Atlas App Sequence Diagrams

1. User Authentication Flow

```
sequenceDiagram
    participant User
    participant Frontend as Atlas Frontend
    participant NextAuth as Next Auth
    participant API as Atlas API
    participant Cognito as AWS Cognito
    participant DB as Database

    Note over User,DB: User Authentication Process

    User->>Frontend: Login with email/password
    activate Frontend
    Frontend->>NextAuth: Submit credentials
    NextAuth->>Cognito: Authenticate user
    activate Cognito
    Cognito-->>NextAuth: Return JWT token
    deactivate Cognito
    NextAuth-->>Frontend: Store auth token in session
    Frontend-->>User: Redirect to dashboard
    deactivate Frontend

    User->>Frontend: Access protected route
    activate Frontend
    Frontend->>API: Request data with auth token
    activate API
    API->>API: JWT middleware validates token
    API->>Cognito: Verify token (if needed)
    Cognito-->>API: Token verification
    API->>DB: Fetch authorized data
    DB-->>API: Return data
    API-->>Frontend: Return authorized data
    deactivate API
    Frontend-->>User: Display content
    deactivate Frontend
```

2. Evaluation Creation and Processing Flow

```
sequenceDiagram
    participant User
    participant Frontend as Atlas Frontend
    participant API as Atlas API Backend
    participant MongoDB as MongoDB
```

```
participant Kafka as Message Queue
participant Worker as Evaluation Worker
participant Contract as Smart Contract
participant Operator as AVS Operator
participant S3 as AWS S3

Note over User,S3: Evaluation Creation and Processing

User->>Frontend: Select model and dataset
activate Frontend
User->>Frontend: Submit evaluation request
Frontend->>API: POST /api/v1/evaluations
activate API
API->>API: Validate request
API->>MongoDB: Store evaluation with PENDING status
MongoDB-->>API: Confirm storage
API->>Kafka: Publish to evaluations topic
Kafka-->>API: Confirm message
API-->>Frontend: Return evaluation ID and status
deactivate API
Frontend-->>User: Show evaluation pending
deactivate Frontend

Kafka-->>Worker: Consume evaluation message
activate Worker
Worker->>MongoDB: Fetch evaluation details
MongoDB-->>Worker: Return evaluation data
Worker->>Contract: Call requestEval()
activate Contract
Contract->>Contract: Assign operator and create task
Contract-->>Worker: Return task ID
deactivate Contract
Worker->>MongoDB: Update with task ID
MongoDB-->>Worker: Confirm update
deactivate Worker

Contract-->>Operator: Assign evaluation task
activate Operator
Note right of Operator: Operator processes evaluation
Operator->>Contract: Submit evaluation results
deactivate Operator

Worker->>Contract: Listen for EvalCompleted events
Contract-->>Worker: EvalCompleted event
activate Worker
Worker->>Worker: Process results
```

```
    Worker->>S3: Store detailed results
    S3-->>Worker: Confirm storage
    Worker->>MongoDB: Update evaluation status to COMPLETED
    MongoDB-->>Worker: Confirm update
    Worker->>API: Notify of completion via WebSocket
    deactivate Worker

    API-->>Frontend: WebSocket event: evaluation complete
    activate Frontend
    Frontend->>API: GET /api/v1/evaluations/{id}
    API->>MongoDB: Fetch evaluation
    MongoDB-->>API: Return evaluation data
    API->>S3: Fetch detailed results
    S3-->>API: Return results data
    API-->>Frontend: Return combined data
    Frontend-->>User: Display evaluation results
    deactivate Frontend
```

3. Dataset Management Flow

```
sequenceDiagram
    participant Admin
    participant Frontend as Atlas Frontend
    participant API as Atlas API
    participant S3 as AWS S3
    participant MongoDB as MongoDB
    participant Registry as Dataset Registry

    Note over Admin,Registry: Dataset Upload and Management

    Admin->>Frontend: Navigate to dataset management
    Admin->>Frontend: Upload dataset files
    activate Frontend
    Frontend->>Frontend: Validate dataset format
    Frontend->>API: POST /api/v1/datasets
    activate API
    API->>API: Validate dataset metadata
    API->>S3: Upload dataset files
    S3-->>API: Confirm upload
    API->>MongoDB: Store dataset metadata
    MongoDB-->>API: Confirm storage
    API->>Registry: Register dataset in registry
    Registry-->>API: Confirm registration
    API-->>Frontend: Return success
    deactivate API
    Frontend-->>Admin: Show success message
```

```
    deactivate Frontend

    User->>Frontend: Browse available datasets
    activate Frontend
    Frontend->>API: GET /api/v1/datasets
    activate API
    API->>MongoDB: Fetch dataset metadata
    MongoDB-->>API: Return datasets
    API-->>Frontend: Return dataset list
    deactivate API
    Frontend-->>User: Display dataset gallery
    deactivate Frontend

    User->>Frontend: View dataset details
    activate Frontend
    Frontend->>API: GET /api/v1/datasets/{id}
    activate API
    API->>MongoDB: Fetch dataset metadata
    MongoDB-->>API: Return dataset
    API->>S3: Get dataset sample
    S3-->>API: Return sample data
    API-->>Frontend: Return combined data
    deactivate API
    Frontend-->>User: Display dataset details
    deactivate Frontend
```

