MongoDB Intensive Project: University Management System

This project will guide you step by step through building a University Management System backend using MongoDB. You will design schemas, perform CRUD operations, use queries, aggregation, indexing, transactions, and advanced features.

■ Collections & Schemas

```
1. Students
{
  "_id": ObjectId,
  "name": "Alice Johnson",
  "email": "alice@example.com",
  "age": 22,
  "courses": [ObjectId],
  "enrolledDate": ISODate
}

2. Courses
{
  "_id": ObjectId,
  "title": "Database Systems",
  "code": "CS301",
  "credits": 3,
  "instructor": ObjectId,
  "students": [ObjectId]
}

3. Instructors
{
  "_id": ObjectId,
  "name": "Dr. Mark Smith",
  "email": "mark.smith@example.com",
  "department": "Computer Science"
}

4. Departments
{
  "_id": ObjectId,
  "name": "Computer Science",
  "building": "Engineering Block A"
}

5. Grades
{
  "_id": ObjectId,
  "studentId": ObjectId,
  "courseId": ObjectId,
  "grade": "A",
  "semester": "Fall 2025"
}
```

■ Features & MongoDB Topics

```
■ CRUD Basics
Insert a student:
db.students.insertOne({ name: "Alice", email: "alice@example.com", age: 22 })

■ Query Operators
Find students older than 20 in CS dept:
db.students.find({ age: { $gt: 20 }, department: "Computer Science" })

■ Aggregation Pipelines
Calculate average grade per course:
db.grades.aggregate([
  { $group: { _id: "$courseId", avgGrade: { $avg: "$grade" } } }
])

■ Indexing & Performance
db.students.createIndex({ email: 1 }, { unique: true })

■ Transactions
Use session.startTransaction() to ensure atomicity.

■ Text Search
db.courses.createIndex({ title: "text" })
db.courses.find({ $text: { $search: "Database" } })
```

■ Exercises

```
1. Insert 20 students, 5 instructors, 10 courses.
2. Write a query to list all courses a student is enrolled in ($lookup).
3. Find all students who failed a course (grade = "F").
4. Aggregate total students in each department.
5. Query top 3 courses with most enrollments.
6. Add index on grades.courseId and analyze speed.
7. Write a transaction for course enrollment.
8. Export students data with mongoexport.
9. Backup & restore using mongodump/mongorestore.
10. Deploy MongoDB to Atlas and connect to Node.js.
```

■ Stretch Goal

Build a frontend dashboard (React/Flutter) where: - Admin adds students & courses. - Students log in and see their grades. - Instructors see enrolled students & grade distributions.