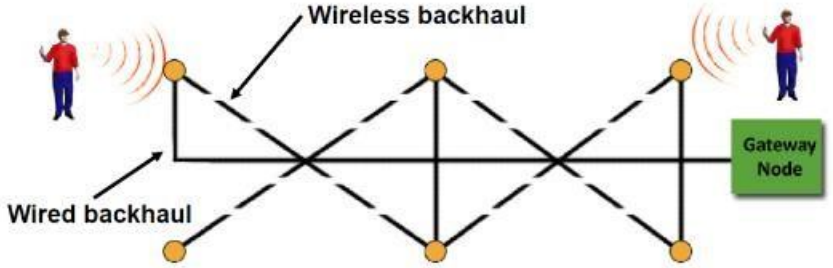
Cell – A zone or segment in the mine that is covered by a certain portion of the leaky feeder system. Defined by signal coverage, amplifier spacing, and network topology. Each cell is essentially a communication coverage area.

Inline T-Junctions – The branching points (black squares) where feeder cables split off to cover different areas. Allow the system to be modular and expand into other shafts or rooms.

2.2 Mesh Networks

In a mesh, each node can talk to its neighbors, passing data along until it reaches the destination. If one node fails, traffic takes another route. This is handy in a mine because layouts change and equipment moves. The downside is that each "hop" adds delay and cuts bandwidth, so big data transfers slow down.





2.3 RFID Tracking

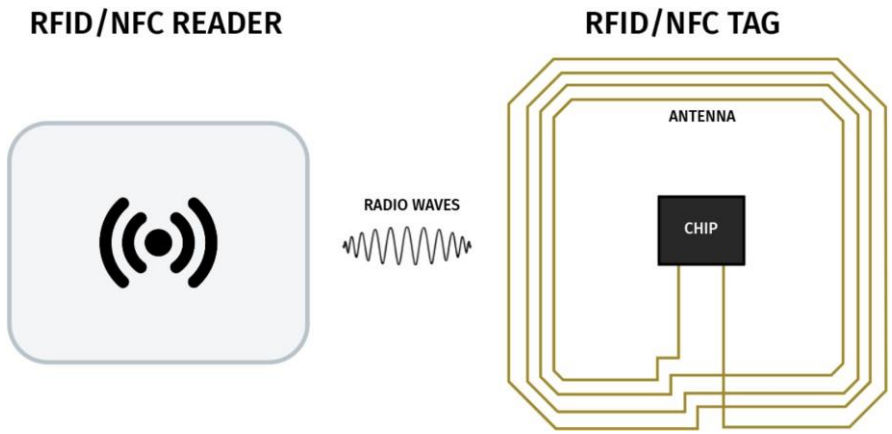
RFID tags — passive or active — are used to track people and equipment. Passive tags get their power from the reader's signal and work only at close range. Active tags have their own battery and can transmit further. This helps track who went where and when, but still depends on readers being placed in the right spots.

A Reader-Tag system typically includes:

- Tags: Small electronic devices worn by miners or attached to equipment. Each tag has a unique ID.
- Readers: Devices installed at intervals along the leaky feeder cable or at strategic points (e.g., entries, intersections) that detect nearby tags and send data to the control center via the leaky feeder system.
- Tracking Software: Software on the surface receives signals from the readers and determines the real-time location of personnel/equipment.

How It Works (Step-by-Step) - Miner wears a tag (often on their helmet or belt). - As they move underground, they pass by a reader installed near the leaky feeder cable. - The reader detects the tag's signal and logs: - Tag ID - Time - Reader location

This data is sent to the surface control center via the leaky feeder network. The control software updates the miner's current location on the map.



---

3. Why Current ERP Models Struggle

Feature	Cloud ERP	On-Prem ERP (called "offline" in research)	True Offline-First ERP
Needs internet	Yes	No, but needs local network	No

Feature	Cloud ERP	On-Prem ERP (called “offline” in research)	True Offline-First ERP
Works during LAN loss	No	No	Yes
Sync after downtime	No	No	Yes
Data storage	Cloud	Central server	On device

From my reading and my own reasoning, the biggest flaw is that **on-prem ERP is still network-bound**. Once the LAN link to the server is broken, all the field devices are useless until it comes back.

