



MongoDB Day Frankfurt

Application Modernization Platform (AMP)



Legacy applications are no longer just
technical debt.

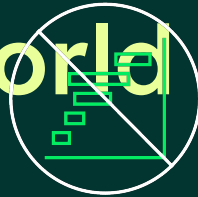
They're business risk.



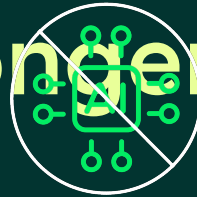
Your legacy systems were built for a different world



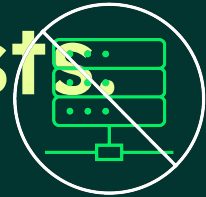
Without
Mobile Apps



Without
Real-Time Analytics



Without
AI & ML



Without
Global Scale

That world no longer exists.

Nobody really want to spent resources on application modernization, but



Escalating Costs and Technical Debt



- High and increasing operational and maintenance expenses.
- Difficulty finding and retaining staff with legacy system expertise.
- Accrual of significant technical debt, making changes costly and slow

Critical Security Risks and Compliance Burdens



- Increased vulnerability to cyber attacks
- Difficulty meeting evolving regulatory requirements
- Risk of data breaches, reputational damages, fines and legal issues

Inability to Innovate and Lack of Business Agility



- Slow time-to-market for new products and services
- Difficulty interacting with new technologies, partners, and 3rd party systems
- Lack of flexibility to adapt quickly to market changes and competitive pressures



So why haven't you modernized yet?



The fear of
breaking what
works



The complexity of
where to start



The uncertainty of
outcome



The lack of
expertise to do it
right

We get it. We've been there.



This is why we build the MongoDB Modernization Methodology

1

De-risk the
modernization
project

2

Accelerate
Application
Modernization

3

Minimize
modernization
requirements



Modernization Approaches

Based on industry research and modernization patterns, three primary approaches emerge:

Incremental Evolution

Gradual migration with minimal disruption

TIMELINE	RISK	INVESTMENT
3-5 Years	Moderate	High

Outcome

Maintains current position but fails to create differentiation. Technical debt persists.

DIY Modernization

Internal teams using traditional development tools and approaches

TIMELINE	RISK	INVESTMENT
2-4 Years	High	Very High

Outcome

Potential competitive advantage but high probability of delays, cost overruns and execution risk.

MongoDB Modernization

AI-driven transformation with proven methodologies and expert guidance

TIMELINE	RISK	INVESTMENT
6-12 Months	Minimal	Optimized

Outcome

Competitive advantage through accelerated innovation velocity, operational excellence, and future-ready architecture.



Why Traditional Modernization Fails

Manual Complexity

Human Error at Scale

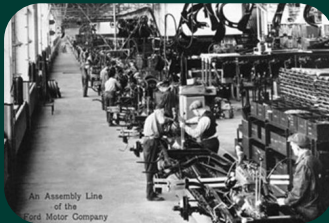
- Can't run business AND modernize
- Resource staffing complex and time consuming
- No docs on decades old legacy code
- Every developer codes differently

AI-Assisted Tool Reality

Unpredictable Code Generation

- LLM hallucinations create incorrect code lines
- Large files exceed LLM limits and attention
- Uncontrolled code generation / removal
- Still manual work, just with different tools

