Admissions & Support: support@mockattempt.com
Phone / WhatsApp: +91 98765 43210
Course Code: MONG • **Updated:** 01 Nov 2026
https://www.mockattempt.com/courses/mongodb-training

MONG • NOSQL DATABASES & DISTRIBUTED DATA ENGINEERING | **BEGINNER TO PRODUCTION MONGODB & ATLAS ARCHITECT**

100% LIVE INDUSTRY EXPERT COHORT + INTERNSHIP LETTER

MongoDB Training

Master MongoDB from BSON ObjectId internals, Schema Validation, Atomic CRUD Operators ($set, $inc, $push, $elemMatch) & ESR Compound Indexes to Aggregation Pipelines ($match, $group, $lookup, $facet), ACID Transactions, Replica Sets, Sharding & Atlas — with mongosh programs, 10×50 MCQ Mock Tests per topic, and Spoken Voice + Gemini AI Interviews at ₹2,499.

Duration: 15–17 Weeks • Live + Hands-On Projects (205 Hrs) • **Modules:** 16 • **Live Topics:** 53 • **Labs & Tasks:** 106 • **Mocks / AI Vivas:** 386 / 386 • **Projects:** 16 Module + 1 Live Client Prod

Fixed Non-Negotiable Tuition Fee: Base Fee **₹2,499** + 18% GST (₹450) = **Total Incl. GST: ₹2,949**

Live Batch Tracks: Weekday Live Online Batch (Mon – Fri) (Monday to Friday (5 Live Interactive Classes / Week) • 1.5 Hours / Day (60 Mins Live Architecture & Coding + 30 Mins Hands-On Lab) • 17–19 Weeks Fast-Track) | Weekend Intensive Live Online Batch (Sat – Sun) (Saturday & Sunday (Designed for Working Professionals & College Students) • 3.0 Hours / Day (2 Hrs Deep-Dive Live Coding + 1 Hr Guided Project Build) • 19–21 Weeks Weekend Track)

Core Technologies & Engineering Stack: MongoDB 8.0 • WiredTiger & BSON • mongosh & Compass • ESR B-Tree Indexes • Aggregation Framework • Replica Sets & Sharding • MongoDB Atlas & Node Driver

1. PROGRAM MASTERY OUTCOMES & ELIGIBILITY

End-to-End Engineering Outcomes:

1. Model high-scale BSON documents choosing accurately between Embedding, Referencing, Bucket, and Outlier patterns within the 16 MB limit
2. Execute atomic updates ($set, $inc, $addToSet, $pull, findOneAndUpdate) and eliminate COLLSCAN bottlenecks using ESR Compound & Multikey Indexes
3. Build multi-stage Aggregation Pipelines ($match, $group, $lookup, $unwind, $facet), Multi-Document ACID Transactions, Replica Sets, and Sharded Clusters
4. Clear 350 Timed Mock Exams (17,500 MongoDB MCQs) and 350 Spoken Voice + Gemini AI NoSQL Technical Interviews
5. Access-pattern schema design: embedding vs referencing, bucket/outlier patterns, and `$jsonSchema` validation.
6. High-speed multi-stage Aggregation Pipelines (`$match`, `$lookup`, `$unwind`, `$facet`, `$project`) replacing slow app loops.
7. ESR (Equality-Sort-Range) compound index design, TTL expiration indexes, and `explain("executionStats")` verification.
8. Multi-document ACID transactions, write/read concerns, Change Streams, and sharding shard-key selection.

Eligibility & Prerequisites: Basic familiarity with JSON objects/arrays — starts from NoSQL & document database fundamentals from scratch • Any laptop/PC running MongoDB Community Server / MongoDB Atlas Free Tier, mongosh, and MongoDB Compass • No prior MongoDB industry experience required — Module 1 starts from environment setup, core mental models, and plain-English fundamentals. • Basic computer & internet familiarity on a Windows, macOS, or Linux laptop (8 GB+ RAM recommended for smooth live coding labs). • Ideal for B.Tech / BCA / MCA / B.Sc IT students, non-CS beginners, career switchers, and working developers upgrading to 2026 production standards. • Commitment to attend live Weekday or Weekend classes, complete daily hands-on lab checklists, and practice topic-wise Mock Tests & AI Interviews.

2. DETAILED MODULE & TOPIC-WISE SYLLABUS (16 MODULES • 53 LIVE TOPICS)

MODULE 1 OF 16 • 12 LIVE + LAB HOURS • 4 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

Module 1: MongoDB & NoSQL Architecture, WiredTiger Engine, BSON Types & RDBMS Comparison

Understand why document databases excel at polymorphic agile workloads, how WiredTiger manages document-level concurrency & compression, and how 12-byte BSON `ObjectId`s and `Decimal128` work under the hood.

Objectives: ✓ Compare Document, Key-Value, Column-Family, and Graph NoSQL databases alongside CAP trade-offs • ✓ Explain WiredTiger document-level concurrency, Snappy/Zstd block compression, and write-ahead journaling • ✓ Inspect 12-byte `ObjectId` structure (4B timestamp + 5B random + 3B counter) and exact `Decimal128` currency storage

1.1 MongoDB Architecture — Document Model, WiredTiger Storage Engine & Document-Level Concurrency [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply MongoDB Architecture — Document Model, WiredTiger Storage Engine & Document-Level Concurrency in real MongoDB systems — specifically: Collections...

