Portfolio Assignment: API Spec

CS493: Cloud Application Development

Colin Francis

Fall 2022

Oregon State University

# Table of Contents

[Table of Contents 1](#_Toc120655796)

[Data Model 2](#_Toc120655797)

[Relationship Between Non-User Entities 4](#_Toc120655798)

[Relationship Between User Entity and Non-User Entity 4](#_Toc120655799)

[Description of How the User Entity is Being Modeled 4](#_Toc120655800)

[The Unique Identifier for a User 5](#_Toc120655801)

[Requesting Aircrafts for User’s (User Identifier for a User) 5](#_Toc120655802)

[Mapping a Supplied JWT to the Identifier of a User 6](#_Toc120655803)

[Create an Aircraft 7](#_Toc120655804)

[Get All Aircrafts 12](#_Toc120655805)

[Get an Aircraft 17](#_Toc120655806)

[Partially Update an Aircraft 20](#_Toc120655807)

[Fully Update an Aircraft 25](#_Toc120655808)

[Delete an Aircraft 30](#_Toc120655809)

[Create a Hangar 32](#_Toc120655810)

[Get All Hangars 37](#_Toc120655811)

[Get a Hangar 41](#_Toc120655812)

[Add an Aircraft to a Hangar 43](#_Toc120655813)

[Remove an Aircraft from a Hangar 46](#_Toc120655814)

[Partially Update a Hangar 49](#_Toc120655815)

[Fully Update a Hangar 54](#_Toc120655816)

[Delete a Hangar 59](#_Toc120655817)

# Data Model

The app stores three kinds of entities in Datastore – Users, Aircrafts, Hangars.

## Users

|  |  |  |  |
| --- | --- | --- | --- |
| **Property** | **Data Type** | **Required?** | **Notes** |
| id | String | No | The unique id of the user. This is automatically generated for the user when they first sign-in with Google. |
| givenName | String | No | The given name of the user. |
| familyName | String | No | The family name of the user. |
| aircrafts | Integer | No | The number of aircrafts owned by the user. |

## Aircrafts

|  |  |  |  |
| --- | --- | --- | --- |
| **Property** | **Data Type** | **Required?** | **Notes** |
| id | Integer | No | The unique id of the aircraft. Datastore automatically generates this when the aircraft is created. |
| make | String | Yes | The make of the aircraft, e.g., Boeing, Cessna, Embraer, etc. |
| model | String | Yes | The model of the aircraft, e.g., 747, 150, EMB 110, etc. |
| wingspan | Double | Yes | The wingspan of the aircraft in meters. |
| hangar | Integer | No | The id of the hangar that the aircraft is parked in. If the hangar is not parked in a hangar, this value will be null. |
| ownerId | String | No | The id of the aircraft’s owner. This will be equal to the id of the User entity that owns the aircraft. |

## Hangars

|  |  |  |  |
| --- | --- | --- | --- |
| **Property** | **Data Type** | **Required?** | **Notes** |
| id | Integer | No | The unique id of the hangar. Datastore automatically generates this when the hangar is created. |
| name | String | Yes | The name of the hangar. |
| city | String | Yes | The United States city that the hangar is located in. |
| state | String | Yes | The Unites States state that the hangar is located in. |
| capacity | Integer | Yes | The maximum number of aircrafts that the hangar can store. |
| aircrafts | Array | No | An array containing the IDs of the aircrafts currently stored in the hangar. |

# Relationship Between Non-User Entities

This application has two non-user entities: Aircrafts and Hangars.

A newly created Aircraft will be considered “in-flight”. This means that the Aircraft is not assigned to a Hangar and the Aircraft’s “hangar” attribute will be null. A newly created Hangar will be considered empty. This means that no Aircrafts will be parked in the Hangar and the Hangar’s “aircrafts” attribute (an array) will be empty, i.e., the length of the array will be zero.

The owner of an Aircraft can add an Aircraft to a Hangar. For this to be successful, the Aircraft must not be parked at another Hangar and the Hangar must not be at capacity. In other words, there is a one-to-many relationship between Aircrafts and Hangars – an Aircraft can be parked at only one Hangar, but a Hangar can have many Aircrafts parked in it.

Additionally, the owner of an Aircraft can remove the Aircraft from the Hangar it is parked at. This will automatically mark the Aircraft as being “in-flight”, i.e., it’s “hangar” attribute will be set to null. This will also automatically remove the Aircraft from the Hangar’s “aircrafts” attribute.

