CREATING A MODEL

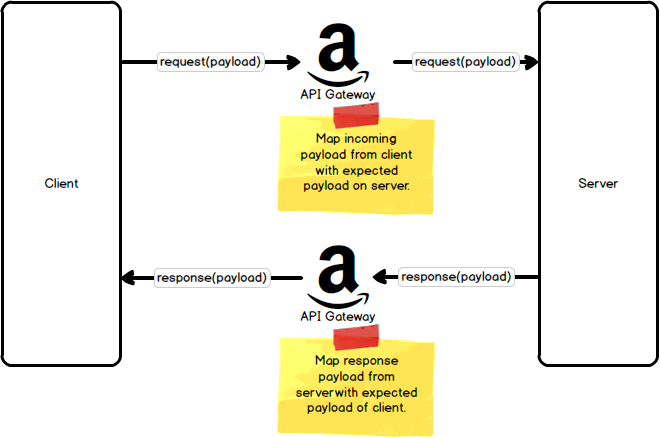
Hello! folks its awesome to have you here working towards understanding how to create a model. Before we get to know the implementation part lets understand why we need a model.

TOPICS:

* [Why to Implement a Model?](#_WHY…?)
* [What Is a Model?](#_WHAT_IS_A)
* [JSON Schema?](#_JSON_SCHEMA)
* [Creating a Model](#_CREATION_OF_MODEL:)

# **WHY…?**

Amazon’s API Gateway provides the facilities to map an incoming request’s payload to match the required format of an integration backend.



To map the payload between the client and the server, API Gateway makes use of “Models” and “Mapping Templates”.

# **WHAT IS A MODEL...?**

A model defines the structure of the incoming payload using [JSON Schema](http://json-schema.org/). The model is an optional, but not required, piece of API Gateway. By providing a model, you make it easier to define the upcoming mapping template that does the transformation between the client and server.

# **JSON SCHEMA**

The API Gateway model for this JSON payload is described using JSON Schema. JSON Schema is a vocabulary allowing you validate JSON documents. The JSON payload being validated is called an instance, and the document describing what a valid payload looks like is called the schema.

JSON Schema is used to describe a JSON document, so it helps to understand what, exactly, a JSON document is composed of these primitives:

* objects: {"field1": "value1", "field2": "value2"}
* arrays: ["first", "second", "third"]
* numbers: 42 or 3.1415926
* strings: "Lorem ipsum dolor sit amet"
* booleans: true or false
* null: null

The responsibility of JSON Schema is to describe a JSON document built from these primitives.

Request Payload:

{

"playerId": "1234567890",

"alias": "soofaloofa",

"displayName": "Kevin Sookocheff",

"profilePhotoUrl": "https://api.example.com/player/1234567890/avatar.png"

}

Respective JSON SCHEMA [Simple]

{

"$schema": "http://json-schema.org/draft-04/schema#",

"type": "object",

"properties": {

"playerId": {“type": "string”},

"alias": {“type": "string”},

"displayName": {“type": "string”},

"profilePhotoUrl": {“type": "string”}

}

}

This JSON Schema follows the Draft 4 specification and simply declares the types expected for each field in the request. For now, we simply declare a root type for the payload as object and allow the string type for each incoming field. You could choose to further restrict each string to be of a certain length, to match a regular expression, or to be of a certain format, such as date-time, or email. For an excellent guide on on JSON Schema, consult [this guide](https://spacetelescope.github.io/understanding-json-schema/index.html).

Since you have understood why to implement the model. Let’s get to the Implementation.

# **CREATION OF MODEL:**

After you have created your API, open it and on the left sections you’d find Models. Choose it and CREATE a MODEL

Graphical user interface, text, application, chat or text message

Description automatically generated

Follow it up by giving the following details as your desire

Graphical user interface, text

Description automatically generated

For the Model Schema give the code that is given in the **studentInfo-Model-Schema** file that I have shared in this folder.

Graphical user interface, text, application

Description automatically generated

After this click save and then go to Resources and go to **Method Request** and click **request body** inside it add the model file

Graphical user interface, application

Description automatically generated

This marks the end of creation of model and attaching it to validate the incoming payload [request information]. **Happy Learning!!**

**Note**: To understand how this model can be used to implement a MAPPING TEMPLATE continue to **Data\_Transformations-Mapping Template** file.

References:

* <https://sookocheff.com/post/api/understanding-api-gateway-payload-mappings/>
* https://docs.aws.amazon.com/apigateway/latest/developerguide/how-to-create-model.html