4. Results Viewing and Comparison Flow

```
sequenceDiagram
    participant User
    participant Frontend as Atlas Frontend
    participant API as Atlas API
    participant MongoDB as MongoDB
    participant MariaDB as MariaDB
    participant S3 as AWS S3

    Note over User,S3: Viewing and Comparing Results

    User->>Frontend: Navigate to evaluations
    activate Frontend
    Frontend->>API: GET /api/v1/evaluations
    activate API
    API->>MongoDB: Fetch user's evaluations
    MongoDB-->>API: Return evaluations
    API-->>Frontend: Return evaluation list
    deactivate API
```

```
    Frontend-->>User: Display evaluations
    deactivate Frontend

    User->>Frontend: Select evaluation for details
    activate Frontend
    Frontend->>API: GET /api/v1/evaluations/{id}
    activate API
    API->>MongoDB: Fetch evaluation metadata
    MongoDB-->>API: Return metadata
    API->>S3: Fetch detailed results
    S3-->>API: Return result data
    API-->>Frontend: Return combined data
    deactivate API
    Frontend->>Frontend: Process and visualize results
    Frontend-->>User: Display detailed metrics and charts
    deactivate Frontend

    User->>Frontend: Request model comparison
    activate Frontend
    Frontend->>API: GET /api/v1/models/compare?ids=model1,model2&dataset=X
    activate API
    API->>MariaDB: Fetch aggregate metrics
    MariaDB-->>API: Return metrics
    API->>MongoDB: Fetch evaluations for models
    MongoDB-->>API: Return evaluations
    API-->>Frontend: Return comparison data
    deactivate API
    Frontend->>Frontend: Generate comparison visualizations
    Frontend-->>User: Display model comparison
    deactivate Frontend
```

5. Complete System Architecture

```
flowchart TB
    subgraph "Frontend"
        NextApp[Next.js App]
        ReactQuery[React Query]
        Components[UI Components]
        Hooks[Custom Hooks]
        Auth[Next Auth]
    end

    subgraph "Backend Services"
        API[Atlas API]
        Worker[Evaluation Worker]
        ResultsWorker[Results Worker]
```

```
    Scheduler[Task Scheduler]
end

subgraph "Data Storage"
    MongoDB[(MongoDB)]
    MariaDB[(MariaDB)]
    S3[(AWS S3)]
end

subgraph "Infrastructure"
    Kafka[Kafka Message Queue]
    Cognito[AWS Cognito]
    CDK[AWS CDK Infrastructure]
end

subgraph "Blockchain Integration"
    Contract[Smart Contract]
    AVS[Evaluation AVS]
end

%% Frontend connections
NextApp --> ReactQuery
NextApp --> Components
NextApp --> Hooks
NextApp --> Auth
Auth <--> Cognito
ReactQuery <--> API

%% Backend connections
API <--> MongoDB
API <--> MariaDB
API <--> S3
API <--> Kafka
API <--> Cognito

Kafka --> Worker
Kafka --> ResultsWorker

Worker <--> MongoDB
Worker <--> S3
Worker <--> Contract

ResultsWorker <--> MongoDB
ResultsWorker <--> MariaDB
ResultsWorker <--> S3
```

```
Scheduler --> Worker
Scheduler --> ResultsWorker
Scheduler <--> MongoDB

%% Blockchain connections
Contract <--> AVS

%% Classification styling
classDef frontend fill:#bbf,stroke:#333,stroke-width:1px
classDef backend fill:#bfb,stroke:#333,stroke-width:1px
classDef storage fill:#fbb,stroke:#333,stroke-width:1px
classDef infra fill:#fbf,stroke:#333,stroke-width:1px
classDef blockchain fill:#fdb,stroke:#333,stroke-width:1px

class NextApp,ReactQuery,Components,Hooks,Auth frontend
class API,Worker,ResultsWorker,Scheduler backend
class MongoDB,MariaDB,S3 storage
class Kafka,Cognito,CDK infra
class Contract,AVS blockchain
```