1.2 NoSQL Database Paradigms — Document, Key-Value, Wide-Column, Graph & CAP / PACELC Trade-Offs [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply NoSQL Database Paradigms — Document, Key-Value, Wide-Column, Graph & CAP / PACELC Trade-Offs in real MongoDB systems — specifically: Why web-scale...

1.3 MongoDB Advantages Over RDBMS — Zero-Downtime Polymorphism, Locality & Native Sharding [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply MongoDB Advantages Over RDBMS — Zero-Downtime Polymorphism, Locality & Native Sharding in real MongoDB systems — specifically: Eliminating ORM obj...

1.4 BSON Data Types — 12-Byte `ObjectId`, `Decimal128`, `ISODate`, `Int32`/`Long`, `BinData` & `16 MB` Cap [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply BSON Data Types — 12-Byte `ObjectId`, `Decimal128`, `ISODate`, `Int32`/`Long`, `BinData` & `16 MB` Cap in real MongoDB systems — specifically: Why...

Module 1 Hands-On Capstone Project (100 Marks): Polymorphic Multi-Track Course Catalog BSON Blueprint (`ObjectId` + `Decimal128` + `ISODate`) — Design a polymorphic `courses` BSON collection storing diverse IT tracks (video courses, live cohorts, coding bootcamps) with shared core metadata, `NumberDecimal("999.00")` exact fees, `ISO...

MODULE 2 OF 16 • 12 LIVE + LAB HOURS • 4 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

Module 2: `mongosh` Shell, Collections, `$jsonSchema` Validation & Production Data Modeling

Navigate `mongosh` & Compass like a pro, enforce server-side `$jsonSchema` rules on collections, use Capped/Time-Series collections, and master Embedding vs Referencing design patterns.

Objectives: ✓ Connect securely via `mongodb://` and `mongodb+srv://` URIs in `mongosh` and Compass • ✓ Enforce server-side `$jsonSchema` document validation (`validationAction: "error"` vs `"warn"`) • ✓ Avoid the Unbounded Growing Array anti-pattern using Bucket, Subset, and Extended Reference patterns

2.1 MongoDB Shell (`mongosh`) & Compass — `mongodb+srv://` URIs, Scripting & Diagnostics [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply MongoDB Shell (`mongosh`) & Compass — `mongodb+srv://` URIs, Scripting & Diagnostics in real MongoDB systems — specifically: Modern Node.js-powere...

2.2 Creating & Dropping Databases — Lazy Materialization, `db.stats()` & Namespace Hygiene [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Creating & Dropping Databases — Lazy Materialization, `db.stats()` & Namespace Hygiene in real MongoDB systems — specifically: Why `use academy_db...

2.3 Collections — `createCollection()`, `$jsonSchema` Validation, Capped & Time-Series Collections [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Collections — `createCollection()`, `$jsonSchema` Validation, Capped & Time-Series Collections in real MongoDB systems — specifically: Implicit vs...

2.4 MongoDB Data Modeling — Embedding vs Referencing, Subset/Bucket Patterns & Unbounded Arrays [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply MongoDB Data Modeling — Embedding vs Referencing, Subset/Bucket Patterns & Unbounded Arrays in real MongoDB systems — specifically: 1-to-1 and 1-t...

Module 2 Hands-On Capstone Project (100 Marks): Validated Student Telemetry & Cohort Schema (`$jsonSchema` + Bucket Pattern) — Create a `students` collection protected by server-side `$jsonSchema` validation (`bsonType`, `required`, `minimum`/`maximum` on scores, regex on email) and a `topic_events_buckets` collecti...

MODULE 3 OF 16 • 12 LIVE + LAB HOURS • 4 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

Module 3: CRUD Mastery — Bulk Inserts, `find()` Projections, Cursors, `sort()` & Keyset Pagination

Perform high-speed ordered/unordered batch inserts, shape lean network payloads with projections, stream million-doc cursors without OOM crashes, and replace slow `skip(100000)` with O(log N) range pagination.

Objectives: ✓ Understand `insertMany({ ordered: false })` partial-continue semantics on duplicate keys • ✓ Write projection specs avoiding illegal inclusion/exclusion mixing (except `_id: 0`) • ✓ Eliminate deep `skip()` latency spikes and 100 MB in-memory sort errors using indexed `_id` cursor pagination

3.1 Inserting Documents — `insertOne()`, `insertMany({ ordered: false })` & Duplicate-Key Idempotency [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Inserting Documents — `insertOne()`, `insertMany({ ordered: false })` & Duplicate-Key Idempotency in real MongoDB systems — specifically: How `ins...

3.2 `find()` & Projections — Nested Dot-Notation, Exact Object Match Trap & Covered Projections [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply `find()` & Projections — Nested Dot-Notation, Exact Object Match Trap & Covered Projections in real MongoDB systems — specifically: Querying top-l...

3.3 MongoDB Cursors — Lazy Batches (`for await`), `batchSize()`, Cursor Timeouts & `countDocuments` vs `estimatedDocumentCount` [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply MongoDB Cursors — Lazy Batches (`for await`), `batchSize()`, Cursor Timeouts & `countDocuments` vs `estimatedDocumentCount` in real MongoDB system...