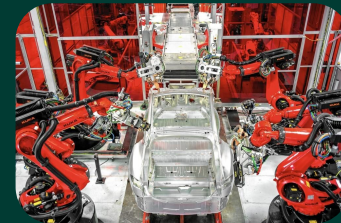
Traditional Software Development



AI-Assisted Tools



Software Factory



Spanning the entire Modernization lifecycle



Convert
Code

Spanning the entire Modernization lifecycle



Generate
Tests



Convert
Code



Validate

Spanning the entire Modernization lifecycle



Analyze



Generate
Tests



Convert
Code



Validate

Spanning the entire Modernization lifecycle



Analyze



Generate
Tests



Convert
Code



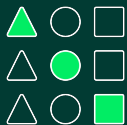
Validate



Deploy &
Migrate

Introducing the MongoDB Modernization Methodology

Accelerate and De-Risk Your Application Modernization



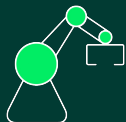
Processes

Agentic Automation &
Codified Processes



Expertise

Our people +
your people



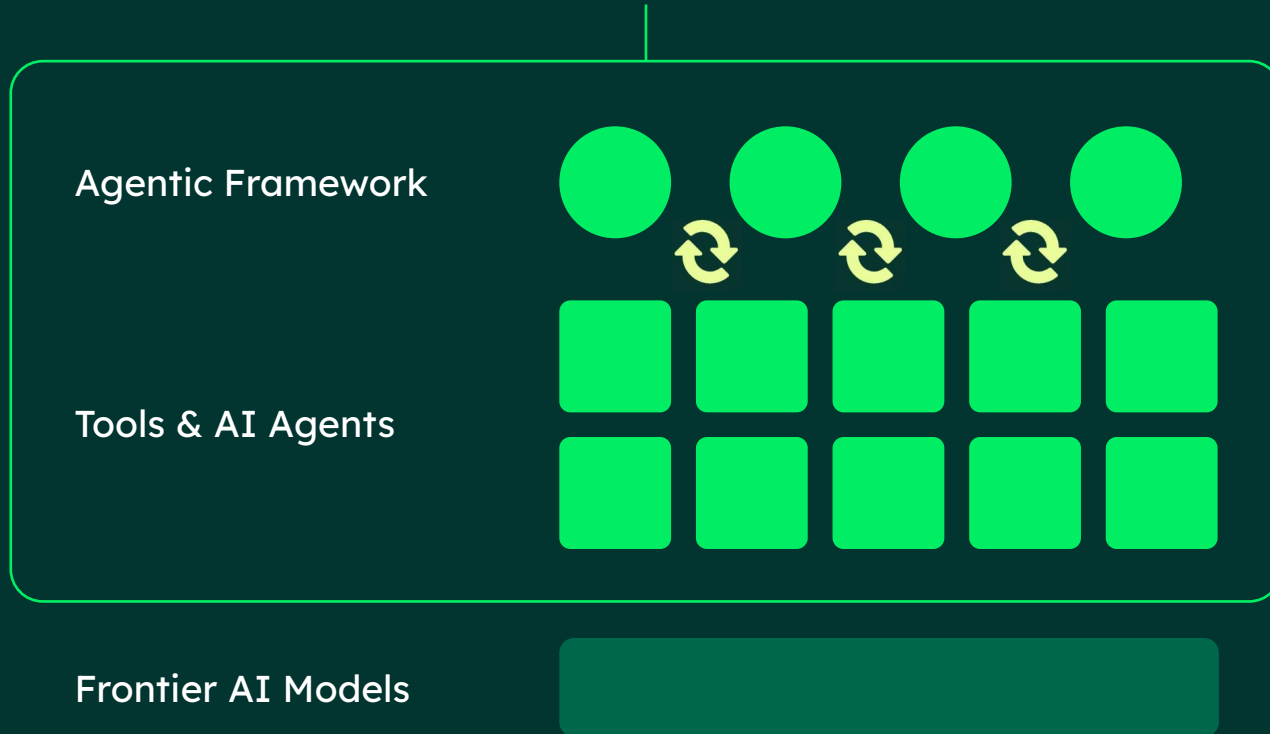
Software

AI + Tooling

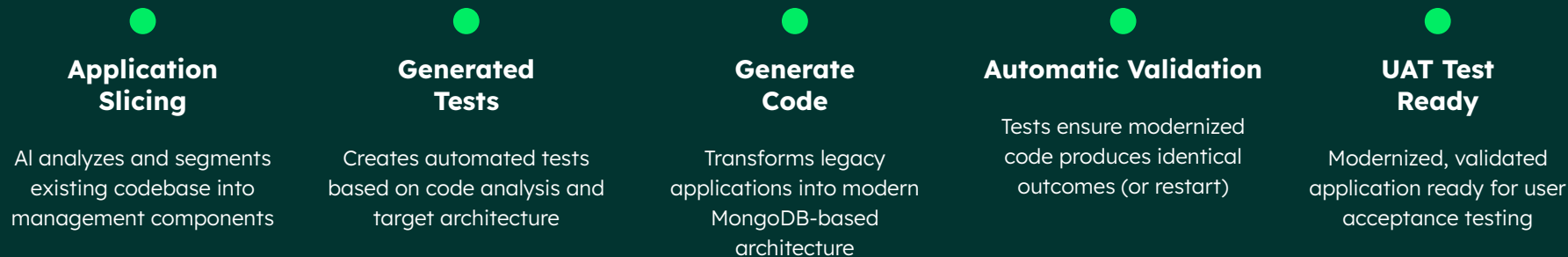
A software-driven solution that
industrializes application
modernization.