---

4. The Architecture I’m Proposing

- **Local Database on Device** — Every field device (tablet, handheld RFID reader, etc.) runs a PouchDB instance.
- **Central CouchDB** — Lives on-site or in the cloud; acts as the sync target.
- **Sync Layer** — As soon as any connection is detected (Wi-Fi, fibre, mesh, leaky feeder), the device sends its local changes and pulls down updates.

This means data collection never stops. If a miner does an equipment check deep underground, it’s stored locally. Hours later, when they pass through a zone with coverage, it syncs — no manual steps needed.

Detailed

The proposed architecture is designed to **ensure uninterrupted data capture** in underground mining environments, where connectivity is unreliable. It enables **device-level storage**, **opportunistic synchronization**, and **automatic integration** with existing enterprise systems.

---

1. Local Database on Device (Edge Layer)

Purpose: Ensure that field operations continue without interruption during network outages.

- **Components:**
  - Rugged tablets
  - Handheld RFID readers
  - IoT-enabled inspection tools and sensors
- **Technology:**
  - **PouchDB** running directly on each device.
- **Functionality:**
  - Captures operational data (e.g., inspection logs, production counts, safety checks) in real time.
  - Stores data locally as JSON documents.
  - Assigns each record a **UUID**, timestamp, and device ID to support later conflict resolution.
  - Allows full CRUD operations offline.
- **Benefits:**
  - Eliminates downtime during LAN or WAN failures.
  - Avoids data loss by persisting records locally until sync.

---

2. Sync Layer (Opportunistic Connectivity Layer)

Purpose: Transfer stored data to a central system whenever a connection is available.

- **Triggers:**
  - **Wi-Fi** (surface or underground hotspots)
  - **Fibre access points** in key locations
  - **Mesh network links** through mobile nodes
  - **Leaky feeder systems** for radio/modem-based data
- **Mechanism:**
  - PouchDB → CouchDB sync via **CouchDB replication protocol**.
  - Sync is **bidirectional**:
    - Pushes local changes to the central database.
    - Pulls latest updates from other devices/systems.
- **Optimizations:**
  - **Incremental replication** — sends only changes since last sync.
  - **Compression** to handle low-bandwidth links.

- **Priority rules** (e.g., safety-related data syncs first).

---

3. Central CouchDB (Surface or Cloud Layer)

Purpose: Act as the **sync target** and central store for all field-collected data.

- **Deployment Options:**
  - **On-premise** (for regulatory/security constraints)
  - **Cloud-hosted** (for central corporate visibility)
- **Functionality:**
  - Receives data from multiple field devices.
  - Stores historical revisions of documents to prevent overwrite conflicts.
  - Supports replication to backup systems.

---

4. Middleware & Integration Layer (Silo Bridge)

Purpose: Distribute synced data into existing mining enterprise systems without replacing them.

- **Responsibilities:**
  1. **Data Transformation** — Map JSON fields from PouchDB to ERP, SCADA, Safety, and Geology DB formats.
  2. **Routing** — Send data to appropriate system endpoints.
  3. **Validation** — Ensure data completeness and compliance before insertion.
  4. **Conflict Resolution** — Merge or flag conflicting entries.
- **Integration Methods:**
  - **ERP** — REST APIs or direct DB connectors.
  - **SCADA** — MQTT or OPC-UA for telemetry.
  - **Safety DB** — SQL or JSON over HTTP.
  - **Geology/Planning Software** — File export or API upload.

---

5. Operational Flow Example

Scenario: Equipment Inspection in a deep mine zone with no live network.

1. Inspector logs findings in the tablet app.
2. Data is stored in **local PouchDB** immediately.
3. Several hours later, inspector enters a mesh-covered tunnel.
4. Device detects connectivity → **automatic sync** with surface CouchDB.
5. **Middleware** maps data to:
    - ERP for maintenance scheduling
    - Safety DB for compliance tracking
    - SCADA alerts if critical equipment is flagged
6. All systems are updated without manual re-entry.

