# Chat Backend Project Checkpoints

## 1. Project Initialization

* 1.1 Repository setup on GitHub with README & folder scaffold
* 1.2 Local dev environment: Go modules, Dockerfiles, basic folder structure

## 2. Authentication Service

* 2.1 Implement JWT-based login & token validation
* 2.2 Publish /docs/auth-api.md and /docs/auth-integrations.md

## 3. Chat Service Core

* 3.1 REST API for room management & history retrieval
* 3.2 WebSocket endpoint for real-time messaging & presence
* 3.3 Publish /docs/chat-protocol.md

## 4. Persistence & Messaging

* 4.1 Design and provision database schema (DynamoDB or PostgreSQL)
* 4.2 Integrate Redis Streams (or Kafka) and document in /docs/broker-setup.md

## 5. Testing & CI/CD

* 5.1 Write unit + integration tests (/docs/testing.md)
* 5.2 Configure GitHub Actions to build, lint, test, and Docker-publish (/docs/ci-cd.md)

## 6. Infrastructure as Code & Deployment

* 6.1 Terraform configs for AWS resources; doc in /docs/deployment.md
* 6.2 Kubernetes manifests for services, ingress, autoscaling; doc in /docs/k8s.md

## 7. Documentation Hosting

* 7.1 Push /docs to gh-pages branch and enable GitHub Pages
* 7.2 Publish /docs/hosting.md with hosting instructions

## 8. Monitoring & Alerting

* 8.1 Instrument code with Prometheus metrics; doc /docs/monitoring.md
* 8.2 Create Grafana dashboards + alert rules; write /docs/runbooks.md

## 9. Final Polish & Review

* 9.1 Proof-read all docs, update diagrams, fix any gaps
* 9.2 Run end-to-end demo and gather feedback

## 10. Resume-Ready Impact Statement

* 10.1 Refine the one-liner:  
  “Designed and implemented a horizontally-scalable real-time chat backend in Go… achieving sub-100 ms latency and 99.9% uptime.”