CASE STUDIES

**Case Study 1: Virtual IT Infrastructure & Security**

**Client Overview**

* **Type**: Product Startup (30+ employees)
* **Industry**: MarTech
* **Location**: Ireland, South Africa, London, and India
* **Duration**: 45 days
* **Budget**: $13K–$15K annually for 40 users
* **Role**: Infra Consultant

**Problem**

The client needed a **complete virtual IT infrastructure and security setup** for its newly formed India development team, operating without a physical office. They had no local IT admin or finance team and required a secure, scalable, and low-maintenance environment.

**Key Requirements**:

* Implement **Mobile Device Management (MDM)**
* Setup **enterprise-grade antivirus**
* Deploy **SASE (Secure Access Service Edge)** solution for remote security
* Deliver with minimal user disruption across global time zones

**Challenges**

* No in-house IT/Finance support in India
* Limited awareness of security tools and deployment practices
* Coordination across time zones with dispersed stakeholders
* Budget-conscious delivery with zero on-premise infra

**Our Approach**

1. **Requirement Discovery**
   * Engaged with global stakeholders to define user profiles, access policies, and compliance needs
   * Mapped minimum viable security architecture for a remote-first team
2. **Tool Identification & Procurement**
   * Recommended cost-effective and scalable tools for:
     + MDM (Mobile Device Management)
     + Antivirus with remote monitoring
     + SASE platform for secure global access
   * Ensured tools required minimal manual intervention and centralized control
3. **Deployment & Configuration**
   * Set up and configured tools across 40+ devices
   * Recommended **rental laptops** to reduce capital expenditure and increase agility
   * Enabled secure onboarding for distributed teams
4. **Handover & Support Enablement**
   * Identified, hired, and trained a local **IT Administrator**
   * Documented and handed over admin processes for ongoing operations

**Value Delivered**

* 🔐 Secure and compliant IT infra setup across multiple geographies
* 💸 Delivered within **$9K/year**—over **35% under budget**
* 🛠️ Setup completed in **< 45 days** with minimal user disruption
* 📲 Fully operational MDM + Antivirus + SASE stack
* 👨‍💼 Hired and trained IT Admin for long-term sustainability

**💡 Our Takeaway & Perspective**

Startups scaling across borders often overlook foundational IT setup until it becomes a blocker. This project highlights how **remote-first infrastructure can be secure, lean, and scalable**—without heavy upfront investment.

Key lessons:

* Minimalist doesn't mean insecure—**MDM + SASE + Antivirus is a powerful trio**
* **Well-defined access control and remote policies** are more valuable than physical office firewalls
* Helping clients hire and train internal staff ensures continuity beyond engagement

This case demonstrates that **even with tight budgets and global constraints, secure infrastructure can be deployed quickly, efficiently, and sustainably**.

**Case Study 2: AWS Cloud Optimization & Security Hardening**

**Client Overview**

* **Type**: Product Startup
* **Industry**: MarTech (Marketing Technology)
* **Location**: Ireland, South Africa, London, and India
* **Duration**: 20 days
* **Role**: Consultant

**Problem**

The client’s AWS infrastructure was unmanaged, insecure, and cost-inefficient—posing serious risks for their public-facing **Lead Management SaaS platform** with 140K+ daily users.

**Key Pain Points**:

* Unused and undocumented AWS services consuming resources
* No tagging strategy, cost tracking, or accountability
* No security best practices in place (IAM, EC2, S3, etc.)
* Entire infrastructure running in on-demand mode
* Monthly AWS cost spiked to ~$1500 without visibility

**Our Approach**

1. **AWS Infra Audit**
   * Conducted deep analysis of all 17 active AWS services
   * Identified unused instances, missing tags, and redundant resources
   * Mapped infrastructure with application usage for clarity
2. **Security Hardening**
   * Defined and implemented **minimum viable security baseline**
   * Applied IAM role-based access policies, custom permission sets
   * Secured S3, EC2, CloudFront, and Security Groups using least privilege principles
   * Updated resource and service-level policies
3. **Architecture Optimization**
   * Designed a **multi-AZ, fault-tolerant**, and **scalable** architecture
   * Ensured high availability for their SaaS platform with zero downtime
   * Prepared the infra for future CI/CD and DevOps adoption
4. **Cost Optimization**
   * Converted select EC2 instances to **dedicated or reserved**
   * Shut down or consolidated idle or overprovisioned services
   * Introduced tagging and budget monitoring for better control