# Relationship Between User Entity and Non-User Entity

There is a one-to-many relationship between User’s and Aircraft’s. A User can be the owner of multiple Aircrafts, but an Aircraft can only be owned by one User.

When an Aircraft is created, a JSON web token is required to be provided as a Bearer Token for authorization. The JWT is decoded and the sub attribute of the JWT is added as the “ownerId” attribute of the Aircraft being created.

Thus, access to Aircraft’s is restricted to the Aircraft’s owner. A User making a request with a specific Aircraft as the target must be authorized to perform the requested action on the specific Aircraft, i.e., they must be the owner of the Aircraft.

# Description of How the User Entity is Being Modeled

As shown in the Data Model section of this API specification, Users are modeled with an id, a givenName, a familyName, and an aircrafts attribute. The “aircrafts” attribute is an integer that represents the number of aircrafts owned by the User.

Users are not created via an API endpoint. Instead, User’s will be automatically created when a User logs in with Google. Logging in / signing up will take the User through the OAuth 2.0 flow using Google as the OAuth provider. The User will need to give permission for the application to access their profile information. The application will then use the given name and family name it has access to for the User to create a User entity with the attributes mentioned above.

A new User will automatically be created with an “aircrafts” attribute value of 0, i.e., they have no aircrafts upon creation.

# The Unique Identifier for a User

The “id” attribute for a User entity is the unique identifier for a User.

During the OAuth 2.0 flow that the User goes through during login or signup, an access code is exchanged with Google for tokens by the client. One of these tokens is an “id\_token” which is a JSON web token issued by Google. The JSON web token will be decoded by the client and the sub attribute in the decoded JSON web token is used as the “id” attribute value for the User entity. Each User entity is thus uniquely identified by their “id” attribute whose value is the sub attribute value from the JSON web token issued by Google.

The “id” attribute for a User will be displayed along with a JSON web token once the User logs in or signs up:

Text

Description automatically generated

# Requesting Aircrafts for User’s (User Identifier for a User)

Aircrafts are created and owned by User’s. Thus, to view a specific Aircraft or a collection of Aircrafts, the requester must provide authorization. The authorization type required is a Bearer Token. The token itself must be the JSON web token that is provided after logging in to their account:

Text

Description automatically generated

# Mapping a Supplied JWT to the Identifier of a User

When a JSON web token is provided as part of a request related to an Aircraft or a collection of Aircrafts, the supplied JWT and the request object will be passed to middleware used to validate the provided JWT. A valid JWT will be decoded and will be the value of an attribute named “jwt” which is added to the request object by the validation middleware before being passed to the next piece of middleware. A missing JWT or an invalid JWT will result in the “jwt” attribute added to the request object being set to null.

Since each aircraft created has an “ownerId” attribute which has the value of the JWT sub attribute used to uniquely identify each User entity, several actions on the server-side can take place to protect Aircrafts:

1. If an endpoint requires a valid JWT to be provided and the “jwt” attribute added to the request object by the validation middleware is null, then the request can be rejected with a 401 status code. In this case, we know that the required JWT was missing or invalid.
2. The supplied JWT sub attribute can be checked against the “ownerId” attribute of an Aircraft being targeted in a request. If the supplied JWT sub attribute does not match the Aircraft’s “ownerId” attribute, the request can be rejected with a 403 status code. In this case, the requester is not the owner of the Aircraft.
3. If a request is being made to view all Aircrafts, Datastore can be queried for all Aircrafts in the Aircrafts kind whose “ownerId” is equal to the sub attribute of the provided JWT. This allows a requester to only view the Aircrafts that they own.

Ultimately, a supplied JWT is mapped to the Identifier of a User via the sub attribute from the JWT. This can then be used to protect Aircraft’s by making sure the User owns the Aircraft by comparing the Aircraft’s “ownerId” attribute to the sub attribute of the provided JWT.

# Create an Aircraft

Allows you to create a new aircraft.

|  |
| --- |
| POST /aircrafts |

## Request

### Media Type(s)

application/json

### Authorization

Required

### Authorization Type

Bearer Token

### Path Parameters

None

### Request Body

Required

### Request Body Format

JSON

### Request JSON Attributes

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Description** | **Notes** | **Required?** |
| make | The make of the aircraft, e.g., Boeing, Cessna, Embraer, etc. | The minimum permissible string length of an aircraft make is 1 character. The maximum permissible string length for an aircraft make is 255 characters.  All aircraft makes must use characters that match the regular expression: ^[A-Za-z ]+$ | Yes |
| model | The model of the aircraft, e.g., 747, 150, EMB 110, etc. | The minimum permissible string length of an aircraft model is 1 character. The maximum permissible string length for an aircraft model is 255 characters.  All aircraft models must use characters that match the regular expression: ^[A-Za-z0-9 -/]+$ | Yes |
| wingspan | The wingspan of the aircraft in meters. | The minimum permissible length of an aircraft wingspan is 2m. The maximum permissible length of an aircraft wingspan is 150m. | Yes |