3.4 `sort()`, `limit()` & `skip()` — 100 MB Blocking Sort Cap, BSON Type Sort Order & Keyset Pagination [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply `sort()`, `limit()` & `skip()` — 100 MB Blocking Sort Cap, BSON Type Sort Order & Keyset Pagination in real MongoDB systems — specifically: Ascend...

Module 3 Hands-On Capstone Project (100 Marks): High-Throughput Question Bank Importer & Sub-Millisecond Keyset Paginator — Build an idempotent `insertMany({ ordered: false })` importer loading 10,000 MCQ documents with deterministic `_id` keys, paired with a lean projected `find({ topicId, _id: { $lt: lastSeenId...

MODULE 4 OF 16 • 12 LIVE + LAB HOURS • 4 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

Module 4: Atomic Document Updates, Array Operators, Idempotent Upserts & TTL Deletions

Mutate documents atomically in place without read-modify-write race conditions using `$set`, `$inc`, `$push`, `$addToSet`, `$pull`, `findOneAndUpdate()`, `{ upsert: true }`, and self-cleaning TTL indexes.

Objectives: ✓ Use `$inc`, `$min`, `$max`, `$set`, and `$unset` atomically instead of fetching & saving in application code • ✓ Manipulate embedded arrays cleanly with `$push` (`$each`, `$slice`, `$sort`), `$addToSet`, `$pull`, and positional `$[<id>]` `arrayFilters` • ✓ Auto-expire OTPs, sessions, and audit logs with zero cron jobs using MongoDB **TTL Indexes**

4.1 Updating Documents — `updateOne`, `updateMany`, `replaceOne` & Atomic `findOneAndUpdate` [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Updating Documents — `updateOne`, `updateMany`, `replaceOne` & Atomic `findOneAndUpdate` in real MongoDB systems — specifically: Targeted field pa...

4.2 MongoDB Update Operators — `$set`, `$unset`, `$inc`, `$min`/`$max`, `$push`/`$slice`, `$addToSet` & `$pull` [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply MongoDB Update Operators — `$set`, `$unset`, `$inc`, `$min`/`$max`, `$push`/`$slice`, `$addToSet` & `$pull` in real MongoDB systems — specifically...

4.3 Idempotent Upserts (`{ upsert: true }`, `$setOnInsert`) & Unique Index Concurrency Safety [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Idempotent Upserts (`{ upsert: true }`, `$setOnInsert`) & Unique Index Concurrency Safety in real MongoDB systems — specifically: Insert-or-update...

4.4 Deleting Documents — `deleteOne()`, `deleteMany()`, Soft Deletes & Self-Expiring **TTL Indexes** [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Deleting Documents — `deleteOne()`, `deleteMany()`, Soft Deletes & Self-Expiring **TTL Indexes** in real MongoDB systems — specifically: Removing...

Module 4 Hands-On Capstone Project (100 Marks): Race-Free Seat Counter, Bounded Recent-Activity Ring Buffer & Self-Expiring OTP Store — Implement an atomic cohort seat reservation using `findOneAndUpdate({ code, seatsLeft: { $gt: 0 } }, { $inc: { seatsLeft: -1 }, $push: { recentJoins: { $each: [entry], $slice: -10 } } }, { r...

MODULE 5 OF 16 • 10 LIVE + LAB HOURS • 3 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

Module 5: Comparison, Logical, Array (`$elemMatch`) & Full-Text Search Query Operators

Filter complex nested BSON structures using `$in`, `$exists`, `$type`, `$elemMatch` multi-condition array matching, boolean `$and`/`$or` index merges, and weighted `$text` search.

Objectives: ✓ Master `$elemMatch` so multi-predicate array queries never match across two separate array items accidentally • ✓ Combine `$and`, `$or`, `$in`, and `$exists` while preserving B-Tree index selectivity • ✓ Build weighted multi-field `$text` search indexes and anchored case-sensitive `^prefix` regex lookups

5.1 Query & Array Operators — `$gt`/`$in`, `$exists`, `$type`, `$all`, `$size` & The `$elemMatch` Rule [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Query & Array Operators — `$gt`/`$in`, `$exists`, `$type`, `$all`, `$size` & The `$elemMatch` Rule in real MongoDB systems — specifically: Compari...

5.2 Logical Query Operators — Implicit vs Explicit `$and`, `$or` Index Merging, `$nor` & `$not` [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Logical Query Operators — Implicit vs Explicit `$and`, `$or` Index Merging, `$nor` & `$not` in real MongoDB systems — specifically: Implicit comma...

5.3 Full-Text Search (`$text`, `$search`, `textScore`) & Anchored Prefix Regular Expressions [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Full-Text Search (`$text`, `$search`, `textScore`) & Anchored Prefix Regular Expressions in real MongoDB systems — specifically: Stemming & stop-w...

Module 5 Hands-On Capstone Project (100 Marks): Multi-Faceted Course & Skill Search Engine (`$elemMatch` + Weighted `$text` + Anchored Prefix) — Build a MongoDB search query layer supporting embedded module array filtering via `$elemMatch`, multi-tag `$all` + `$in` filters, weighted full-text relevance ranking (`{ score: { $meta: "te...

