Q3: Business Rules and Constraints:

1. User Authentication and Authorization

- **Rule**: Users must register with email verification and authenticate to create, edit, and distribute forms.
- **Constraint**: User credentials (passwords) must be securely stored (hashed).
- **Constraint**: Access to forms and responses is controlled by permissions. A user can only access forms and responses for which they have the appropriate permissions (READ, WRITE, ADMIN).
- **Rule**: Authorization is verified using JWT tokens.
- **Rule**: The document creator (ownerUserId) automatically receives ADMIN permissions on their created documents.

2. Form Creation and Editing

- **Rule**: A user must be authorized to create a new form and edit existing form.
- **Constraint**: Each form must have a unique identifier (documentId).
- **Rule**: Forms can contain multiple types of questions (multiple choice, short answer, rating scale, etc.)
- **Constraint**: Form metadata (title, description, questions, etc.) must be stored.
- **Rule**: Forms can be customized (form title, description, color themes, logo addition, etc.).
- **Constraint**: Form html metadata (form title, description, color themes, logo addition, questions, etc.) must be stored.

3. Document Collaboration

- **Rule**: Multiple users can work on the same form simultaneously in real-time using CRDT operation.
- **Constraint**: Concurrent form edits are resolved using CRDTs to ensure consistency.
- **Constraint**: Form versions are tracked to maintain history and ensure consistent distribution.
- **Rule**: Client operations are summarized and transmitted based on hybrid approaches of event parsing:
    - When the user pauses typing (after 200ms)
    - At regular intervals (every 2 seconds)
    - After reaching a certain character threshold (20 characters)
- **Rule**: CRDT operations must be processed in causal order using version vectors.
- **Constraint**: CRDT operations must be idempotent to prevent duplicate application of the same changes.
- **Constraint**: A distributed lock must be acquired before writing CRDT operations to MongoDB to prevent race conditions.
- **Rule**: WebSocket connections must maintain sticky sessions for consistent real-time communication.

4. Document Versioning & History

- **Rule**: Each document modification generates a new document version with an incremental version number.
- **Constraint**: Once created, a document version cannot be modified (immutable versions).
- **Rule**: The Kafka consumer must periodically create snapshots of document states.
- **Constraint**: Form distributions must be linked to *document_versions* entity to ensure consistency.

5. Document Distribution

- **Rule**: Users can distribute forms via different channels (link sharing, email, embedding in websites, etc.) and details stored in the *document_distributions* entity.
- **Rule**: Users can have the option to collect responses anonymously or require respondent identification.
- **Constraint**: A form distribution is associated with a specific version of the form.
- **Constraint**: Distribution links should be unique.

6. Response Collection

- **Rule**: The system collects responses to distributed forms.
- **Rule**: Responses must be validated against the form structure defined in the document version.
- **Constraint**: Response data is stored in a flexible JSON format to accommodate different question types.
- **Constraint**: Responses are associated with a specific version of the form.
- **Constraint**: When anonymous responses are disabled, respondents must provide identifying information.
- **Constraint**: Once submitted, a Response cannot be modified (response immutability).

7. Data Consistency

- **Constraint**: User data, permissions, and responses in PostgreSQL must adhere to ACID properties.
- **Constraint**: Concurrent form edits must be applied consistently across all users using CRDTs.
- **Constraint**: CRDT operations must be applied in causal order, as determined by version vectors.
- **Constraint**: The system must recover from server failures without data loss, using mechanisms like operation logging and snapshotting.

8. System Constraints

- **Constraint**: The system must be scalable to handle a large number of users and responses.
- **Constraint**: The system must be reliable, with measures for data backup and recovery.
- **Constraint**: User data and form data must be secured to prevent unauthorized access.

9. System Recovery

- **Rule**: The system must track the last processed version vector for each document for recovery purposes.
- **Constraint**: During server recovery, operations must be applied in the correct causal order determined by version vectors.
- **Rule**: In case of Kafka failure, the server must directly retrieve pending operations from MongoDB.
- **Rule**: Recovery procedures must ensure no operations are missed or duplicated.

10. Data Storage

- **Constraint**: Different entity types must be stored in appropriate databases:
    - PostgreSQL: *user, permission, response, respondent_details* (relational data)
    - MongoDB: *documents, crdt_operations, document_versions, document_distributions, last_processed_version* (document data)
- **Constraint**: Response.responseData and RespondentDetails.respondentData must use JSONB format.
- **Rule**: Document snapshots must be cached in Redis for performance optimization.

11. Performance & Scalability

- **Rule**: Requests must be distributed across the server cluster while maintaining session stickiness.
- **Constraint**: The system must operate across distributed server clusters to support high loads.
- **Rule**: Document snapshots must be retrieved from Redis if available before falling back to MongoDB.
- **Constraint**: The system must handle millions of simultaneous users without degradation in performance.