End-to-End Flow Diagrams for Flyway Rollback Framework

1. Application Startup Flow

mermaid

```
flowchart TB
    Start([Application Start]) --> LoadConfig[Load application.yml]
    LoadConfig --> CheckProfile{Check Active Profile}

    CheckProfile -->|local| H2Config[Configure H2 Database]
    CheckProfile -->|mysql| MySQLConfig[Configure MySQL]
    CheckProfile -->|test| TestConfig[Configure Test H2]

    H2Config --> InitDS[Initialize DataSource]
    MySQLConfig --> InitDS
    TestConfig --> InitDS

    InitDS --> InitFlyway[Initialize Flyway]
    InitFlyway --> LoadRollbackConfig[Load Rollback Configuration]

    LoadRollbackConfig --> CheckSnapshot{Snapshot Enabled?}
    CheckSnapshot -->|Yes| CreatePreSnapshot[Create Pre-Migration Snapshot]
    CheckSnapshot -->|No| RunMigrations

    CreatePreSnapshot --> SaveSnapshot[Save Snapshot Metadata]
    SaveSnapshot --> RunMigrations[Execute Flyway Migrations]

    RunMigrations --> ScanMigrations[Scan db/migration folder]
    ScanMigrations --> CheckPending{Pending Migrations?}

    CheckPending -->|Yes| ApplyMigrations[Apply Migrations in Order]
    CheckPending -->|No| StartApp

    ApplyMigrations --> UpdateHistory[Update flyway_schema_history]
    UpdateHistory --> CheckSuccess{Migration Success?}

    CheckSuccess -->|Yes| StartApp[Start Spring Boot Application]
    CheckSuccess -->|No| CheckAutoRollback{Auto-Rollback Enabled?}

    CheckAutoRollback -->|Yes| TriggerRollback[Trigger Automatic Rollback]
    CheckAutoRollback -->|No| FailStart[Fail Application Start]

    TriggerRollback --> RestoreSnapshot[Restore from Snapshot]
    RestoreSnapshot --> FailStart

    StartApp --> InitControllers[Initialize REST Controllers]
    InitControllers --> InitSecurity[Initialize Security]
    InitSecurity --> Ready([Application Ready])

    style Start fill:#90EE90
    style Ready fill:#90EE90
    style FailStart fill:#FFB6C1
```

2. Migration Execution Flow

```mermaid
flowchart TB
    MigrationStart([Migration Process Start]) --> GetCurrentVersion[Get Current DB Version]
    GetCurrentVersion --> QueryHistory[Query flyway_schema_history]

    QueryHistory --> ScanScripts[Scan Migration Scripts]
    ScanScripts --> FilterPending[Filter Pending Migrations]

    FilterPending --> SortByVersion[Sort by Version Number]
    SortByVersion --> ForEachMigration{For Each Migration}

    ForEachMigration --> ReadScript[Read SQL Script]
    ReadScript --> ParseSQL[Parse SQL Statements]

    ParseSQL --> BeginTx[Begin Transaction]
    BeginTx --> ExecuteSQL[Execute SQL Statements]

    ExecuteSQL --> CheckDBType{Database Type?}
    CheckDBType -->|H2| H2Syntax[Use H2 Syntax]
    CheckDBType -->|MySQL| MySQLSyntax[Use MySQL Syntax]

    H2Syntax --> ValidateResult
    MySQLSyntax --> ValidateResult[Validate Execution]

    ValidateResult --> Success{Success?}
    Success -->|Yes| CommitTx[Commit Transaction]
    Success -->|No| RollbackTx[Rollback Transaction]

    CommitTx --> RecordSuccess[Record in Schema History]
    RecordSuccess --> NextMigration{More Migrations?}

    RollbackTx --> RecordFailure[Record Failure]
    RecordFailure --> HandleFailure[Handle Migration Failure]

    NextMigration -->|Yes| ForEachMigration
    NextMigration -->|No| Complete([Migrations Complete])

    HandleFailure --> NotifyError[Notify Error Handler]
    NotifyError --> StopMigration([Stop Migration])

    style MigrationStart fill:#87CEEB
    style Complete fill:#90EE90
    style StopMigration fill:#FFB6C1
```