---

6. Key Advantages Over Traditional On-Prem ERP

- Works **independently of LAN** — no "all systems down" during network loss.
- Supports **gradual, asynchronous sync** instead of requiring constant connectivity.
- Bridges **data silos** by integrating with multiple mining databases.
- Minimizes downtime and improves decision-making.

4.1. Example of Offline-First Sync Logic

```
Start device application
Connect to local PouchDB

LOOP while device in use:
    Collect new data from user or sensors
    Save to PouchDB
    IF network_available():
        Sync PouchDB with CouchDB
```

```
    ELSE:
        Keep collecting offline
END LOOP
```

This is simple, but in a mine, simplicity is a strength — fewer moving parts means fewer things to break.

---

5. Data Silo Issues

Mining operations rarely have a single, unified data platform [3]. Instead, they rely on a patchwork of systems: ERP for finance, SCADA for telemetry, safety reporting tools, geological modeling software, and ad-hoc spreadsheets.

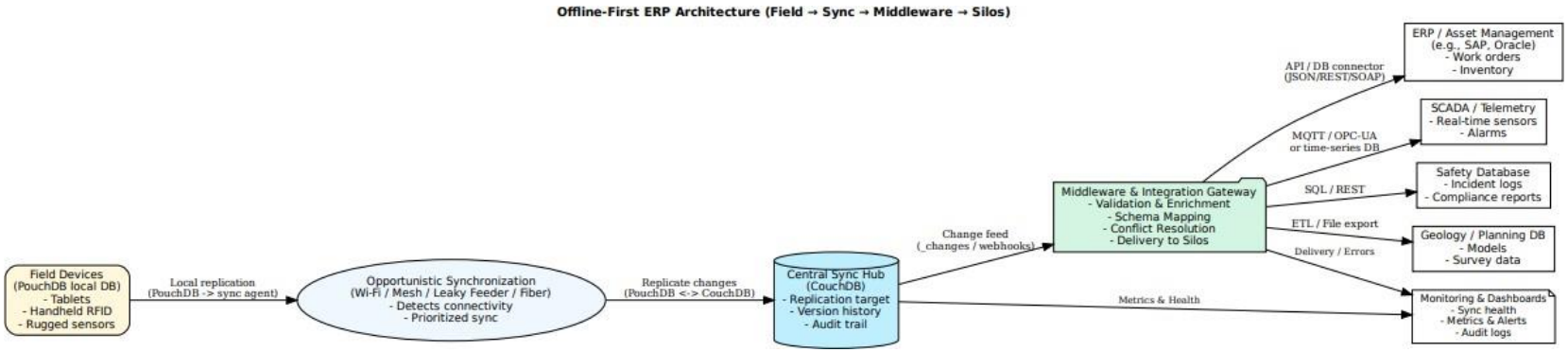
This creates "data silos" — isolated pockets of information that cannot easily be combined [3]. This leads to: - **Duplicate data entry** - **Inconsistent records** - **Delayed decisions** - **Higher maintenance costs**

---

6. Integrating Offline-First Architecture with Existing Data Silos

Overview

To be useful in live mines, an offline-first layer (PouchDB on devices + CouchDB sync server) must sit *on top of*, not *instead of*, the existing data silos. The pragmatic goal is to *bridge* silos with minimum disruption: capture data reliably at the edge during outages, then transform and route it into established systems when sync is possible.



High-level architecture

- **Field devices (PouchDB)**: Capture operational data offline.
- **Connectivity layer**: Any available link is used for sync.
- **CouchDB (surface)**: Acts as sync hub.
- **Middleware**: Transforms and routes to ERP, SCADA, Safety DB, Geology DB.
- **Existing silos**: Receive updates without changes to their architecture.

Practical flow (equipment inspection)

1. Inspector records check in PouchDB.
2. Stays local if offline.
3. Syncs when online to CouchDB.
4. Middleware maps and pushes to ERP, maintenance DB, and safety logs.

Middleware responsibilities

- Subscribe to CouchDB changes.

- Validate and enrich data.
- Map to target schema.
- Deliver via APIs/connectors.
- Handle conflicts and security.