### Request Body Example

|  |
| --- |
| Content-Type: application/json  {  “make”: “Boeing”,  “model”: “747”,  “wingspan”: 59.6  } |

## Response

### Media Type(s)

application/json

### Response Body Format

JSON

### Response Statuses

|  |  |  |
| --- | --- | --- |
| **Outcome** | **Status Code** | **Notes** |
| Success | 201 Created |  |
| Failure | 401 Unauthorized | If the request is missing a JSON web token or does not contain a valid JSON web token as a Bearer Token the aircraft will not be created and a 401 status code will be returned. |
| Failure | 415 Unsupported Media Type | If the media type of the content provided in the request body is not JSON and the associated Content-Type request header is not application/json, the aircraft will not be created and a 415 status code will be returned. |
| Failure | 406 Not Acceptable | If the Accept header in the request is not application/json, the aircraft will not be created and a 406 status code will be returned.  Note: If the client has an Accept header set to \*/\*, i.e., any MIME type, the server will serve application/json in the response. The server assumes the client is indifferent to a specific media type as one was not explicitly set in the Accept header. |
| Failure | 400 Bad Request | If the request is missing any of the 3 required attributes, the aircraft will not be created and a 400 status code will be returned.  If any of the 3 required attributes in the request are misconfigured, the aircraft will not be created and a 400 status code will be returned. |
| Failure | 500 Internal Server Error | An internal server error has occurred. |

### Response Examples

#### Success

|  |
| --- |
| Status: 201 Created  Content-Type: application/json  {  “id”: 1234,  “make”: “Boeing”,  “model”: “747”,  “wingspan”: 59.6,  “hangar”: null,  “ownerId”: “5678”,  “self”: “https://appspot.com/aircrafts/1234”  } |

#### Failure

|  |
| --- |
| Status Code: 401 Unauthorized  Content-Type: application/json  {  “Error”: “Bearer token is missing or invalid”  } |

#### Failure

|  |
| --- |
| Status: 415 Unsupported Media Type  Content-Type: application/json  {  “Error”: “This endpoint only accepts application/json”  } |

#### Failure

|  |
| --- |
| Status: 406 Not Acceptable  Content-Type: application/json  {  “Error”: “This endpoint only serves application/json”  } |

#### Failure

|  |
| --- |
| Status: 400 Bad Request  Content-Type: application/json  {  “Error”: “Make attribute is missing or invalid”  } |

#### Failure

|  |
| --- |
| Status: 400 Bad Request  Content-Type: application/json  {  “Error”: “Model attribute is missing or invalid”  } |

#### Failure

|  |
| --- |
| Status: 400 Bad Request  Content-Type: application/json  {  “Error”: “Wingspan attribute is missing or invalid”  } |

#### Failure

|  |
| --- |
| Status: 500 Internal Server Error  Content-Type: application/json  {  “Error”: “An internal server error has occurred”  } |

# Get All Aircrafts

Allows you to fetch all aircrafts that you own.

|  |
| --- |
| GET /aircrafts |

## Request

### Media Type(s)

None

### Authorization

Required

### Authorization Type

Bearer Token

### Path Parameters

None

### Request Body

None

## Response

### Media Type(s)

application/json

### Response Body Format

JSON

### Response Statuses

|  |  |  |
| --- | --- | --- |
| **Outcome** | **Status Code** | **Notes** |
| Success | 200 OK | The maximum number of aircrafts returned by a single request will be 5.  If there are additional aircrafts that can be retrieved, the JSON object returned in the response body will contain an aircrafts array and a “next” attribute with the attribute value being a cursor allowing you to retrieve more aircrafts via pagination.  A “totalAircrafts” attribute in the response body shows the total number of aircrafts that you own. This attribute also represents the total number of aircrafts that you can retrieve.  When there are no aircrafts left to retrieve, the object returned will only include the aircrafts array and the “totalAircrafts” attribute. |
| Failure | 401 Unauthorized | If the request is missing a JSON web token or does not contain a valid JSON web token as a Bearer Token the request will fail and a 401 status code will be returned. |
| Failure | 406 Not Acceptable | If the Accept header in the request is not application/json, the request will fail and a 406 status code will be returned.  Note: If the client has an Accept header set to \*/\*, i.e., any MIME type, the server will serve application/json in the response. The server assumes the client is indifferent to a specific media type as one was not explicitly set in the Accept header. |
| Failure | 500 Internal Server Error | An internal server error has occurred. |