3. Rollback Execution Flow

mermaid

```
flowchart TB
    RollbackAPI([Rollback API Called]) --> ValidateRequest[Validate Request]
    ValidateRequest --> CheckAuth{Authorized?}

    CheckAuth -->|No| ReturnUnauth[Return 401 Unauthorized]
    CheckAuth -->|Yes| ParseRequest[Parse Target Version]

    ParseRequest --> CheckApproval{Approval Required?}
    CheckApproval -->|Yes| CheckApprovalStatus{Has Approval?}
    CheckApproval -->|No| StartRollback

    CheckApprovalStatus -->|No| RequestApproval[Request Approval]
    CheckApprovalStatus -->|Yes| StartRollback

    RequestApproval --> ReturnPending[Return Approval Pending]

    StartRollback[Start Rollback Process] --> GetCurrentVer[Get Current Version]
    GetCurrentVer --> CreateSnapshot{Create Snapshot?}

    CreateSnapshot -->|Yes| SnapshotProcess[Execute Snapshot Creation]
    CreateSnapshot -->|No| IdentifyScripts

    SnapshotProcess --> SaveMetadata[Save Snapshot Metadata]
    SaveMetadata --> IdentifyScripts[Identify Rollback Scripts]

    IdentifyScripts --> ListVersions[List Versions to Rollback]
    ListVersions --> ForEachVersion{For Each Version}

    ForEachVersion --> CheckScript{Rollback Script Exists?}
    CheckScript -->|No| UseGeneric[Use Generic Rollback]
    CheckScript -->|Yes| LoadScript[Load Ux__rollback.sql]

    LoadScript --> ExecuteRollback[Execute Rollback SQL]
    UseGeneric --> ExecuteRollback

    ExecuteRollback --> ArchiveData{Archive Data?}
    ArchiveData -->|Yes| CreateArchive[Create Archive Tables]
    ArchiveData -->|No| DropObjects

    CreateArchive --> CopyData[Copy Data to Archive]
    CopyData --> DropObjects[Drop Database Objects]

    DropObjects --> UpdateHistory[Update Schema History]
    UpdateHistory --> RemoveEntry[Remove Migration Entry]

    RemoveEntry --> NextVersion{More Versions?}
    NextVersion -->|Yes| ForEachVersion
    NextVersion -->|No| FinalizeRollback

    FinalizeRollback[Finalize Rollback] --> AuditLog[Create Audit Log Entry]
    AuditLog --> ReturnSuccess[Return Success Response]

    style RollbackAPI fill:#FFD700
    style ReturnSuccess fill:#90EE90
    style ReturnUnauth fill:#FFB6C1
    style ReturnPending fill:#FFA500
```

4. Snapshot Creation Flow

```mermaid
flowchart TB
    SnapshotStart([Snapshot Creation Start]) --> GenerateID[Generate Snapshot ID]
    GenerateID --> CreateDir[Create Snapshot Directory]

    CreateDir --> GetTables[Get All Tables List]
    GetTables --> FilterTables[Filter System Tables]

    FilterTables --> ForEachTable{For Each Table}
    ForEachTable --> CheckDB{Database Type?}

    CheckDB -->|H2| H2Export[Export to CSV]
    CheckDB -->|MySQL| MySQLExport[Create Backup Table]

    H2Export --> WriteCSV[Write CSV File]
    MySQLExport --> CreateTable[CREATE TABLE snapshot_x]

    WriteCSV --> RecordTable
    CreateTable --> CopyData[INSERT INTO snapshot_x SELECT * FROM x]
    CopyData --> RecordTable[Record Table Snapshot]

    RecordTable --> NextTable{More Tables?}
    NextTable -->|Yes| ForEachTable
    NextTable -->|No| CreateMetadata

    CreateMetadata[Create Metadata JSON] --> SaveTimestamp[Save Timestamp]
    SaveTimestamp --> SaveVersion[Save Current Version]
    SaveVersion --> SaveTableList[Save Table List]

    SaveTableList --> WriteMetadata[Write metadata.json]
    WriteMetadata --> CompressOption{Compress Enabled?}

    CompressOption -->|Yes| CompressFiles[Compress Snapshot Files]
    CompressOption -->|No| Complete

    CompressFiles --> Complete([Snapshot Complete])

    style SnapshotStart fill:#87CEEB
    style Complete fill:#90EE90
```