Conflict resolution

Define sources of truth per domain. Use field-level merges or manual review queues where necessary.

Data model

Each record should include UUID, type, device_id, user_id, timestamp, source, and sync status.

Connectors

ERP via API/DB connectors, SCADA via MQTT/OPC-UA, safety logs via SQL/JSON API, assets via object storage.

Security

TLS, signed payloads, token-based authentication, full audit logs.

Deployment plan

- Stage 0: Lab simulation.
- Stage 1: Pilot in single section.
- Stage 2: Gradual rollout.

Limitations

- Latency not suitable for real-time control loops.
- Middleware mapping maintenance overhead.
- Edge storage constraints.

---

7. Benefits in the Mining Context

- Safety data never lost.

- No downtime during outages.
- Better reporting without double entry.
- Works in remote exploration sites.

---

8. Database Comparison: Traditional ERP vs. Offline-First

Feature	Traditional ERP DB	Offline-First (PouchDB + CouchDB)
Storage Location	Central server	On-device + sync to server
Network Requirement	Always-on LAN/internet	Only for sync events
Conflict Handling	Locked records	Multi-version merge resolution
Sync Method	Direct DB transactions	Batch replication
Resilience During Outages	Stops working	Fully functional
Schema Flexibility	Rigid, pre-defined	JSON-based, flexible
Ease of Edge Deployment	Complex setup	Simple app install
Suitable for Real-Time Control	Yes	Limited by sync delay

9. Operational Feature Comparison: Before vs. After Offline-First

Feature/Process	Current Operations (On-Prem)	With Offline-First ERP
Data Capture	Stops when LAN down	Continues regardless of connectivity
Safety Reporting	Delayed if network unavailable	Logs instantly, syncs later
Equipment Tracking	Limited to connected zones	Logs anywhere, updates when connected
Decision-Making	Based on last synced server data	Includes latest field data after sync
Integration Effort	Manual entries in multiple systems	Single entry, middleware distributes
Downtime Costs	High during outages	Reduced significantly
Field Productivity	Workers idle during network issues	Continuous work without interruption

10. Conclusion

The industry's definition of "offline ERP" has been too narrow — often meaning only "on-premise" [2]. In practice, on-prem ERP still halts without a live network. A true offline-first ERP — with device-level data capture, PouchDB/CouchDB sync, and middleware integration — ensures mining operations keep moving even when connectivity fails.

By addressing data silos [3], using proven underground communication systems [1][4], and leveraging offline-first tech, mines can dramatically reduce downtime, improve safety logging, and unify reporting without overhauling existing systems.

---

11. Future Work

While the offline-first ERP concept addresses immediate operational challenges, further research and prototyping can strengthen its applicability in mining:

11.1 Communication System Simulations

Develop simulation algorithms to model: - **Leaky Feeder Performance**: Signal decay over distance, amplifier placement optimization, and effect of branching on coverage. - **Mesh Network Routing**: Node failure scenarios, latency per hop, and optimal routing algorithms for variable mine topologies. - **RFID Tag Detection**: Probability of read success based on tag orientation, movement speed, and interference levels.

11.2 Data Synchronization Strategies

Experiment with: - **Incremental vs. Full Sync**: Testing how partial replication can reduce bandwidth usage underground. - **Conflict Resolution Policies**: Applying domain-specific merge rules for safety vs. production data. - **Compression Before Sync**: To optimize data transfer over low-bandwidth connections.

11.3 Middleware Optimization

- Dynamic schema mapping between PouchDB JSON and silo-specific database schemas.
- Automated API endpoint detection for legacy systems.
- Event-driven data routing to reduce sync-to-reporting latency.

11.4 Hybrid Communication Models

Integrating leaky feeder, mesh, and short-burst high-speed fibre access points to create “connectivity bubbles” in strategic mine zones, enabling periodic high-volume syncs without full coverage.

These future efforts will ensure that the offline-first ERP is not only resilient but also optimized for the unique constraints of underground mining operations.

Proof of Concept Done for this

Introduction

To validate the opportunistic sync architecture proposed in this paper, a practical proof of concept (PoC) was developed. The PoC replicates core aspects of field data capture in remote environments, using modern, open-source technologies that support offline-first operation and seamless synchronization with a central server.

