|  |  |  |  |
| --- | --- | --- | --- |
| **Day** | **Feature** | **Description** | **Skills You'll Master** |
| **Day 1** | **Advanced JWT Refresh Flow + Rotation** | Use short-lived access tokens + refresh tokens with rotation & blacklist | Security best practices, token lifecycle control |
| **Day 2** | **Rate Limiting with Redis (Sliding Window)** | Build a scalable rate limiter using Redis and middleware | Throttling, Redis pipelines |
| **Day 3** | **Role-Based Access Control (RBAC)** | Implement admin, user, moderator level access with route protection | Auth logic, clean middleware design |
| **Day 4** | **Background Jobs with BullMQ (Redis)** | Queue emails, cleanup tasks, and background processing jobs | Async architecture, queue management |
| **Day 5** | **Real-Time Backend with Socket.IO + Redis Pub/Sub** | Build chat or notification backend using events and pub/sub | Real-time communication, horizontal scaling basics |
| **Day 6** | **Database Transactions**  **(ACID) with Mongoose or**  **SQL** | Perform multi-step DB operations with rollback on failure | Data consistency, transaction handling |
| **Day 7** | **Caching with Redis for Expensive Queries** | Cache frequent queries with TTL and fallback mechanism | Read/write cache logic, performance boost |
| **Day 8** | **Secure File Upload with Validation + S3** | Upload, validate, and store files in AWS S3 with signed URLs | Cloud integration, file safety |
| **Day 9** | **OAuth2 with**  **Google/GitHub Login** | Securely log in users using social login flows | Auth standards, token exchange flow |
| **Day 10** | **Multi-Versioned API Strategy (v1, v2)** | Build /api/v1 and /api/v2 routes with deprecation notices | Future-proofing, scalable API architecture |

10 Day **Challenge**

**Day 1 : 12:00 am to 12:00am ~ Done**

🔐 **Advanced JWT Auth API – Endpoint List**

|  |  |  |
| --- | --- | --- |
| **Method** | **Endpoint** | **Purpose** |
| **POST** | /api/auth/register | Register a new user (creates account) |
| **POST** | /api/auth/login | Log in user, returns Access + Refresh tokens |
| **POST** | /api/auth/refresh | Use refresh token to get new access + refresh tokens (rotation) |
| **POST** | /api/auth/logout | Invalidate refresh token (adds to blacklist) |
| **GET** | /api/protected | Example of a protected route (needs access token) |