- Fast
- Repeatable
- Safe

Modernization with MongoDB



MongoDB Methodology



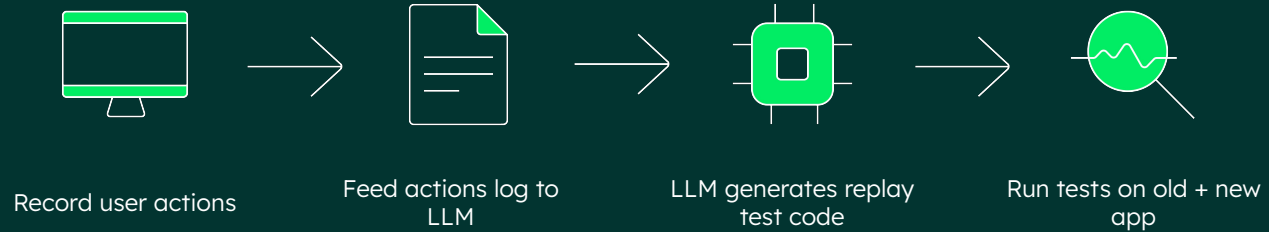
60%

Less code complexity → fewer bugs → reduced testing = accelerated innovation

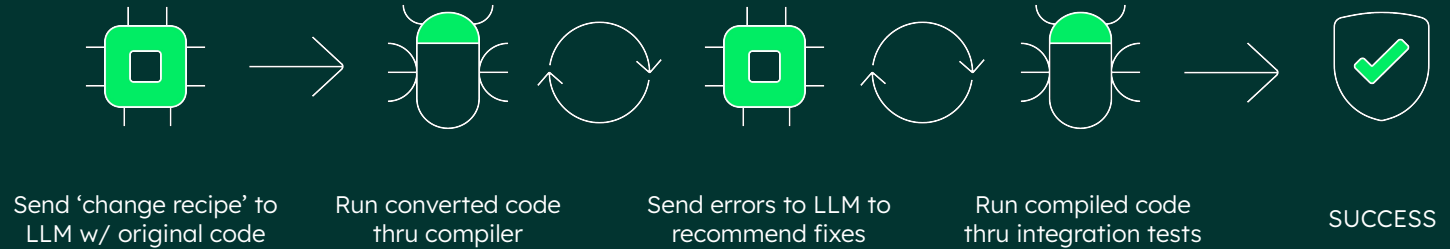


Examples of AI approach

End-to-end functional tests generation



Automated build & repair cyclic refactoring



Why modernizing to MongoDB?



RDBMS Technical Reality

Implementation Complexity

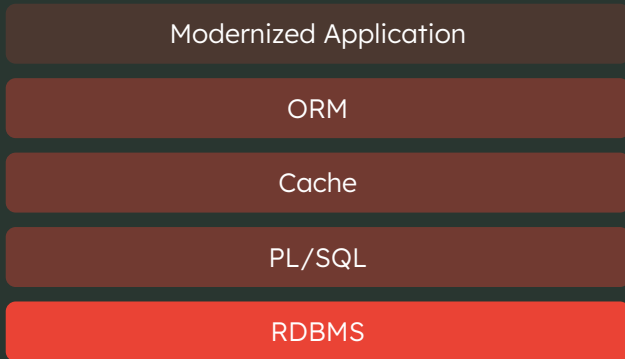
- Complex ORM layers create technical debt
- Rigid schemas require expensive migrations
- Multiple systems needed (cache, search, analytics)
- SQL expertise increasingly scarce and expensive

MongoDB Platform Advantage

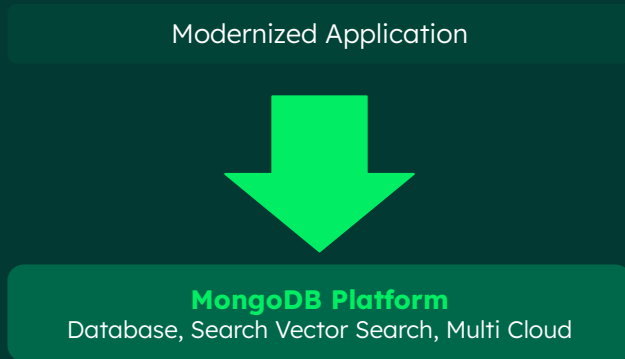
Platform Capabilities

- AI-ready document model supports vector embeddings natively
- Schema flexibility adapts to business changes
- Cloud provider neutrality, keyword search, tree search, etc.
- JSON- native development accelerates teams