Implementation Details

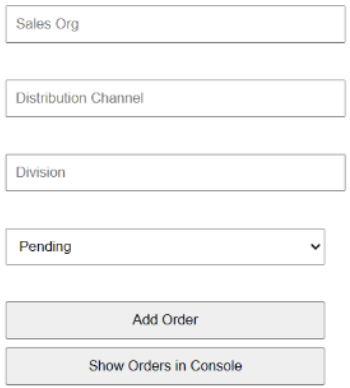
The PoC application is a web-based sales order management system designed with SAP-like data fields, intended as an analogous use case to mining data collection. Key components include:

- **Client-Side Storage:**
  Utilizes PouchDB, a JavaScript database running in-browser, which enables storing and retrieving data locally without requiring network connectivity. PouchDB is designed to replicate bi-directionally with CouchDB-compatible servers, supporting robust conflict resolution and eventual consistency.
- **Server-Side Sync Endpoint:**
  Employs express-pouchdb, a Node.js module providing a CouchDB-compatible HTTP API backed by LevelDB. This server acts as the central repository for synchronized data from multiple client devices.
- **Sync Mechanism:**
  The client performs live, continuous synchronization (`live: true, retry: true`) with the server database, ensuring all locally added or modified documents are replicated opportunistically once connectivity is restored.
- **User Interface:**
  Includes a responsive form for entering sales orders with fields like Order ID, Customer ID, Material Code, Quantity, Price, and Sales Organization. The interface updates in real time to reflect locally stored data and sync status.

Testing and Results

- The application was tested under simulated network conditions, including offline mode in browsers and manual network disconnections.
- Orders added while offline were immediately persisted locally, and upon network restoration, were synchronized automatically with the server.
- Server-side data consistency was confirmed via the Fauxton interface, showing all replicated documents accurately.
- Network status indicators in the UI provided feedback on connectivity, improving usability in intermittent networks.
- Sync logs demonstrated conflict-free replication and reliable data persistence.

Before turning offline, we have 3 orders in our DB (Local & server both synced)



Orders (Local)

Order ID	Date	Customer	Cust ID	Material	Description	Qty	Unit Price	Currency	Sales Org	Dist. Channel	Division	Status
2324	2025-07-30	Yash Tech	120427094	INV-SKU	SKU	213	311235.00	USD	0010	001	001	Pending
8232241	2025-08-19	Tesla	-0812307	INV-SKU	SKU	23	141.00	USD	0020	00	10	Pending
279369	2025-08-06	TeslaD	2214214413531	Inv-SKU	INvalid	13	1324.00	USD	0020	00	10	Pending


```
{"total_rows":3,"offset":0,"rows":[{"id":"2025-08-10T13:25:05.629Z","key":"2025-08-10T13:25:05.629Z","value":{"rev":"1-27b45184e530faf8247b13b4f876946a"},"doc":{"orderId":"279369","orderDate":"2025-08
06","customer":"TeslaD","customerId":"2214214413531","materialCode":"Inv-
SKU","materialDesc":"INvalid","quantity":13,"unitPrice":1324,"currency":"USD","salesOrg":"0020","distChannel":"00","division":"10","status":"Pending","_id":"2025-08-10T13:25:05.629Z","_rev":"1-
27b45184e530faf8247b13b4f876946a"}},{"id":"2025-08-10T15:18:32.433Z","key":"2025-08-10T15:18:32.433Z","value":{"rev":"1-4fc3cb82903dc1f6247b419bc0a8b56e"},"doc":{"orderId":"8232241","orderDate":"2025-
19","customer":"Tesla","customerId":"-0812307","materialCode":"INV-
SKU","materialDesc":"SKU","quantity":23,"unitPrice":141,"currency":"USD","salesOrg":"0020","distChannel":"00","division":"10","status":"Pending","_id":"2025-08-10T15:18:32.433Z","_rev":"1-
4fc3cb82903dc1f6247b419bc0a8b56e"}},{"id":"2025-08-10T15:22:02.937Z","key":"2025-08-10T15:22:02.937Z","value":{"rev":"1-60ab308832751c2038910117737e61b2"},"doc":{"orderId":"2324","orderDate":"2025-07-
30","customer":"Yash Tech","customerId":"120427094","materialCode":"INV-
SKU","materialDesc":"SKU","quantity":213,"unitPrice":311235,"currency":"USD","salesOrg":"0010","distChannel":"001","division":"001","status":"Pending","_id":"2025-08-10T15:22:02.937Z","_rev":"1-
60ab308832751c2038910117737e61b2"}}]}
```

