Contoh Lengkap Submit & Retrieve Blob di 0G DA

1. Jalankan DA Client Node (Docker)

```
COMBINED_SERVER_CHAIN_RPC=https://0g-galileo-testnet.drpc.org/
COMBINED_SERVER_PRIVATE_KEY=0xYOUR_PRIVATE_KEY
ENTRANCE_CONTRACT_ADDR=0xE75A073dA5bb7b0eC622170Fd268f35E675a957B
GRPC_SERVER_PORT=51001

docker run --env-file .env -p 51001:51001 0g-da-client
```

2. Install dependensi gRPC untuk Node.js

```
npm install @grpc/grpc-js @grpc/proto-loader
```

3. Submit Data (submit.js)

```
import grpc from "@grpc/grpc-js";
import protoLoader from "@grpc/proto-loader";

const PROTO_PATH = "./proto/da.proto"; // pastikan path sesuai
const packageDef = protoLoader.loadSync(PROTO_PATH, {
  keepCase: true,
  longs: String,
  enums: String,
  defaults: true,
  oneofs: true,
});
const grpcObj = grpc.loadPackageDefinition(packageDef);
const daService = grpcObj.da.DataAvailabilityService;

const client = new daService(
  "localhost:51001", // gRPC endpoint dari DA Client Node
  grpc.credentials.createInsecure()
);

function submitData() {
  const data = Buffer.from("Hello 0G DA via Galileo Testnet!", "utf-8");

  client.SubmitBlob({ data }, (err, response) => {
    if (err) {
      console.error("■ Error submit:", err);
      return;
    }
    console.log("■ Blob submitted successfully");
    console.log("Blob ID:", response.blobId);
    console.log("Commitment:", response.commitment);
  });
}

submitData();
```

4. Retrieve Data (retrieve.js)

```
import grpc from "@grpc/grpc-js";
import protoLoader from "@grpc/proto-loader";

const PROTO_PATH = "./proto/da.proto";
const packageDef = protoLoader.loadSync(PROTO_PATH, {
  keepCase: true,
  longs: String,
```

```
  enums: String,
  defaults: true,
  oneofs: true,
});
const grpcObj = grpc.loadPackageDefinition(packageDef);
const daService = grpcObj.da.DataAvailabilityService;

const client = new daService(
  "localhost:51001",
  grpc.credentials.createInsecure()
);

function getData(blobId) {
  client.GetBlob({ blobId }, (err, response) => {
    if (err) {
      console.error("■ Error retrieve:", err);
      return;
    }
    console.log("■ Retrieved blob data:", Buffer.from(response.data).toString("utf-8"));
  });
}

// ganti dengan blobId dari hasil submit
getData("ISI_BLOB_ID");
```

5. Integrasi ke Smart Contract (Opsional)

```
pragma solidity ^0.8.0;

contract DACommitment {
    mapping(bytes32 => bool) public storedCommitments;

    function storeCommitment(bytes32 commitment) external {
        storedCommitments[commitment] = true;
    }

    function verifyCommitment(bytes32 commitment) external view returns (bool) {
        return storedCommitments[commitment];
    }
}
```