**Value Delivered**

* 💰 **Reduced AWS monthly cost from $1500 to < $450**
* 🔐 **Security guidelines enforced** across core services
* ☁️ **Reliable, scalable AWS setup** ready for production load
* 📊 Gained visibility and accountability through tagging & monitoring
* 🚀 Delivered the engagement within **20 days** with lasting impact

**💡 Our Takeaway & Perspective**

Cloud environments can **quickly become liabilities** when unmanaged—even for modern SaaS startups. This engagement reinforced key lessons:

* **Security hardening and cost control go hand-in-hand**
* A lean, multi-AZ AWS setup can serve 100K+ daily users without high spend
* Early DevOps and tagging practices **save thousands later**
* Fast turnarounds are possible when the goal is clear and focused

This case illustrates that **cloud efficiency is less about cutting tools and more about visibility, governance, and purposeful architecture**.

**Case Study 3: .NET Modernization Strategy**

**Client Overview**

* **Type**: Software Services Firm (120+ staff)
* **Industry**: Software Services
* **Location**: India (HQ in US)
* **Duration**: 1 month
* **Role**: Tech Lead

**Problem**

The client had a legacy .NET 4.5 monolithic application that had been in production for over 7 years. It faced serious maintainability, performance, and deployment challenges.

**Key Pain Points**:

* Monolithic architecture with tightly coupled modules
* Frequent downtime and server crashes
* Increased complexity in bug fixes and feature development
* No CI/CD pipeline or cloud-native readiness
* Outdated tech stack not aligned with modern Azure capabilities

**Our Approach**

1. **Architecture Review**
   * Conducted in-depth code and architecture audit
   * Identified high-risk modules and legacy dependencies
2. **Migration Strategy**
   * Proposed phase-wise modernization roadmap
   * Designed a modular decomposition strategy using **Ocelot API Gateway** for service separation
   * Recommended migration from .NET 4.5 to **.NET Core / .NET 6+**
3. **Azure Optimization & DevOps Setup**
   * Suggested Azure deployment strategies to reduce costs
   * Introduced a lightweight CI/CD pipeline for future scalability
   * Proposed auto-scaling and monitoring to improve uptime
4. **Proof of Concept (PoC)**
   * Delivered a working PoC to validate microservice conversion feasibility
   * Included separated service boundaries, token-based auth, and routing via Ocelot

**Value Delivered**

* 🧱 Broke down monolith into manageable, modular services
* ⚙️ Reduced Azure infrastructure dependency and costs
* 📉 Downtime brought down to under **2%**
* ⏱️ Faster development and bug fix cycles
* 🚀 Client now equipped with a roadmap for full modernization

**💡 Our Takeaway & Perspective**

Modernizing legacy applications isn't just a tech upgrade—it’s a strategic business enabler. In this engagement:

* The challenge wasn’t only technical debt—it was also **organizational inertia**
* We focused on delivering **quick wins (PoC + Azure cost savings)** to gain stakeholder buy-in
* Phased migration using **API Gateway + .NET Core** allowed for **zero-downtime transformation**
* This laid the foundation for DevOps maturity and future cloud-native scale

This case highlights how **even a short engagement (1 month)** can spark transformation—when the right vision and incremental strategy are in place.

**Case Study 4: Delivery Assurance & Performance Management**

**Client Overview**

* **Type**: IT Services Firm (120+ staff)
* **Industry**: Software Services
* **Location**: India (HQ in US)
* **Duration**: 3 months
* **Role**: Management Consultant

**Problem**

The client's delivery organization lacked structure in managing both traditional and Agile projects. With no standard processes or tool usage, delivery outcomes were inconsistent and high-risk.

