## **Slide Deck Outline: Resource Upskilling Portal – Capstone Project**

[Slide Deck Outline: Resource Upskilling Portal – Capstone Project 1](#_Toc388028018)

[Slide 1: Cover Slide 1](#_Toc923578942)

[Slide 2: Project Overview 1](#_Toc356360522)

[Slide 3: Architecture Diagram 1](#_Toc1126303886)

[Slide 4: Entity Relationship Diagram (ERD) 2](#_Toc1736952543)

[Slide 5: Process Flow – Skill Management & Material Browsing 2](#_Toc696532031)

[Slide 6: Process Flow – Registration & Self-Assessment 2](#_Toc1705284921)

[Slide 7: Process Flow – Learning Journey Logging 2](#_Toc909550578)

[Slide 8: Process Flow – MCQ Test Generation & Attempt 2](#_Toc1657063581)

[Slide 9: GitHub Repository 3](#_Toc201430600)

[Slide 10: Demo Plan 3](#_Toc578058189)

[Slide 11: Feature Summary Table 3](#_Toc1812310650)

[Slide 12: Challenges and Learnings 3](#_Toc1902675869)

[Slide 13: Next Steps / Roadmap 4](#_Toc934104517)

[Slide 14: Thank You 4](#_Toc1811548647)

### **Slide 1: Cover Slide**

* **Title**: Resource Upskilling Portal – Monolithic Architecture
* **Subtitle**: Capstone Project Demo
* **Team Members**: Durga, Rakesh, Anshu, Jyothi, Rohan, Dileep
* **Date**

### **Slide 2: Project Overview**

* **Objective**:  
   To build an integrated portal where users can register, self-assess, track learning progress, and take skill-based MCQ assessments.
* **Tech Stack**:
  + Spring Boot (Monolith)
  + REST APIs
  + MySQL / H2
  + Swagger
* **Modules**:
  + Skill & Knowledgebase Management
  + Learning Journey & Logs
  + MCQ Testing Engine
  + User Registration & Self-Assessment

### **Slide 3: Architecture Diagram**

* Monolithic layered design:
  + Controller → Service → Repository → DB
* DB interactions
* Integrations: Swagger UI
* Tools: Lucidchart / Draw.io

### **Slide 4: Entity Relationship Diagram (ERD)**

* Key Entities:
  + **Skills**, **SkillAreas**
  + **Users**, **LearningLogs**, **Bookmarks**, **SelfAssessment**
  + **MCQQuestion**, **Option**, **Test**, **TestTemplate**, **TestResult**
* Show relationships across learning and testing flows

### **Slide 5: Process Flow – Skill Management & Material Browsing**

* Admin creates Skill + Areas
* Tags content → Content searchable by users
* User filters by skill, area, type

### **Slide 6: Process Flow – Registration & Self-Assessment**

* New user registers → Fills profile
* Selects skills → Self-assesses (Beginner / Developer / Architect)

### **Slide 7: Process Flow – Learning Journey Logging**

* User logs daily learning under a skill area
* View timeline of logs
* Bookmark content

### **Slide 8: Process Flow – MCQ Test Generation & Attempt**

* SME defines template: area + # of questions
* System generates test from question bank
* User attempts test → auto-evaluation → view score

### **Slide 9: GitHub Repository**

* Single repo: [github.com/Mmind007/MCQ](https://github.com/Mmind007/MCQ)
* Contains:
  + All modules (controllers, services, entities)
  + data.sql for sample test data
  + README.md with setup and usage
  + REST endpoints

### **Slide 10: Demo Plan**

Steps to demonstrate:

* Admin: Create skill, area, upload content
* User: Register → Search content → Log progress
* User: Take MCQ test → View result
* SME: Create test template → System generates test

### **Slide 11: Feature Summary Table**

|  |  |  |  |
| --- | --- | --- | --- |
| **Epic** | **Priority** | **Status** | **Assigned** |
| Skill Mgmt | High | Done | Durga |
| Learning Logs | High | In Progress | Rohan |
| MCQ Testing | High | Done | Rakesh |
| Registration | High | Done | Dileep |

### **Slide 12: Challenges and Learnings**

* Designing extensible test model
* Dynamic test generation
* Managing learning logs and assessments
* Coordinating workflows between admin, SME, user

### **Slide 13: Next Steps / Roadmap**

* User dashboards and progress reports
* UI integration (React or Thymeleaf)
* Role-based access control
* Deployment to Docker/AWS

### **Slide 14: Thank You**

* Questions?
* GitHub URL

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***