### Response Examples

#### Success

|  |
| --- |
| Status Code: 200 OK  Content-Type: application/json  {  “totalAircrafts”: 6,  “aircrafts”: [  {  “id”: 1234,  “make”: “Boeing”,  “model”: “747”,  “wingspan”: 59.6,  “hangar”: null,  “ownerId”: “5678”,  “self”: “https://appspot.com/aircrafts/1234”  },  {  “id”: 1357,  “make”: “Tarrant”,  “model”: “Tabor”,  “wingspan”: 40.0,  “hangar”: {  “id”: 8765,  “self”: “https://appspot.com/hangars/8765”  },  “ownerId”: “5678”,  “self”: “https://appspot.com/aircrafts/1357”  },  {  “id”: 2468,  “make”: “Airbus”,  “model”: “Beluga”,  “wingspan”: 44.8,  “hangar”: null,  “ownerId”: “5678”,  “self”: “https://appspot.com/aircrafts/2468”  },  {  “id”: 4690,  “make”: “Convair”,  “model”: “B-36”,  “wingspan”: 70.1,  “hangar”: {  “id”: 5928,  “self”: “https://appspot.com/hangars/5928”  },  “ownerId”: “5678”,  “self”: “https://appspot.com/aircrafts/4690”  },  {  “id”: 9876,  “make”: “Antonov”,  “model”: “An-225”,  “wingspan”: 88.4,  “hangar”: {  “id”: 0629,  “self”: “https://appspot.com/hangars/0629”  },  “ownerId”: “5678”,  “self”: “https://appspot.com/aircrafts/9876”  },  ],  “next”: “https://appspot.com/aircrafts?cursor=CjYSMGoWZH5jczQ5My1maW5hbC1mcmFuY2ljb3IWCxIJQWlyY3JhZnRzGICAgJiH2JAKDBgAIAA%3D”  } |

#### Success

|  |
| --- |
| Status: 200 OK  Content-Type: application/json  {  “totalAircrafts”: 1,  “aircrafts”: [  {  “id”: 1234,  “make”: “Boeing”,  “model”: “747”,  “wingspan”: 59.6,  “hangar”: null,  “ownerId”: “5678”,  “self”: “https://appspot.com/aircrafts/1234”  },  ]  } |

#### Failure

|  |
| --- |
| Status Code: 401 Unauthorized  Content-Type: application/json  {  “Error”: “Bearer token is missing or invalid”  } |

#### Failure

|  |
| --- |
| Status: 406 Not Acceptable  Content-Type: application/json  {  “Error”: “This endpoint only serves application/json”  } |

#### Failure

|  |
| --- |
| Status: 500 Internal Server Error  Content-Type: application/json  {  “Error”: “An internal server error has occurred”  } |

# Get an Aircraft

Allows you to fetch a specific aircraft that you own.

|  |
| --- |
| GET /aircrafts/:aircraftId |

## Request

### Media Type(s)

None

### Authorization

Required

### Authorization Type

Bearer Token

### Path Parameters

|  |  |
| --- | --- |
| **Name** | **Description** |
| aircraftId | The ID of the aircraft |

### Request Body

None

## Response

Media Type(s)

application/json

### Response Body Format

JSON

### Response Statuses

|  |  |  |
| --- | --- | --- |
| **Outcome** | **Status Code** | **Notes** |
| Success | 200 OK |  |
| Failure | 401 Unauthorized | If the request is missing a JSON web token or does not contain a valid JSON web token as a Bearer Token the request will fail and a 401 status code will be returned. |
| Failure | 406 Not Acceptable | If the Accept header in the request is not application/json, the request will fail and a 406 status code will be returned.  Note: If the client has an Accept header set to \*/\*, i.e., any MIME type, the server will serve application/json in the response. The server assumes the client is indifferent to a specific media type as one was not explicitly set in the Accept header. |
| Failure | 404 Not Found | No aircraft with this aircraftId exists. |
| Failure | 403 Forbidden | If the user making the request does not own the aircraft they are making the request for, the request will fail and a 403 status code will be returned. |
| Failur | 500 Internal Server Error | An internal server error has occurred. |

### Response Examples

#### Success

|  |
| --- |
| Status: 200 OK  Content-Type: application/json  {  “id”: 1234,  “make”: “Boeing”,  “model”: “747”,  “wingspan”: 59.6,  “hangar”: null,  “ownerId”: “5678”,  “self”: “https://appspot.com/aircrafts/1234”  } |