MODULE 6 OF 16 • 12 LIVE + LAB HOURS • 3 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

Module 6: B-Tree Indexing Internals, The **ESR Rule**, `explain("executionStats")` & `bulkWrite()`

Eliminate slow `COLLSCAN` queries by mastering MongoDB's **ESR (Equality, Sort, Range)** compound index rule, Multikey/Partial/Wildcard indexes, `.explain("executionStats")` ratio tuning, and `bulkWrite()`.

Objectives: ✓ Order compound index fields strictly by **Equality -> Sort -> Range (ESR)** so queries never hit blocking `SORT` stages • ✓ Know why MongoDB forbids indexing TWO parallel arrays in the same Compound Multikey Index • ✓ Achieve a near `1:1` `totalKeysExamined : nReturned` ratio in `.explain("executionStats")` and batch mutations with `bulkWrite()`

6.1 B-Tree Indexing Mastery — Compound **ESR Rule, Multikey Array Limits, Partial & Sparse Indexes** [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply B-Tree Indexing Mastery — Compound **ESR Rule**, Multikey Array Limits, Partial & Sparse Indexes in real MongoDB systems — specifically: Single-fi...

6.2 Query Plan Cache & `.explain("executionStats")` — `IXSCAN` vs `COLLSCAN`, `FETCH` & Covered Queries [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Query Plan Cache & `.explain("executionStats")` — `IXSCAN` vs `COLLSCAN`, `FETCH` & Covered Queries in real MongoDB systems — specifically: How Mo...

6.3 High-Throughput `bulkWrite()` — Combining `insertOne`, `updateOne`, `replaceOne` & `deleteOne` [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply High-Throughput `bulkWrite()` — Combining `insertOne`, `updateOne`, `replaceOne` & `deleteOne` in real MongoDB systems — specifically: Executing m...

Module 6 Hands-On Capstone Project (100 Marks): ESR Index Redesign, Covered Query Verification & 50,000-Op `bulkWrite` Sync Engine — Optimize a slow student attempt query (`status == "submitted"`, sorted by `submittedAt DESC`, range `score >= 60`) using an exact ESR compound index `{ status: 1, submittedAt: -1, score: 1 }...

MODULE 7 OF 16 • 16 LIVE + LAB HOURS • 4 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

Module 7: MongoDB Aggregation Framework (`$match`, `$group`, `$unwind`, `$lookup`, `$facet`) & Geospatial

Build real-time analytical pipelines directly inside MongoDB using `$match`, `$project`, `$group`, `$unwind`, `$lookup` joins, single-scan `$facet` dashboards, and `2dsphere` Geospatial queries.

Objectives: ✓ Place selective `$match` and `$sort` first so pipelines exploit B-Tree indexes before `$group`/`$lookup` • ✓ Aggregate KPIs with `$group`, multi-widget `$facet`, and `$merge` / `$out` materialized rollups • ✓ Perform correlated/filtered `$lookup` sub-pipelines, safe `$unwind`, and `2dsphere` GeoJSON `$nearSphere` radius searches

7.1 Aggregation Pipeline Architecture — `$match`, `$project`, `$addFields`/`$set`, `$sort`, `$limit` & `$merge` [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Aggregation Pipeline Architecture — `$match`, `$project`, `$addFields`/`$set`, `$sort`, `$limit` & `$merge` in real MongoDB systems — specifically...

7.2 `$group`, `$bucket` & `$facet` — Accumulators (`$sum`, `$avg`, `$topN`) & Single-Scan Dashboards [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply `$group`, `$bucket` & `$facet` — Accumulators (`$sum`, `$avg`, `$topN`) & Single-Scan Dashboards in real MongoDB systems — specifically: Grouping...

7.3 `$unwind` Array Deconstruction & `$lookup` Cross-Collection Left-Outer Joins [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply `$unwind` Array Deconstruction & `$lookup` Cross-Collection Left-Outer Joins in real MongoDB systems — specifically: Flattening an N-element array...

7.4 Geospatial Indexing (`2dsphere`) — GeoJSON `[lng, lat]` Order, `$nearSphere` & `$geoNear` [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Geospatial Indexing (`2dsphere`) — GeoJSON `[lng, lat]` Order, `$nearSphere` & `$geoNear` in real MongoDB systems — specifically: Storing RFC-7946...

Module 7 Hands-On Capstone Project (100 Marks): Multi-Faceted Executive Analytics Pipeline (`$match` -> `$lookup` -> `$facet`) & Nearby Center Locator — Write a single-pass `$facet` MongoDB Aggregation Pipeline returning paginated course leaderboard rows, score distribution buckets (`$bucket`), and overall revenue KPIs simultaneously, plus a...

MODULE 8 OF 16 • 10 LIVE + LAB HOURS • 3 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

Module 8: Multi-Document ACID Transactions, Read/Write Concerns & SQL-to-MongoDB Translation

Execute multi-collection ACID transactions safely with `session.withTransaction()`, tune durability vs latency via `writeConcern` (`w: "majority"`) & `readConcern`, and translate complex SQL mindsets to MongoDB.