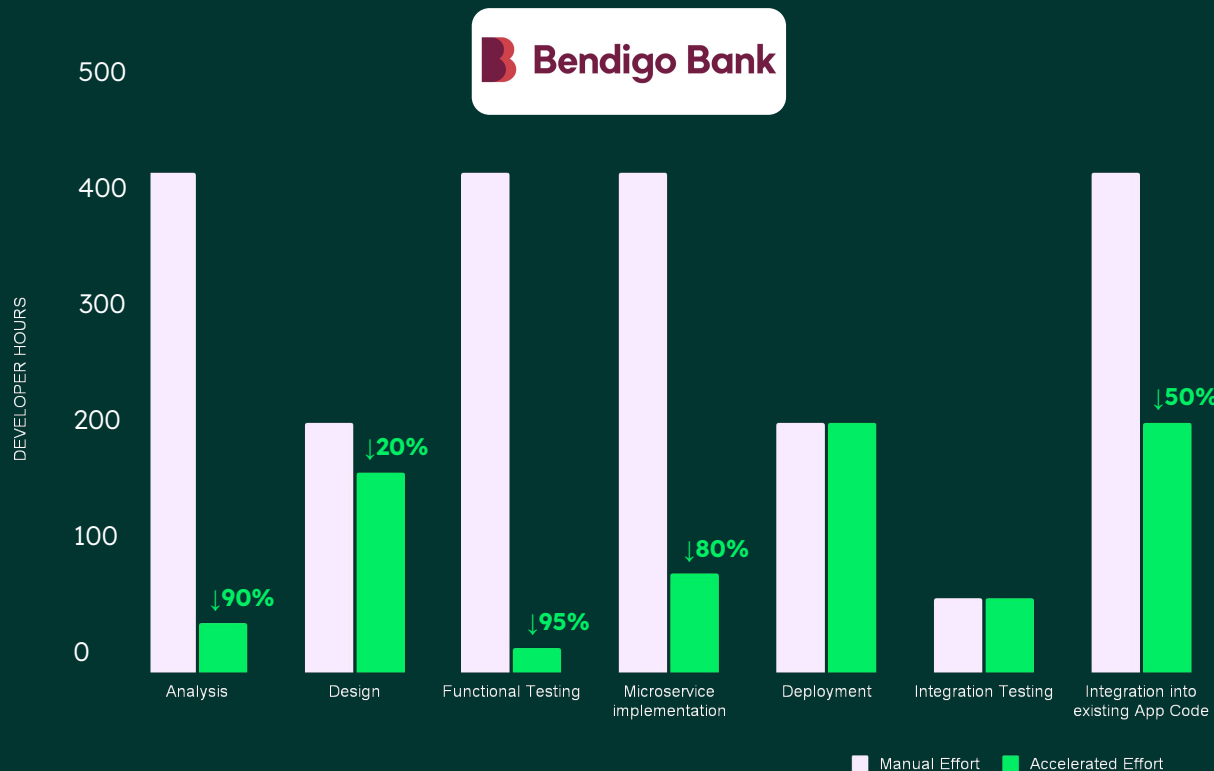
RDBMS Architecture



MongoDB Architecture



Proven Impact. Measurable Results.



63%

reduction in developer
hours spent on migration

3x

Developer Acceleration

Customer Story Bendigo Bank



Industry

Finance, serving over 2.7 million Australians

Application

ADS - Teller Application

Objectives

Consolidation of multiple applications that used a shared database.
Microservice extraction and shared use of Microservices

Target Stack

- TypeScript
- MongoDB



Legacy Stack

- Java
- SQL Server

 **63%**

reduction in developer hours
spent on migration

 **95%**

reduction in Functional
Testing

 **3x**

Developer Acceleration



“ **We chose MongoDB because** it offers us a cloud-agnostic database platform and an AI modernization approach, which helps to automate time-consuming tasks, **accelerate the upgrade of existing applications, and migrate them at a faster rate than ever before.**”

Having up to date technology has made a big impact on our employees and customers while proving to be fast, cost-effective, and reducing maintenance overheads.”