#### Failure

|  |
| --- |
| Status: 401 Unauthorized  Content-Type: application/json  {  “Error”: “Bearer token is missing or invalid”  } |

#### Failure

|  |
| --- |
| Status: 406 Not Acceptable  Content-Type: application/json  {  “Error”: “This endpoint only serves application/json”  } |

#### Failure

|  |
| --- |
| Status: 404 Not Found  Content-Type: application/json  {  “Error”: “No aircraft with this aircraftId exists”  } |

#### Failure

|  |
| --- |
| Status: 403 Forbidden  Content-Type: application/json  {  “Error”: “You are not authorized to perform this action”  } |

#### Failure

|  |
| --- |
| Status: 500 Internal Server Error  Content-Type: application/json  {  “Error”: “An internal server error has occurred”  } |

# Partially Update an Aircraft

Allows you to partially update the attributes of an already created aircraft.

|  |
| --- |
| PATCH /aircrafts/:aircraftId |

## Request

### Media Type(s)

application/json

### Authorization

Required

### Authorization Type

Bearer Token

### Path Parameters

|  |  |
| --- | --- |
| **Name** | **Description** |
| aircraftId | The ID of the aircraft |

### Request Body

Required

### Request Body Format

JSON

### Request JSON Attributes

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Description** | **Notes** | **Required?** |
| make | The make of the aircraft, e.g., Boeing, Cessna, Embraer, etc. | The minimum permissible string length of an aircraft make is 1 character. The maximum permissible string length for an aircraft make is 255 characters.  All aircraft makes must use characters that match the regular expression: ^[A-Za-z ]+$ | No |
| model | The model of the aircraft, e.g., 747, 150, EMB 110, etc. | The minimum permissible string length of an aircraft model is 1 character. The maximum permissible string length for an aircraft model is 255 characters.  All aircraft models must use characters that match the regular expression: ^[A-Za-z0-9 -/]+$ | No |
| wingspan | The wingspan of the aircraft in meters. | The minimum permissible length of an aircraft wingspan is 2m. The maximum permissible length of an aircraft wingspan is 150m. | No |

\*Note: Any additional attributes provided in the request body that aren’t make, model, or wingspan will be ignored.

### Request Body Example

|  |
| --- |
| Content-Type: application/json  {  “model”: “Super Hornet”  } |

## Response

### Media Type(s)

application/json

### Response Body Format

JSON

### Response Statuses

|  |  |  |
| --- | --- | --- |
| **Outcome** | **Status Code** | **Notes** |
| Success | 200 OK |  |
| Failure | 401 Unauthorized | If the request is missing a JSON web token or does not contain a valid JSON web token as a Bearer Token the aircraft will not be updated and a 401 status code will be returned. |
| Failure | 415 Unsupported Media Type | If the media type of the content provided in the request body is not JSON and the associated Content-Type request header is not application/json, the aircraft will not be updated and a 415 status code will be returned. |
| Failure | 406 Not Acceptable | If the Accept header in the request is not application/json, the aircraft will not be updated and a 406 status code will be returned.  Note: If the client has an Accept header set to \*/\*, i.e., any MIME type, the server will serve application/json in the response. The server assumes the client is indifferent to a specific media type as one was not explicitly set in the Accept header. |
| Failure | 400 Bad Request | If any of the 3 possible attributes in the request are provided but misconfigured, the aircraft will not be updated and a 400 status code will be returned. |
| Failure | 404 Not Found | No aircraft with this aircraftId exists. |
| Failure | 403 Forbidden | If the user making the request does not own the aircraft they are trying to update, the request will fail and a 403 status code will be returned. |
| Failure | 500 Internal Server Error | An internal server error has occurred. |

### Response Examples

#### Success

|  |
| --- |
| Status: 200 OK  Content-Type: application/json  {  “id”: 1234,  “make”: “Boeing”,  “model”: “Super Hornet”,  “wingspan”: 59.6,  “hangar”: null,  “ownerId”: “5678”,  “self”: “https://appspot.com/aircrafts/1234”  } |

#### Failure

|  |
| --- |
| Status Code: 401 Unauthorized  Content-Type: application/json  {  “Error”: “Bearer token is missing or invalid”  } |

#### Failure

|  |
| --- |
| Status: 415 Unsupported Media Type  Content-Type: application/json  {  “Error”: “This endpoint only accepts application/json”  } |