**Key Pain Points**:

* No defined delivery methodology, checklists, or reusable artifacts
* Project managers and leads untrained in industry-standard tools like MPP, Jira, Azure DevOps, Git, Confluence
* Delivery cycles were unpredictable, impacting timelines, quality, and client confidence
* No accountability framework or clarity on roles

**Our Approach**

We focused on introducing **lightweight governance** and **tool-enabled delivery maturity** without disrupting ongoing work:

1. **Delivery Assessment**
   * Reviewed active projects and engagement models
   * Identified patterns of inconsistency, tool underutilization, and lack of metrics
2. **Framework Design**
   * Developed a fit-for-purpose delivery framework combining Traditional + Agile best practices
   * Defined core artifacts: charters, WBS, sprint plans, RAID logs, retrospectives
3. **Tool Enablement & Training**
   * Set up PM tools: MPP for planning, Jira & ADO for execution, Confluence for documentation, Git for code management
   * Conducted role-based training and playbooks for PMs, Leads, and QA
4. **Pilot & Handover**
   * Piloted framework across 3 diverse project types (Fixed Bid, Agile Retainer, Hybrid)
   * Identified internal leaders to own and drive the framework post-handover

**Value Delivered**

* 📘 **Standardized project execution** across delivery teams
* 🔁 **Repeatable processes** leading to more predictable outcomes
* 🧑‍🏫 **Empowered PMs and leads** through tool training and clear roles
* 🛠️ **Unified tooling stack** adopted across teams
* ✅ Delivery framework fully implemented and transitioned in **3 months**

**💡 Our Takeaway & Perspective**

Delivery assurance is not about heavy frameworks—it's about the **right-sized process with high adoption**. In this case, the success came from:

* Keeping the framework **light yet structured**
* Aligning with both Agile and Traditional modes
* Enabling change **without slowing delivery**
* Prioritizing **people enablement over policy enforcement**

This project is a strong example of how **small to mid-sized IT firms can mature their delivery orgs in 90 days**—with tangible impact on client trust, velocity, and operational confidence.

**Case Study 5: GenAI-Powered Claims Audit Insights**

**Client Overview**

* **Type**: HealthCare Product BU (120+ staff)
* **Industry**: HealthCare
* **Location**: India (HQ in US)
* **Duration**: 2 months
* **Role**: Consultant / Architect

**Problem**

The Business Unit (BU) struggled with slow and technical-heavy processes to access claims audit insights.

**Key Pain Points**:

* Analysts depended on SQL experts or raised IT tickets, with 2-day SLA for simple data pulls
* SQL knowledge, while budgeted, wasn’t empowering non-technical users
* Teams lacked real-time access to patterns such as claim outliers, delays, or anomalies
* Reports were rigid, static, and delayed—impacting audit efficiency and decision-making

**Our Approach**

To democratize access to data and improve turnaround, we introduced a GenAI-native interface tailored to the BU’s needs:

1. **Data Contextualization**:  
    Transformed SQL tables into business-intelligible formats—mapping raw data fields to understandable concepts (e.g., “Outlier Claim > ₹X” instead of “SELECT \* FROM claims WHERE...”).
2. **LLM Integration**:
   * Used OpenAI’s GPT-4x and Mistral models
   * Enabled prompt-based querying for non-technical users to fetch summaries, identify anomalies, or generate on-the-fly reports
3. **Scalable Data Indexing**:
   * Built a monthly vectorized index of historical and new data using Qdrant
   * Allowed semantic search and context-aware querying
4. **Enablement for Future AI**:
   * Delivered a structured, growing corpus of domain-specific knowledge
   * Enabled client to later use this dataset as a foundation for **training their own internal LLM models**

**Value Delivered**

* ⚡ **Instant, self-serve insights** for business users with zero SQL dependency
* 📉 **Reduced ticket load** and faster decision-making across audit cycles
* 🧠 **AI-readiness built-in**: the project didn’t just solve a current problem—it paved the way for internal GenAI initiatives
* 🔁 **Reusable architecture**: the prompt-based system can be extended to other BUs (Finance, Legal, Compliance)
* ✅ Delivered in just **2 months**, with immediate ROI

**💡 Our Takeaway & Perspective**

This case showcases how **GenAI is not just a futuristic add-on—it can solve today's operational inefficiencies**. The most powerful transformation wasn't the LLM itself—it was **bridging business context and technical data**, and giving control back to the user.

This kind of lightweight, composable AI architecture—**Vector DB + Context Indexing + LLM + Prompt UI**—can be deployed in any domain that suffers from over-reliance on technical gatekeeping for data access. It’s low-cost, scalable, and fast to implement.