GEOFFROY DE RIDDER

Head of Technology & Operations
Lombard Odier Group

6x

Acceleration in
automated testing

50x

Faster migration from
JBoss to Quarkus

20x

Efficiency gain for
code migration



Industry

Finance

Application

Modernize over 250 applications

Objectives

JBoss retirement

Target Stack

- Java/Quarkus
- MongoDB



Legacy Stack

- Java
- JBOSS
- Oracle

↑ 50x

Faster code migration

↑ 20x

Faster migration from
Relationship to MongoDB

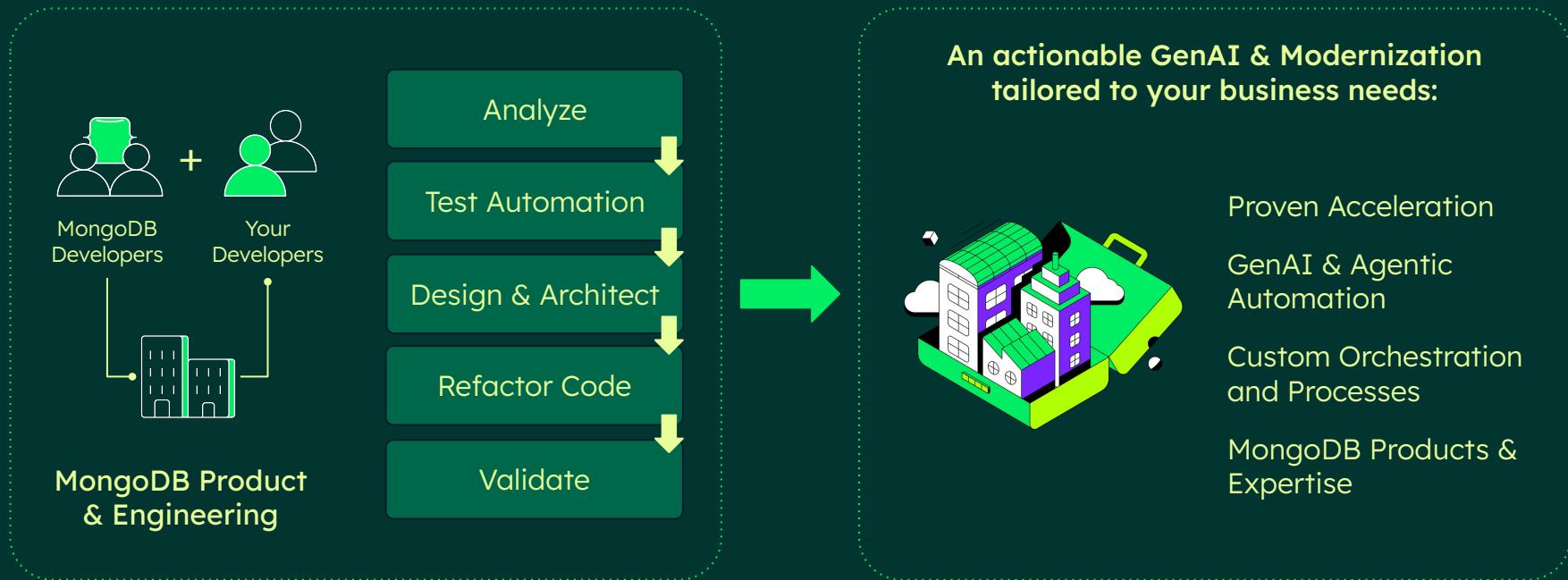
↑ 6x

Acceleration in
automated testing

Engagement Model



Start with an **incubator** to verify the baseline





AI Adoption Questions

1. How are you using Generative AI in your development lifecycle today?
2. We've found that AI-generated code often has functional and performance gaps. Does this match your experience?
3. Would you prefer a hybrid AI model with built-in guardrails and human verification over a pure "black box" AI solution?
4. How are you currently testing AI-generated code for security vulnerabilities?



Our teams working together..

1. How do you envision your engineering teams engaging with a platform like ours?
2. How can we best integrate with your teams to create a tight feedback loop with MongoDB?
3. Our initial focus is on the Java/Oracle stack. How prevalent is this in your portfolio compared to .NET/SQL Server? What other tech stacks do you need to modernize?



Deployment Questions

1. For your teams to adopt our platform, what are the most critical enterprise features we would need to build?
2. What are your security or compliance concerns with a hybrid model where our tools run in your environment but are managed by a cloud control plane?
3. Does a self-service modernization platform that empowers your own IT teams align with your long-term strategy?
4. Beyond the code and data, what are your biggest operational hurdles when deploying modernized applications into production?



MongoDB Day Frankfurt

Thank you