#### Failure

|  |
| --- |
| Status: 406 Not Acceptable  Content-Type: application/json  {  “Error”: “This endpoint only serves application/json”  } |

#### Failure

|  |
| --- |
| Status: 400 Bad Request  Content-Type: application/json  {  “Error”: “Make attribute is invalid”  } |

#### Failure

|  |
| --- |
| Status: 400 Bad Request  Content-Type: application/json  {  “Error”: “Model attribute is invalid”  } |

#### Failure

|  |
| --- |
| Status: 400 Bad Request  Content-Type: application/json  {  “Error”: “Wingspan attribute is invalid”  } |

#### Failure

|  |
| --- |
| Status: 404 Not Found  Content-Type: application/json  {  “Error”: “No aircraft with this aircraftId exists”  } |

#### Failure

|  |
| --- |
| Status: 403 Forbidden  Content-Type: application/json  {  “Error”: “You are not authorized to perform this action”  } |

#### Failure

|  |
| --- |
| Status: 500 Internal Server Error  Content-Type: application/json  {  “Error”: “An internal server error has occurred”  } |

# Fully Update an Aircraft

Allows you to update all attributes of an aircraft that you own.

|  |
| --- |
| PUT /aircrafts/:aircraftId |

## Request

### Media Type(s)

application/json

### Authorization

Required

### Authorization Type

Bearer Token

### Path Parameters

|  |  |
| --- | --- |
| **Name** | **Description** |
| aircraftId | The ID of the aircraft |

### Request Body

Required

### Request Body Format

JSON

### Request JSON Attributes

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Description** | **Notes** | **Required?** |
| make | The make of the aircraft, e.g., Boeing, Cessna, Embraer, etc. | The minimum permissible string length of an aircraft make is 1 character. The maximum permissible string length for an aircraft make is 255 characters.  All aircraft makes must use characters that match the regular expression: ^[A-Za-z ]+$ | Yes |
| model | The model of the aircraft, e.g., 747, 150, EMB 110, etc. | The minimum permissible string length of an aircraft model is 1 character. The maximum permissible string length for an aircraft model is 255 characters.  All aircraft models must use characters that match the regular expression: ^[A-Za-z0-9 -/]+$ | Yes |
| wingspan | The wingspan of the aircraft in meters. | The minimum permissible length of an aircraft wingspan is 2m. The maximum permissible length of an aircraft wingspan is 150m. | Yes |

\*Note: Any additional attributes provided in the request body that aren’t make, model, or wingspan will be ignored.

### Request Body Example

|  |
| --- |
| Content-Type: application/json  {  “make”: “McDonnell”,  “model”: “F-4 Phantom II”,  “wingspan”: 11.7  } |

## Response

### Media Type(s)

application/json

### Response Body Format

JSON

### Response Statuses

|  |  |  |
| --- | --- | --- |
| **Outcome** | **Status Code** | **Notes** |
| Success | 200 OK |  |
| Failure | 401 Unauthorized | If the request is missing a JSON web token or does not contain a valid JSON web token as a Bearer Token the aircraft will not be updated and a 401 status code will be returned. |
| Failure | 415 Unsupported Media Type | If the media type of the content provided in the request body is not JSON and the associated Content-Type request header is not application/json, the aircraft will not be updated and a 415 status code will be returned. |
| Failure | 406 Not Acceptable | If the Accept header in the request is not application/json, the aircraft will not be updated and a 406 status code will be returned.  Note: If the client has an Accept header set to \*/\*, i.e., any MIME type, the server will serve application/json in the response. The server assumes the client is indifferent to a specific media type as one was not explicitly set in the Accept header. |
| Failure | 400 Bad Request | If the request is missing any of the 3 required attributes, the aircraft will not be updated and a 400 status code will be returned.  If any of the 3 required attributes in the request are misconfigured, the aircraft will not be updated and a 400 status code will be returned. |
| Failure | 404 Not Found | No aircraft with this aircraftId exists. |
| Failure | 403 Forbidden | If the user making the request does not own the aircraft they are trying to update, the request will fail and a 403 status code will be returned. |
| Failure | 500 Internal Server Error | An internal server error has occurred. |

### Response Examples

#### Success

|  |
| --- |
| Status: 200 OK  Content-Type: application/json  {  “id”: 1234,  “make”: “McDonnell”,  “model”: “F-4 Phantom II”,  “wingspan”: 11.7,  “hangar”: null,  “ownerId”: “5678”,  “self”: “https://appspot.com/aircrafts/1234”  } |