5. REST API Request Flow

mermaid

```
flowchart TB
    HTTPRequest([HTTP Request]) --> SecurityFilter[Spring Security Filter]
    SecurityFilter --> CheckEndpoint{Protected Endpoint?}

    CheckEndpoint -->|No| PassThrough
    CheckEndpoint -->|Yes| ValidateAuth[Validate Authentication]

    ValidateAuth --> Authenticated{Authenticated?}
    Authenticated -->|No| Return401[Return 401]
    Authenticated -->|Yes| PassThrough[Pass to Controller]

    PassThrough --> RouteRequest[Route to Controller]
    RouteRequest --> ControllerMethod{Which Controller?}

    ControllerMethod -->|UserController| UserOps[User Operations]
    ControllerMethod -->|RollbackController| RollbackOps[Rollback Operations]
    ControllerMethod -->|Actuator| ActuatorOps[Monitoring Operations]

    UserOps --> UserAction{Action?}
    UserAction -->|GET /users| GetAllUsers[Query All Users]
    UserAction -->|POST /users| CreateUser[Create New User]
    UserAction -->|GET /users/{id}| GetUser[Get Specific User]
    UserAction -->|DELETE /users/{id}| DeleteUser[Delete User]

    RollbackOps --> RollbackAction{Action?}
    RollbackAction -->|POST /execute| ExecuteRollback[Execute Rollback]
    RollbackAction -->|POST /snapshot| CreateSnapshot[Create Snapshot]
    RollbackAction -->|GET /health| CheckHealth[Check Service Health]

    GetAllUsers --> UserRepo[UserRepository.findAll()]
    CreateUser --> ValidateUser[Validate User Data]
    ValidateUser --> UserRepo2[UserRepository.save()]

    ExecuteRollback --> RollbackManager[FlywayRollbackManager]
    CreateSnapshot --> SnapshotManager[SnapshotManager]

    UserRepo --> ReturnJSON
    UserRepo2 --> ReturnJSON
    RollbackManager --> ReturnJSON
    SnapshotManager --> ReturnJSON[Return JSON Response]

    ActuatorOps --> MetricsData[Collect Metrics]
    MetricsData --> ReturnJSON

    style HTTPRequest fill:#FFD700
    style ReturnJSON fill:#90EE90
    style Return401 fill:#FFB6C1
```

6. Database Transaction Flow

mermaid

```
flowchart TB
    Operation([Database Operation]) --> GetConnection[Get DB Connection]
    GetConnection --> CheckPool{Connection Available?}

    CheckPool -->|No| WaitForConnection[Wait for Connection]
    CheckPool -->|Yes| SetAutoCommit[Set AutoCommit = false]

    WaitForConnection --> Timeout{Timeout?}
    Timeout -->|Yes| ThrowException[Throw Timeout Exception]
    Timeout -->|No| CheckPool

    SetAutoCommit --> BeginTransaction[Begin Transaction]
    BeginTransaction --> OperationType{Operation Type?}

    OperationType -->|DDL| DDLOps[DDL Operations]
    OperationType -->|DML| DMLOps[DML Operations]
    OperationType -->|Query| QueryOps[Query Operations]

    DDLOps --> CheckDDLSupport{DDL Transaction Support?}
    CheckDDLSupport -->|MySQL/H2| ExecuteDDL[Execute DDL]
    CheckDDLSupport -->|PostgreSQL| ExecuteDDLTx[Execute in Transaction]

    DMLOps --> PrepareStatement[Prepare Statement]
    PrepareStatement --> BindParams[Bind Parameters]
    BindParams --> ExecuteDML[Execute DML]

    QueryOps --> PrepareQuery[Prepare Query]
    PrepareQuery --> ExecuteQuery[Execute Query]
    ExecuteQuery --> FetchResults[Fetch Results]

    ExecuteDDL --> CheckSuccess
    ExecuteDDLTx --> CheckSuccess
    ExecuteDML --> CheckSuccess
    FetchResults --> CheckSuccess{Success?}

    CheckSuccess -->|Yes| CommitTx[Commit Transaction]
    CheckSuccess -->|No| RollbackTx[Rollback Transaction]

    CommitTx --> ReleaseConnection[Release Connection]
    RollbackTx --> LogError[Log Error]
    LogError --> ReleaseConnection

    ReleaseConnection --> ReturnToPool[Return to Connection Pool]
    ReturnToPool --> Complete([Operation Complete])

    ThrowException --> ErrorHandler[Handle Error]
    ErrorHandler --> Complete

    style Operation fill:#87CEEB
    style Complete fill:#90EE90
    style ThrowException fill:#FFB6C1
```