Objectives: ✓ Use `session.withTransaction()` for automatic `TransientTransactionError` retry handling across Replica Sets/Shards • ✓ Configure `w: "majority", j: true` + `readConcern: "majority"` to eliminate dirty/rollback reads during failovers • ✓ Map relational SQL constructs (`JOIN`, `GROUP BY`, `HAVING`, `ALTER`) cleanly to idiomatic MongoDB

8.1 Client Sessions & Multi-Document ACID Transactions (`startSession`, `withTransaction`) [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Client Sessions & Multi-Document ACID Transactions (`startSession`, `withTransaction`) in real MongoDB systems — specifically: Introduced in Mongo...

8.2 Tunable Durability & Isolation — `writeConcern` (`w: "majority"`, `j: true`) & `readConcern` (`majority`, `snapshot`) [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Tunable Durability & Isolation — `writeConcern` (`w: "majority"`, `j: true`) & `readConcern` (`majority`, `snapshot`) in real MongoDB systems — sp...

8.3 SQL-to-MongoDB Translation Guide — DDL, DML, `JOIN`, `GROUP BY` & `HAVING` Equivalents [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply SQL-to-MongoDB Translation Guide — DDL, DML, `JOIN`, `GROUP BY` & `HAVING` Equivalents in real MongoDB systems — specifically: Side-by-side mental...

Module 8 Hands-On Capstone Project (100 Marks): Multi-Collection Wallet Debit + Certificate Issuance ACID Transaction Engine — Implement a Replica-Set-safe `session.withTransaction(async () => { ... }, { readConcern: { level: "snapshot" }, writeConcern: { w: "majority" } })` workflow atomically debiting `wallets`, i...

MODULE 9 OF 16 • 12 LIVE + LAB HOURS • 3 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

Module 9: High Availability **Replica Sets** & Horizontal **Sharding** Architecture

Keep production clusters online 24/7 with 3-Node Replica Set automatic elections & idempotent Oplog replication, and scale petabyte write workloads horizontally across Sharded Clusters (`mongos` + Config Servers + Shard Keys).

Objectives: ✓ Understand Primary-Secondary-Secondary (`P-S-S`) Replica Set elections, Oplog capped collection replay, and why Arbiters (`P-S-A`) risk data durability • ✓ Trace Sharded Cluster components (`mongos` stateless router, CSRS Config Servers, Shard Replica Sets, Chunk Balancer) • ✓ Design high-cardinality, non-monotonic Compound Shard Keys avoiding Jumbo Chunks & Scatter-Gather broadcasts

9.1 MongoDB Replica Sets — Primary Elections, Idempotent `oplog.rs`, Heartbeats & Read Preferences [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply MongoDB Replica Sets — Primary Elections, Idempotent `oplog.rs`, Heartbeats & Read Preferences in real MongoDB systems — specifically: 3+ odd-numb...

9.2 Horizontal Sharding Architecture — `mongos` Query Routers, CSRS Config Servers & Chunk Balancing [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Horizontal Sharding Architecture — `mongos` Query Routers, CSRS Config Servers & Chunk Balancing in real MongoDB systems — specifically: When vert...

9.3 Shard Key Selection — Cardinality, Frequency (Jumbo Chunks) & Non-Monotonic Hashed/Compound Keys [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Shard Key Selection — Cardinality, Frequency (Jumbo Chunks) & Non-Monotonic Hashed/Compound Keys in real MongoDB systems — specifically: The three...

Module 9 Hands-On Capstone Project (100 Marks): Multi-Tenant SaaS Sharding & Replica-Set Read-Scaling Architecture Blueprint — Architect a global SaaS learning telemetry cluster using 3-member `P-S-S` Replica Sets per shard, design a compound shard key `{ tenantId: 1, eventDate: 1 }` avoiding monotonic `_id` / `ISOD...

MODULE 10 OF 16 • 10 LIVE + LAB HOURS • 3 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

Module 10: Security RBAC, `MongoDB Atlas` Cloud Operations, Backups & Node.js Driver Patterns

Lock down MongoDB with SCRAM-SHA-256, Least-Privilege RBAC roles & NoSQL-injection sanitization, operate managed `MongoDB Atlas` clusters + PITR backups, and write leak-free singleton `MongoClient` services in Node.js.

Objectives: ✓ Provision least-privilege `readWrite` application users (`SCRAM-SHA-256`), TLS encryption, and block `$ne`/`$where` NoSQL injection • ✓ Manage MongoDB Atlas VPC Peering, Performance Advisor, `mongodump --oplog` / continuous PITR backups, and Atlas Vector/Full-Text Search • ✓ Share a single pooled `MongoClient` singleton across your Node.js app and subscribe to real-time `watch()` Change Streams

10.1 MongoDB Security — `SCRAM-SHA-256`, Least-Privilege RBAC Roles, Encryption at Rest & Blocking NoSQL Injection [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply MongoDB Security — `SCRAM-SHA-256`, Least-Privilege RBAC Roles, Encryption at Rest & Blocking NoSQL Injection in real MongoDB systems — specifical...