#### Failure

|  |
| --- |
| Status Code: 401 Unauthorized  Content-Type: application/json  {  “Error”: “Bearer token is missing or invalid”  } |

#### Failure

|  |
| --- |
| Status: 415 Unsupported Media Type  Content-Type: application/json  {  “Error”: “This endpoint only accepts application/json”  } |

#### Failure

|  |
| --- |
| Status: 406 Not Acceptable  Content-Type: application/json  {  “Error”: “This endpoint only serves application/json”  } |

#### Failure

|  |
| --- |
| Status: 400 Bad Request  Content-Type: application/json  {  “Error”: “Make attribute is missing or invalid”  } |

#### Failure

|  |
| --- |
| Status: 400 Bad Request  Content-Type: application/json  {  “Error”: “Model attribute is missing or invalid”  } |

#### Failure

|  |
| --- |
| Status: 400 Bad Request  Content-Type: application/json  {  “Error”: “Wingspan attribute is missing or invalid”  } |

#### Failure

|  |
| --- |
| Status: 404 Not Found  Content-Type: application/json  {  “Error”: “No aircraft with this aircraftId exists”  } |

#### Failure

|  |
| --- |
| Status: 403 Forbidden  Content-Type: application/json  {  “Error”: “You are not authorized to perform this action”  } |

#### Failure

|  |
| --- |
| Status: 500 Internal Server Error  Content-Type: application/json  {  “Error”: “An internal server error has occurred”  } |

# Delete an Aircraft

Allows you to delete an aircraft that you own. If the aircraft was parked in a hangar, deleting the aircraft will automatically remove it from the hangar.

|  |
| --- |
| DELETE /aircrafts/:aircraftId |

## Request

### Media Type(s)

None

### Authorization

Required

### Authorization Type

Bearer Token

### Path Parameters

|  |  |
| --- | --- |
| **Name** | **Description** |
| aircraftId | The ID of the aircraft |

### Request Body

None

## Response

### Media Type(s)

Success: None

Failure: application/json

### Response Body Format

Success: No Body

Failure: JSON

### Response Statuses

|  |  |  |
| --- | --- | --- |
| **Outcome** | **Status Code** | **Notes** |
| Success | 204 No Content |  |
| Failure | 401 Unauthorized | If the request is missing a JSON web token or does not contain a valid JSON web token as a Bearer Token the request will fail and a 401 status code will be returned. |
| Failure | 404 Not Found | No aircraft with this aircraftId exists. |
| Failure | 403 Forbidden | If the user making the request does not own the aircraft they are trying to delete, the request will fail and a 403 status code will be returned. |
| Failure | 500 Internal Server Error | An internal server error has occurred. |

### Response Examples

#### Success

|  |
| --- |
| Status: 204 No Content |

#### Failure

|  |
| --- |
| Status: 401 Unauthorized  Content-Type: application/json  {  “Error”: “Bearer token is missing or invalid”  } |

#### Failure

|  |
| --- |
| Status: 404 Not Found  Content-Type: application/json  {  “Error”: “No aircraft with this aircraftId exists”  } |

#### Failure

|  |
| --- |
| Status: 403 Forbidden  Content-Type: application/json  {  “Error”: “You are not authorized to perform this action”  } |

#### Failure

|  |
| --- |
| Status: 500 Internal Server Error  Content-Type: application/json  {  “Error”: “An internal server error has occurred”  } |

# Create a Hangar

Allows you to create a new hangar.

|  |
| --- |
| POST /hangars |

## Request

### Media Type(s)

application/json

### Authorization

None

### Path Parameters

None

### Request Body

Required

### Request Body Format

JSON

### Request JSON Attributes

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Description** | **Notes** | **Required?** |
| name | The name of the hangar. | The minimum permissible string length of a hangar name is 1 character. The maximum permissible string length for a hangar name is 255 characters.  All hangar name must use characters that match the regular expression: ^[A-Za-z0-9 ./,-]+$ | Yes |
| city | The United States city that the hangar is located in. | The minimum permissible string length of a hangar city is 1 character. The maximum permissible string length for a hangar city is 255 characters.  All hangar cities must use characters that match the regular expression: ^[A-Za-z .,]+$ | Yes |
| state | The United States state that the hangar is located in. | The minimum permissible string length of a hangar state is 1 character. The maximum permissible string length for a hangar state is 255 characters.  All hangar states must use characters that match the regular expression: ^[A-Za-z ]+$ | Yes |
| capacity | The maximum number of aircrafts that the hangar can hold. | The smallest permissible capacity value is 1. The largest permissible capacity value is 5. | Yes |

