Architecture Diagrams Summary - Reporting System

1. Component Diagram

Purpose: Illustrates the logical structure of the system by showing main modules and their interactions.

Main Points:

- Inbound Adapters: REST API, Scheduler, Kafka Listener trigger report requests.
- Core Domain: ReportConsumer orchestrates report creation and data retrieval.
- Ports: Abstractions for external dependencies (Kafka, PDF generator, storage, notification, data providers, database).
- Outbound Adapters: Implementations for Kafka broker, PDF generation API, S3 storage, notification services, data sources (ElasticSearch, MDM, Bloxx), databases (Report DB, Job Config DB), and Redis cache.

2. State Machine Diagram

Purpose: Shows the lifecycle and status changes of a report entity throughout processing.

Main Points:

- Report transitions through states: Requested -> In Progress -> Generated -> Stored -> Notified -> Completed.
- Failure transitions to Failed state from various processing steps.
- Enables tracking and managing report status at runtime.

3. Data Flow Diagram (DFD)

Purpose: Represents the flow of data between processes, data stores, and external entities.

Main Points:

- Data flows from Users/Scheduler/Kafka -> ReportConsumer -> external data sources (ElasticSearch, MDM, Bloxx).
- Generated PDF flows to S3 storage.
- Metadata and report status stored in Report DB; cache handled by Redis.
- Notifications sent to users; completion events published on Kafka.

4. Use Case Diagram

Architecture Diagrams Summary - Reporting System

Purpose: Identifies actors interacting with the system and their available use cases.

Main Points:

- Actors: User, Admin, Scheduler, Kafka Event Source.
- Use cases include: Generate report, check status, store report, notify user, publish report event, and manage scheduled jobs.
- Highlights human vs system-triggered interactions.

5. Activity Diagram

Purpose: Visualizes the step-by-step workflow for report generation, including decision points.

Main Points:

- Workflow: Receive report request -> Validate -> Fetch data -> Generate PDF -> Store PDF -> Publish event -> Notify user -> End.
- Explicit success/failure paths at each stage.
- Shows flow control and error handling.

6. Deployment Diagram

Purpose: Maps the physical deployment of system components and their infrastructure environment.

Main Points:

- Nodes representing user clients, application servers, messaging brokers, external APIs, storage, and databases.
- Communication paths (e.g., HTTP calls, Kafka events) between components.
- Separation between application logic, infrastructure services, and data persistence layers.

7. Sequence Diagram

Purpose: Details the dynamic interactions between components during a report generation request.

Main Points:

- Actors: User/Scheduler/Kafka event initiates process.
- Components: REST API, Scheduler, Kafka Listener, Report Consumer, external data providers, PDF generation service, storage service, notification service, Kafka broker.
- Stepwise interactions including request validation, data fetching, PDF generation, storage, event publishing,

and user notification.

- Includes error handling and retry triggers.

Summary Table

Diagram Type	Main Purpose	Key Highlights
Component Diagram	Logical system structure	Core domain, ports/adapters, inbound/outbound layers
State Machine	Report lifecycle states	Requested -> In Progress -> Generated -> Completed/Failed
Data Flow Diagram	Data movement and processing flow	Data retrieval, PDF storage, notifications
Use Case Diagram	Actors and their interactions	User, Admin, Scheduler, Kafka; Generate, Notify
Activity Diagram	Workflow steps and decision points	Validate, fetch data, generate PDF, notify
Deployment Diagram	Physical deployment environment	App servers, Kafka, storage, external APIs
Sequence Diagram	Detailed runtime interactions	Validation, data fetch, PDF gen, store, notify