10.2 MongoDB Atlas & Production Backups — Multi-AZ Clusters, `mongodump --oplog`, PITR & Atlas Search [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply MongoDB Atlas & Production Backups — Multi-AZ Clusters, `mongodump --oplog`, PITR & Atlas Search in real MongoDB systems — specifically: Operating...

10.3 Official Node.js `mongodb` Driver in Production — Singleton `MongoClient` Pool, Timeouts & Change Streams [50 mins • Assignment (100M) • Hands-On Lab (50M) • 10 Mock Sets • 10 AI Vivas]

Purpose: To understand and apply Official Node.js `mongodb` Driver in Production — Singleton `MongoClient` Pool, Timeouts & Change Streams in real MongoDB systems — specifically:...

Module 10 Hands-On Capstone Project (100 Marks): Hardened Node.js + MongoDB Atlas Production Microservice & Change-Stream Notifier — Build a production-ready Node.js repository layer using a singleton pooled `MongoClient` (`maxPoolSize: 20`, `serverSelectionTimeoutMS: 5000`), strict scalar/schema sanitization blocking `{"...

MODULE 11 OF 16 • 14 LIVE + LAB HOURS • 3 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

Production Document Schema Patterns & JSON Schema Validation

Production-grade MongoDB Training engineering module covering Production Document Schema Patterns & JSON Schema Validation with hands-on labs and capstone build.

Objectives: ✓ Extended Reference, Subset, Outlier & Bucket Design Patterns — Purpose: Keep working sets in WiredTiger RAM and avoid unbounded array growth traps. • ✓ Server-Side $jsonSchema Collection Validation Rules — Purpose: Enforce required fields and data types inside MongoDB regardless of which app writes. • ✓ Polymorphic Collections & Zero-Downtime Schema Versioning — Purpose: Store diverse product attributes cleanly and evolve document versions incrementally.

11.1 Extended Reference, Subset, Outlier & Bucket Design Patterns [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply Extended Reference, Subset, Outlier & Bucket Design Patterns in real MongoDB systems — specifically: Keep working sets in WiredTiger RAM and avoid...

11.2 Server-Side $jsonSchema Collection Validation Rules [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply Server-Side $jsonSchema Collection Validation Rules in real MongoDB systems — specifically: Enforce required fields and data types inside MongoDB...

11.3 Polymorphic Collections & Zero-Downtime Schema Versioning [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply Polymorphic Collections & Zero-Downtime Schema Versioning in real MongoDB systems — specifically: Store diverse product attributes cleanly and evo...

Module 11 Hands-On Capstone Project (100 Marks): Multi-Tenant Catalog & Time-Series Telemetry Data Model — Design, implement, test, and document 'Multi-Tenant Catalog & Time-Series Telemetry Data Model' applying every production concept from Module 11 (Production Document Schema Patterns & JSON S...

MODULE 12 OF 16 • 14 LIVE + LAB HOURS • 3 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

Advanced Aggregation Pipelines, Facets, Window Stages & Atlas Search

Production-grade MongoDB Training engineering module covering Advanced Aggregation Pipelines, Facets, Window Stages & Atlas Search with hands-on labs and capstone build.

Objectives: ✓ Multi-Stage $match, $lookup, $facet & $setWindowFields — Purpose: Compute filtered product pages, brand counts, and price buckets in one fast query. • ✓ Pipeline Optimization: Early $match/$project & Index Pushdown — Purpose: Ensure aggregation pipelines leverage B-Tree indexes instead of scanning every BSON doc. • ✓ Text/Fuzzy Search, Geospatial 2dsphere Queries & Materialized $merge — Purpose: Power store-locator radius search and pre-aggregated hourly KPI collections.

12.1 Multi-Stage $match, $lookup, $facet & $setWindowFields [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply Multi-Stage $match, $lookup, $facet & $setWindowFields in real MongoDB systems — specifically: Compute filtered product pages, brand counts, and p...

12.2 Pipeline Optimization: Early $match/$project & Index Pushdown [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply Pipeline Optimization: Early $match/$project & Index Pushdown in real MongoDB systems — specifically: Ensure aggregation pipelines leverage B-Tree...

12.3 Text/Fuzzy Search, Geospatial 2dsphere Queries & Materialized $merge [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply Text/Fuzzy Search, Geospatial 2dsphere Queries & Materialized $merge in real MongoDB systems — specifically: Power store-locator radius search and...

Module 12 Hands-On Capstone Project (100 Marks): Single-Roundtrip E-Commerce Faceted Search & Revenue Pipeline — Design, implement, test, and document 'Single-Roundtrip E-Commerce Faceted Search & Revenue Pipeline' applying every production concept from Module 12 (Advanced Aggregation Pipelines, Facets...

MODULE 13 OF 16 • 14 LIVE + LAB HOURS • 3 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

ESR Compound Indexing, WiredTiger Profiling & Multi-Doc Transactions

Production-grade MongoDB Training engineering module covering ESR Compound Indexing, WiredTiger Profiling & Multi-Doc Transactions with hands-on labs and capstone build.

