EC2 → RDS Integration Lab

1) What was built (in plain language)

I built a basic but real AWS application setup with these parts:

- **EC2 instance (app server):** runs the web app.
- **RDS MySQL database:** stores the application data.
- **VPC + Security Groups:** controls network traffic between app and database.
- **IAM Role on EC2:** gives the app secure permission to read secrets.
- **AWS Secrets Manager:** stores DB username/password securely.
- **Simple app endpoints (`/init`, `/add`, `/list`):**
  - `/init` creates the table
  - `/add` inserts data
  - `/list` reads data back

So the full data flow is:

User request → EC2 app → gets DB secret securely → connects to RDS → reads/writes data → returns result

---

2) What this proves technically

This lab proves the core enterprise pattern works end-to-end:

1. App is running on compute (**EC2**).
2. Database is running as managed service (**RDS**).
3. App can securely authenticate to DB (via **Secrets Manager + IAM role**).
4. Network path is locked down (**RDS allows only EC2 SG on port 3306**, not open internet).
5. Data persists in DB and can be retrieved by the app.

In short: **secure application-to-database integration is working**.

---

3) Business benefits

A) Stronger security

- No hardcoded DB passwords in code or AMIs.

- DB is private (not publicly accessible).
- Least-privilege access reduces risk of lateral movement.

B) Better reliability

- App and DB are separated (stateless compute + stateful DB).
- Data survives app restart/redeploy.
- Easier to troubleshoot using clear layers (app, IAM, network, DB).

C) Better operations

- RDS reduces maintenance burden (managed backups, patching options, monitoring integrations).
- Clear CLI-based validation makes support and audits easier.
- Pattern is reusable for internal tools, APIs, and production services.

D) Workforce/job relevance

- This is a real-world architecture commonly used in enterprise AWS workloads.
- Demonstrates practical skills in Cloud, DevOps, and Security.
- Directly maps to interview and on-the-job scenarios.

---

4) Verification evidence completed

I validated the solution using AWS CLI, including:

- EC2 running + reachable
- IAM instance profile attached to EC2
- RDS available with endpoint on 3306
- Security group allows DB access **from EC2 SG only**
- Secret retrieval from Secrets Manager
- DB connectivity test from EC2
- App-level proof: `/init`, `/add`, `/list` works and data persists

This confirms both **infrastructure** and **application** were functioning correctly.

---

5) Simple Summary

I implemented a secure AWS app pattern where an EC2-hosted application connects to a private RDS MySQL database using IAM role–based access to Secrets Manager (no hardcoded

credentials).
I verified compute, database, networking, identity, and data flow end-to-end through CLI checks and app endpoint tests.
This build demonstrates production-relevant cloud architecture, secure design practices, and practical troubleshooting capability.