7. H2 vs MySQL Flow Differences

mermaid

```
flowchart TB
    Start([Database Operation]) --> DetectDB{Detect Database Type}

    DetectDB --> H2Flow[H2 Database Flow]
    DetectDB --> MySQLFlow[MySQL Database Flow]

    H2Flow --> H2Conn[H2 In-Memory Connection]
    H2Conn --> H2Features{Feature}

    H2Features -->|Migration| H2Migration[Direct SQL Execution]
    H2Features -->|Snapshot| H2Snapshot[CSV Export]
    H2Features -->|Console| H2Console[Web Console Available]

    MySQLFlow --> MySQLConn[MySQL Network Connection]
    MySQLConn --> MySQLFeatures{Feature}

    MySQLFeatures -->|Migration| MySQLMigration[Transaction Support]
    MySQLFeatures -->|Snapshot| MySQLSnapshot[Table Duplication]
    MySQLFeatures -->|Console| MySQLClient[MySQL Client Required]

    H2Migration --> H2Syntax[H2 SQL Syntax]
    MySQLMigration --> MySQLSyntax[MySQL SQL Syntax]

    H2Snapshot --> CSVWrite[CSVWRITE Function]
    MySQLSnapshot --> CreateBackup[CREATE TABLE AS SELECT]

    H2Console --> WebUI[http://localhost:8080/h2-console]
    MySQLClient --> CLI[mysql -h localhost -u root]

    H2Syntax --> Execute
    MySQLSyntax --> Execute
    CSVWrite --> StoreSnapshot
    CreateBackup --> StoreSnapshot
    WebUI --> Access
    CLI --> Access[Access Database]

    Execute --> Result([Operation Result])
    StoreSnapshot --> Result
    Access --> Result

    style Start fill:#87CEEB
    style Result fill:#90EE90
```

8. Error Handling and Recovery Flow

mermaid

```mermaid
flowchart TB
    Error([Error Occurs]) --> ErrorType{Error Type?}

    ErrorType -->|Migration Failed| MigrationError[Migration Error Handler]
    ErrorType -->|Rollback Failed| RollbackError[Rollback Error Handler]
    ErrorType -->|Connection Lost| ConnectionError[Connection Error Handler]
    ErrorType -->|Constraint Violation| ConstraintError[Constraint Error Handler]

    MigrationError --> CheckPartial{Partial Migration?}
    CheckPartial -->|Yes| IdentifyState[Identify Current State]
    CheckPartial -->|No| RecordFailure

    IdentifyState --> AttemptRepair[Attempt Repair]
    AttemptRepair --> RepairSuccess{Success?}
    RepairSuccess -->|Yes| ContinueMigration[Continue Migration]
    RepairSuccess -->|No| InitiateRollback[Initiate Rollback]

    RollbackError --> CheckSnapshot{Snapshot Available?}
    CheckSnapshot -->|Yes| RestoreSnapshot[Restore from Snapshot]
    CheckSnapshot -->|No| ManualIntervention[Require Manual Intervention]

    ConnectionError --> RetryLogic{Retry Available?}
    RetryLogic -->|Yes| WaitBackoff[Wait with Backoff]
    RetryLogic -->|No| FailOperation

    WaitBackoff --> RetryOperation[Retry Operation]
    RetryOperation --> RetrySuccess{Success?}
    RetrySuccess -->|Yes| Continue[Continue Operation]
    RetrySuccess -->|No| RetryLogic

    ConstraintError --> IdentifyConstraint[Identify Constraint]
    IdentifyConstraint --> FKViolation{Foreign Key?}
    FKViolation -->|Yes| DisableFK[Disable FK Checks]
    FKViolation -->|No| UniqueViolation[Handle Unique Violation]

    DisableFK --> RetryWithoutFK[Retry Operation]
    RetryWithoutFK --> ReenableFK[Re-enable FK Checks]

    InitiateRollback --> RecordFailure[Record in Audit Log]
    RestoreSnapshot --> RecordFailure
    ManualIntervention --> RecordFailure
    FailOperation --> RecordFailure
    ReenableFK --> RecordFailure
    UniqueViolation --> RecordFailure

    RecordFailure --> NotifyAdmin[Notify Administrator]
    NotifyAdmin --> Complete([Error Handled])

    ContinueMigration --> Success([Recovery Success])
    Continue --> Success

    style Error fill:#FFB6C1
    style Complete fill:#FFA500
    style Success fill:#90EE90
```

Flow Diagram Legend

- 🟢 **Green**: Successful completion states
- 🔴 **Red**: Error or failure states
- 🟡 **Yellow**: API entry points
- 🔵 **Blue**: Process start points
- 🟠 **Orange**: Warning or pending states
- ⬜ **White**: Normal process steps
- 🔷 **Diamond**: Decision points

Key Process Flows Summary

1. Normal Migration Flow

```
Start → Load Config → Initialize Flyway → Create Snapshot → Run Migrations → Update History → Start App
```

2. Rollback Flow

```
API Request → Validate → Create Snapshot → Execute Rollback Scripts → Update History → Audit → Response
```

3. Error Recovery Flow

```
Error → Identify Type → Check Recovery Options → Execute Recovery → Log Result → Notify
```

4. Database Detection Flow

```
Operation → Detect DB Type → Choose Strategy → Execute with DB-Specific Syntax → Return Result
```

These flow diagrams provide a comprehensive view of how the Flyway rollback framework operates from startup to error handling, showing all the decision points and process flows in detail.