Now we are turning the app offline –

Create Sales Order (Offline-First)

Offline

Order ID

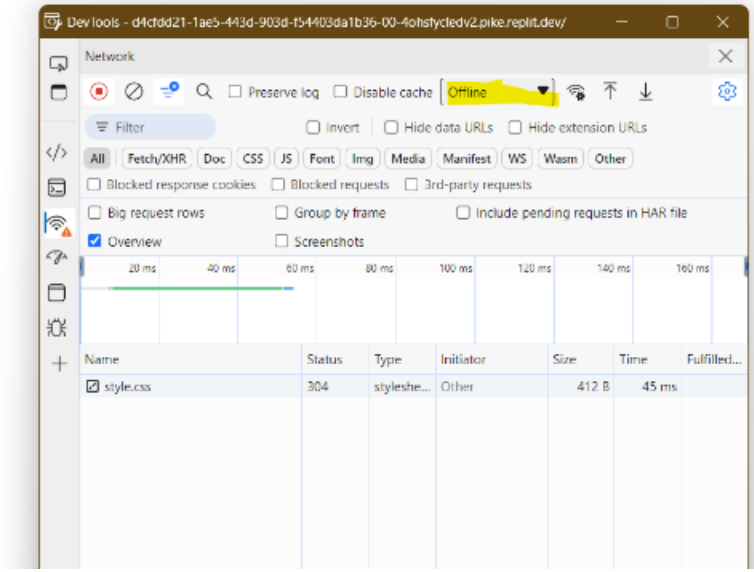
mm/dd/yyyy

Customer Name

Customer ID

Material Code

Material Description



After this, we add a new row of data, it gets made in the offline / local DB

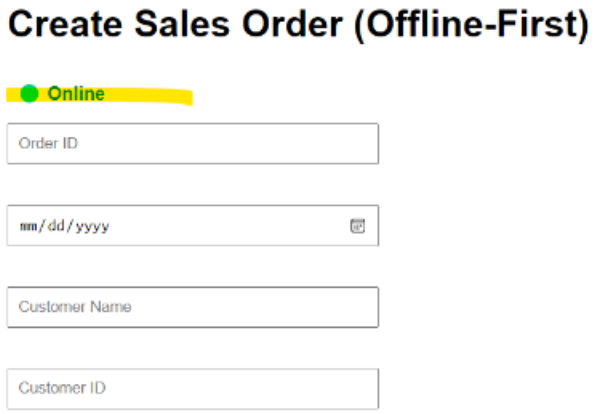
Orders (Local)

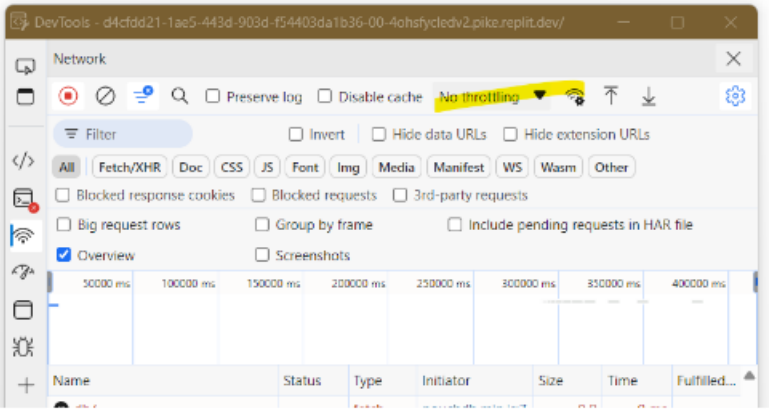
Order ID	Date	Customer	Cust ID	Material	Description	Qty	Unit Price	Currency	Sales Org	Dist. Channel	Division	Status
99074	2027-09-06	OfflineTestCust	224243	Offli-MAT2027	Offline sync test SKU	2	230.00	INR	IN01	00	12	Shipped