Objectives: ✓ Equality–Sort–Range (ESR) Rule, Partial & TTL Expiring Indexes — Purpose: Eliminate 32MB in-memory sort errors and auto-expire OTP/session documents cleanly. • ✓ ExecutionStats Forensics (totalKeysExamined vs nReturned) & Profiler — Purpose: Spot low-selectivity indexes and slow >100ms queries before CPU saturates. • ✓ Multi-Document ACID Transactions, Read/Write Concerns & Change Streams — Purpose: Commit cross-collection ledger updates atomically and stream real-time CDC events.

13.1 Equality–Sort–Range (ESR) Rule, Partial & TTL Expiring Indexes [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply Equality–Sort–Range (ESR) Rule, Partial & TTL Expiring Indexes in real MongoDB systems — specifically: Eliminate 32MB in-memory sort errors and au...

13.2 ExecutionStats Forensics (totalKeysExamined vs nReturned) & Profiler [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply ExecutionStats Forensics (totalKeysExamined vs nReturned) & Profiler in real MongoDB systems — specifically: Spot low-selectivity indexes and slow...

13.3 Multi-Document ACID Transactions, Read/Write Concerns & Change Streams [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply Multi-Document ACID Transactions, Read/Write Concerns & Change Streams in real MongoDB systems — specifically: Commit cross-collection ledger upda...

Module 13 Hands-On Capstone Project (100 Marks): Index Audit & ACID Multi-Document Escrow Settlement Engine — Design, implement, test, and document 'Index Audit & ACID Multi-Document Escrow Settlement Engine' applying every production concept from Module 13 (ESR Compound Indexing, WiredTiger Profili...

MODULE 14 OF 16 • 14 LIVE + LAB HOURS • 3 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

Replica Set High Availability, Sharding Keys, RBAC & Backup Ops

Production-grade MongoDB Training engineering module covering Replica Set High Availability, Sharding Keys, RBAC & Backup Ops with hands-on labs and capstone build.

Objectives: ✓ Replica Set Elections, Oplog Sizing & Retryable Reads/Writes — Purpose: Survive primary node hardware crashes with automatic ~2-second failover. • ✓ Shard Key Selection (Cardinality, Frequency & Monotonic Pitfalls) — Purpose: Avoid jumbo chunks and hot-shard bottlenecks when scaling past terabytes. • ✓ SCRAM/TLS Encryption, Least-Privilege Roles & Point-in-Time Restore — Purpose: Lock down production clusters against unauthorized network or credential access.

14.1 Replica Set Elections, Oplog Sizing & Retryable Reads/Writes [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply Replica Set Elections, Oplog Sizing & Retryable Reads/Writes in real MongoDB systems — specifically: Survive primary node hardware crashes with au...

14.2 Shard Key Selection (Cardinality, Frequency & Monotonic Pitfalls) [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply Shard Key Selection (Cardinality, Frequency & Monotonic Pitfalls) in real MongoDB systems — specifically: Avoid jumbo chunks and hot-shard bottlen...

14.3 SCRAM/TLS Encryption, Least-Privilege Roles & Point-in-Time Restore [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply SCRAM/TLS Encryption, Least-Privilege Roles & Point-in-Time Restore in real MongoDB systems — specifically: Lock down production clusters against...

Module 14 Hands-On Capstone Project (100 Marks): Horizontally Sharded Multi-Region MongoDB Cluster Blueprint — Design, implement, test, and document 'Horizontally Sharded Multi-Region MongoDB Cluster Blueprint' applying every production concept from Module 14 (Replica Set High Availability, Sharding...

MODULE 15 OF 16 • 14 LIVE + LAB HOURS • 3 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

MongoDB Schema Governance, Time-Series / Bucketing & Change Streams

Self-explanatory list-and-bullet production mastery module covering MongoDB Schema Governance, Time-Series / Bucketing & Change Streams in MongoDB Training, paired with hands-on labs, capstone repository, Topic Mock Tests, and Spoken Gemini AI Viva.

Objectives: ✓ JSON Schema Validation Rules, Polymorphic Patterns & Avoiding Unbounded Arrays — Purpose: Enforce document data quality at the database layer while preventing 16MB BSON growth traps. • ✓ Extended Reference, Subset & Bucket Patterns for High-Volume Workloads — Purpose: Cut WiredTiger working-set RAM usage and disk IOPS on activity feeds and telemetry streams. • ✓ Resumable MongoDB Change Streams & Multi-Document ACID Transaction Boundaries — Purpose: React to live data mutations reliably using resume tokens and keep financial writes atomic. • ✓ Build, verify, and document 'Schema-Validated IoT/Telemetry Bucket Store & Real-Time Change-Stream Consumer' with clean Git commits and edge-case protection. • ✓ Clear 100% of Topic Timed Mock Tests and Spoken Voice + Google Gemini AI Technical Interviews.

15.1 JSON Schema Validation Rules, Polymorphic Patterns & Avoiding Unbounded Arrays [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply JSON Schema Validation Rules, Polymorphic Patterns & Avoiding Unbounded Arrays in real MongoDB systems — specifically: Enforce document data quali...

15.2 Extended Reference, Subset & Bucket Patterns for High-Volume Workloads [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply Extended Reference, Subset & Bucket Patterns for High-Volume Workloads in real MongoDB systems — specifically: Cut WiredTiger working-set RAM usag...

