AWS SAM Template Walkthrough for JEDx CAR Service

Top-level metadata
- AWSTemplateFormatVersion / Transform (AWS::Serverless-2016-10-31) — declares this as a SAM template so you can use shorthand (AWS::Serverless::*) and SAM CLI.
- Description — human-readable summary.
- Globals.Function — default settings applied to all Lambda functions unless overridden (here: Timeout: 3s, structured JSON logs).

Parameters
- Prefix — string used to prefix every resource name (e.g., jedx-car-bucket).
- CollectorUrl — URL for an external “collector” API (currently not used).

App grouping & monitoring
- ApplicationResourceGroup (AWS::ResourceGroups::Group) — groups all stack resources for easier discovery/monitoring.
- ApplicationInsightsMonitoring (AWS::ApplicationInsights::Application) — turns on CloudWatch Application Insights for the above resource group (auto-config enabled).

Storage & streams (CAR service)
- JedxCarInputBucket (S3) — landing bucket for incoming files (versioned). Triggers S3ToKinesisFunction on object creation.
- JedxCarBucket (S3) — main application bucket for validated/processed artifacts (versioned).
- JedxCarErrorBucket (S3) — captures validation failures or error payloads (versioned).
- JedxCarInputStream (Kinesis) — stream (ShardCount: 1) carrying records from S3 into the validation pipeline.

Public API (Gateway + Cognito auth)
- JedxCarApi (AWS::Serverless::Api)
- Name/Stage: ${Prefix}-car-api, stage Prod.
- CORS: allows DELETE,GET,HEAD,PUT,POST from any origin with common headers.
- Auth: Cognito User Pool authorizer (JedxUserPoolAuthorizer); Authorization header expected; reauth disabled.

Lambda: API handlers
- JedxCarApiFunction (car_api/app.lambda_handler)
- Event: ANY /car on JedxCarApi (generic endpoint).
- Env: KINESIS_STREAM_NAME, DDB_TABLE_NAME.
- Policies: kinesis:PutRecord to input stream; basic DynamoDB write/query on the table.
- JedxCarRecordApiFunction (car_api/record.lambda_handler)
- Event: ANY /car/record/{senderId}/{object_type}/{RefId}.
- Env: S3_CAR_BUCKET, DDB_TABLE_NAME.
- Policies: S3 read/write on main bucket; DynamoDB put/update/get/query.
- JedxCarRecordsApiFunction (same handler as above)
- Event: ANY /car/records/{senderId} (list multiple records).
- Env/Policies: similar S3 + DynamoDB access.
- JedxCarCollectorSendApiFunction (car_api/collector.lambda_handler)
- Event: ANY /car/collector/send (forwards to external collector).
- Env: S3_CAR_BUCKET, DDB_TABLE_NAME, COLLECTOR_URL.
- Note: COLLECTOR_URL is hard-coded to a specific API URL; you’ve commented out the !Ref CollectorUrl. If you want per-env flexibility, switch to the parameter (uncomment the !Ref and remove the hardcoded lines).

- Policies: S3 read/write on main bucket; DynamoDB put/update/get/query.
- JedxCarUserApiFunction (car_api/user.lambda_handler)
- Event: POST /car/login (auth/user ops).
- Env: DDB_TABLE_NAME.
- Policies: DynamoDB put/update/get/query.

Lambda: data pipeline
- S3ToKinesisFunction (s3_to_kinesis_lambda/app.lambda_handler)
- Trigger: s3:ObjectCreated:* on JedxCarInputBucket.
- Purpose: reads new S3 objects and pushes a record into JedxCarInputStream.
- Policies: s3:GetObject on input bucket; kinesis:PutRecord on stream.
- CarValidationFunction (car_validation_lambda/app.lambda_handler)
- Trigger: Kinesis (stream = JedxCarInputStream, BatchSize: 10, StartingPosition: LATEST).
- Purpose: validates incoming records and writes results.
- Env: S3_CAR_BUCKET, S3_CAR_ERROR_BUCKET, DDB_TABLE_NAME.
- EventInvokeConfig: MaximumRetryAttempts: 0 (no async retries—bad records won’t be retried).
- Policies: s3:PutObject to the main and error buckets; DynamoDB PutItem to persist validation outcomes.

Database
- JedxCarTable (DynamoDB)
- Name: ${Prefix}-car-table.
- Keys: pk (HASH) + sk (RANGE) — typical single-table design for flexible access patterns.
- Billing: PAY_PER_REQUEST (on-demand).

Outputs
- JedxCollectorFunction — ARN of the primary API Lambda (for quick reference/integration).
- JedxCollectorFunctionIamRole — implicit role ARN created for that function (handy for attaching extra perms).

How the pieces flow
1) Files in → You upload to JedxCarInputBucket → S3ToKinesisFunction fires → sends a record to JedxCarInputStream.
2) Stream → validate → store → CarValidationFunction consumes Kinesis, writes good results to JedxCarBucket (and/or DynamoDB), errors to JedxCarErrorBucket.
3) API surface → JedxCarApi exposes /car, /car/record/..., /car/records/{senderId}, /car/collector/send, /car/login secured by Cognito; each path maps to the corresponding Lambda handler above.

Suggestions
- Parameterize COLLECTOR_URL: replace the hardcoded value with !Ref CollectorUrl for per-environment deploys.
- Least-privilege S3 policies: if possible, scope S3 ARNs down to paths/prefixes actually used.
- Retries on validation: consider a small MaximumRetryAttempts (e.g., 1–2) or a DLQ to avoid silent drops.