In the server side, we see that only 3 rows are there still,

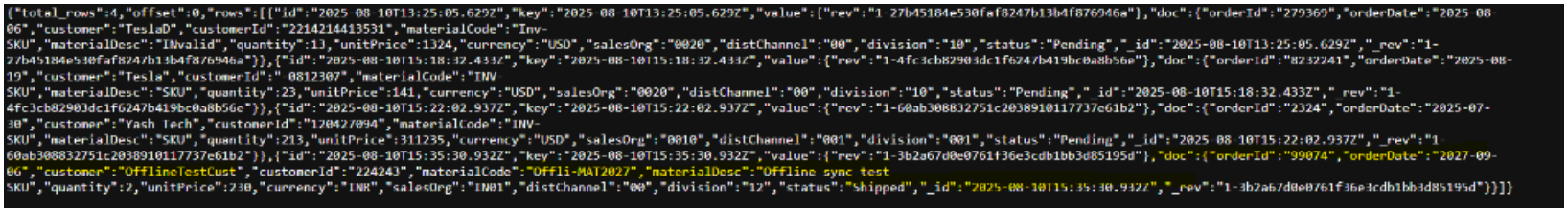

```
{"total_rows":3,"offset":0,"rows":[{"id":"2025-08-10T13:25:05.629Z","key":"2025-08-10T13:25:05.629Z","value":{"rev":"1-27b45184e530faf8247b13b4f876946a"},"doc":{"orderId":"279369","orderDate":"2025-08-06","customer":"TeslaD","customerId":"2214214413531","materialCode":"Inv-SKU","materialDesc":"INvalid","quantity":13,"unitPrice":1324,"currency":"USD","salesOrg":"0020","distChannel":"00","division":"10","status":"Pending","_id":"2025-08-10T13:25:05.629Z","_rev":"1-27b45184e530faf8247b13b4f876946a"}},{"id":"2025-08-10T15:18:32.433Z","key":"2025-08-10T15:18:32.433Z","value":{"rev":"1-4fc3cb82903dc1f6247b419bc0a8b56e"},"doc":{"orderId":"8232241","orderDate":"2025-08-19","customer":"Tesla","customerId":"-0812307","materialCode":"INV-SKU","materialDesc":"SKU","quantity":23,"unitPrice":141,"currency":"USD","salesOrg":"0020","distChannel":"00","division":"10","status":"Pending","_id":"2025-08-10T15:18:32.433Z","_rev":"1-4fc3cb82903dc1f6247b419bc0a8b56e"}},{"id":"2025-08-10T15:22:02.937Z","key":"2025-08-10T15:22:02.937Z","value":{"rev":"1-60ab308832751c2038910117737e61b2"},"doc":{"orderId":"2324","orderDate":"2025-07-30","customer":"Yash Tech","customerId":"120427094","materialCode":"INV-SKU","materialDesc":"SKU","quantity":213,"unitPrice":311235,"currency":"USD","salesOrg":"0010","distChannel":"001","division":"001","status":"Pending","_id":"2025-08-10T15:22:02.937Z","_rev":"1-60ab308832751c2038910117737e61b2"}}]}
```


Now, we are turning the app online –





The new order gets synced



References

1. Ghosh, A., Mishra, S., & Varma, A. *Status of Communication and Tracking Technologies in Underground Mines*.
2. Rahman, A., et al. *Agile Data Architecture in Mining Industry for Continuous Business-IT Alignment: EA Perspective*.
3. Ntwist Technologies. (2023). *Mine-to-Mill: Breaking Down Siloed Data in Mining Operations*. Retrieved from https://ntwist.com/blog/mine-to-mill-siloed-data
4. Nowak, M. (2010). *Basic Leaky Feeder Layout for Underground Mine Communications*. [Technical Document].