15.3 Resumable MongoDB Change Streams & Multi-Document ACID Transaction Boundaries [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply Resumable MongoDB Change Streams & Multi-Document ACID Transaction Boundaries in real MongoDB systems — specifically: React to live data mutations...

Module 15 Hands-On Capstone Project (100 Marks): Schema-Validated IoT/Telemetry Bucket Store & Real-Time Change-Stream Consumer — Architect, implement, test, and document 'Schema-Validated IoT/Telemetry Bucket Store & Real-Time Change-Stream Consumer' applying every bullet-wise engineering concept from Module 15 (Mongo...

MODULE 16 OF 16 • 14 LIVE + LAB HOURS • 3 TOPICS • 1 CAPSTONE PROJECT (100 MARKS)

MongoDB Atlas/Production Security, ESR Index Audits & NoSQL System Design

Self-explanatory list-and-bullet production mastery module covering MongoDB Atlas/Production Security, ESR Index Audits & NoSQL System Design in MongoDB Training, paired with hands-on labs, capstone repository, Topic Mock Tests, and Spoken Gemini AI Viva.

Objectives: ✓ Equality-Sort-Range (ESR) Index Rule, Covered Queries & COLLSCAN Elimination — Purpose: Turn multi-second collection scans into single-millisecond IXSCAN lookups verified via explain("executionStats"). • ✓ Shard Key Cardinality/Frequency Analysis, Hotspot Prevention & Backup Restore — Purpose: Choose shard keys that distribute writes evenly without scatter-gather query penalties. • ✓ SQL vs MongoDB Trade-Offs & Real-World NoSQL Data Modeling Interviews — Purpose: Justify embedding vs referencing and consistency/read-concern choices in technical interviews. • ✓ Build, verify, and document 'ESR Compound Index Optimizer, Profiler Audit & Sharding Key Blueprint' with clean Git commits and edge-case protection. • ✓ Clear 100% of Topic Timed Mock Tests and Spoken Voice + Google Gemini AI Technical Interviews.

16.1 Equality-Sort-Range (ESR) Index Rule, Covered Queries & COLLSCAN Elimination [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply Equality-Sort-Range (ESR) Index Rule, Covered Queries & COLLSCAN Elimination in real MongoDB systems — specifically: Turn multi-second collection...

16.2 Shard Key Cardinality/Frequency Analysis, Hotspot Prevention & Backup Restore [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply Shard Key Cardinality/Frequency Analysis, Hotspot Prevention & Backup Restore in real MongoDB systems — specifically: Choose shard keys that distr...

16.3 SQL vs MongoDB Trade-Offs & Real-World NoSQL Data Modeling Interviews [65 mins • Assignment (100M) • Hands-On Lab (50M) • 2 Mock Sets • 2 AI Vivas]

Purpose: To understand and apply SQL vs MongoDB Trade-Offs & Real-World NoSQL Data Modeling Interviews in real MongoDB systems — specifically: Justify embedding vs referencing and...

Module 16 Hands-On Capstone Project (100 Marks): ESR Compound Index Optimizer, Profiler Audit & Sharding Key Blueprint — Architect, implement, test, and document 'ESR Compound Index Optimizer, Profiler Audit & Sharding Key Blueprint' applying every bullet-wise engineering concept from Module 16 (MongoDB Atlas/...

3. LIVE CLIENT PRODUCTION PROJECT, GLOBAL CERTIFICATIONS & CAREER SCOPE

End-of-Course Live Client Production Project + Official Internship Confirmation Letter

Project: CatalogSphere Polymorphic Product Catalog, Search & Real-Time Event Store • **Confirmed Role:** MongoDB NoSQL Data Modeling & Aggregation Intern

Domain & Production Infra: Omnichannel Marketplace & IoT Telemetry Client (MongoDB 7 Replica Set + Atlas Search + Compound/TTL Indexes + Change Streams)

- Modeled high-read polymorphic product documents using embedding vs referencing patterns and JSON Schema validation.
- Built multi-stage Aggregation Pipelines ($match, $lookup, $facet, $group) powering sub-40ms storefront facets.
- Configured compound, partial, text, and TTL indexes verified via explain("executionStats") plans.
- Implemented multi-document ACID transactions and Change Stream listeners on a live MongoDB replica set.

Aligned Global Vendor Certifications: **C100DEV** (MongoDB Certified Associate Developer) • **C100DBA** (MongoDB Certified Associate Database Administrator)

Target Job Roles: MongoDB / NoSQL Developer, Backend Data Layer Engineer, MERN Database Specialist, Associate Cloud Data Engineer

Realistic Annual CTC Scope: **Fresher / Entry-Level (0–1 Yr):** ₹4.5 LPA – ₹8.2 LPA **Mid-Level NoSQL / Backend Engineer (1–3 Yrs):** ₹9.0 LPA – ₹17.5 LPA **Senior Full-Stack / Data Platform Architect (3–6+ Yrs):** ₹18.0 LPA – ₹34.0+ LPA

MockAttempt Academy Pvt. Ltd. • Email: support@mockattempt.com • Phone/WhatsApp: +91 98765 43210 • Web: https://www.mockattempt.com/courses/mongodb-training

Try 2 Free Topic-Wise 50-MCQ Mock Tests & Spoken Gemini AI Interviews daily on every topic above before enrolling.