### Request Body Example

|  |
| --- |
| Content-Type: application/json  {  “name”: “Arch Hangar”,  “city”: “Limestone”,  “state”: “Maine”,  “capacity”: 2  } |

## Response

### Media Type(s)

application/json

### Response Body Format

JSON

### Response Statuses

|  |  |  |
| --- | --- | --- |
| **Outcome** | **Status Code** | **Notes** |
| Success | 201 Created |  |
| Failure | 415 Unsupported Media Type | If the media type of the content provided in the request body is not JSON and the associated Content-Type request header is not application/json, the hangar will not be created and a 415 status code will be returned. |
| Failure | 406 Not Acceptable | If the Accept header in the request is not application/json, the hangar will not be created and a 406 status code will be returned.  Note: If the client has an Accept header set to \*/\*, i.e., any MIME type, the server will serve application/json in the response. The server assumes the client is indifferent to a specific media type as one was not explicitly set in the Accept header. |
| Failure | 400 Bad Request | If the request is missing any of the 4 required attributes, the hangar will not be created and a 400 status code will be returned.  If any of the 4 required attributes in the request are misconfigured, the hangar will not be created and a 400 status code will be returned. |
| Failure | 500 Internal Server Error | An internal server error has occurred. |

### Response Examples

#### Success

|  |
| --- |
| Status: 201 Created  Content-Type: application/json  {  “id”: 1234,  “name”: “Arch Hangar”,  “city”: “Limestone”,  “state”: “Maine”,  “capacity”: 2,  “aircrafts”: [],  “self”: “https://appspot.com/hangars/1234”  } |

#### Failure

|  |
| --- |
| Status: 415 Unsupported Media Type  Content-Type: application/json  {  “Error”: “This endpoint only accepts application/json”  } |

#### Failure

|  |
| --- |
| Status: 406 Not Acceptable  Content-Type: application/json  {  “Error”: “This endpoint only serves application/json”  } |

#### Failure

|  |
| --- |
| Status: 400 Bad Request  Content-Type: application/json  {  “Error”: “Name attribute is missing or invalid”  } |

#### Failure

|  |
| --- |
| Status: 400 Bad Request  Content-Type: application/json  {  “Error”: “City attribute is missing or invalid”  } |

#### Failure

|  |
| --- |
| Status: 400 Bad Request  Content-Type: application/json  {  “Error”: “State attribute is missing or invalid”  } |

#### Failure

|  |
| --- |
| Status: 400 Bad Request  Content-Type: application/json  {  “Error”: “Capacity attribute is missing or invalid”  } |

#### Failure

|  |
| --- |
| Status: 500 Internal Server Error  Content-Type: application/json  {  “Error”: “An internal server error has occurred”  } |

# Get All Hangars

Allows you to fetch all hangars.

|  |
| --- |
| GET /hangars |

## Request

### Media Type(s)

None

### Authorization

None

### Path Parameters

None

### Request Body

None

## Response

### Media Type(s)

application/json

### Response Body Format

JSON

### Response Statuses

|  |  |  |
| --- | --- | --- |
| **Outcome** | **Status Code** | **Notes** |
| Success | 200 OK | The maximum number of hangars returned will be 5.  If there are additional hangars that can be retrieved, the JSON object returned in the response body will contain a hangars array and a “next” attribute allowing you to retrieve more hangars via pagination. Additionally, a “totalHangars” attribute in the response body shows the total number of hangars.  When there are no hangars left to retrieve, the object returned will only include the hangars array and the “totalHangars” attribute. |
| Failure | 406 Not Acceptable | If the Accept header in the request is not application/json, the request will fail and a 406 status code will be returned.  Note: If the client has an Accept header set to \*/\*, i.e., any MIME type, the server will serve application/json in the response. The server assumes the client is indifferent to a specific media type as one was not explicitly set in the Accept header. |
| Failure | 500 Internal Server Error | An internal server error has occurred. |

### Response Examples

#### Success

|  |
| --- |
| Status Code: 200 OK  Content-Type: application/json  {  “totalHangars”: 7,  “hangars”: [  {  “id”: 1234,  “name”: “Arch Hangar”,  “city”: “Limestone”,  “state”: “Maine”,  “capacity”: 2,  “aircrafts”: [  { “id”: 8765, “self”: “https://appspot.com/aircrafts/8765” },  { “id”: 4321, “self”: “https://appspot.com/aircrafts/4321” }  ],  “self”: “https://appspot.com/hangars/1234”  },  {  “id”: 5678,  “name”: “Riverside Hangar”,  “city”: “St. Paul”,  “state”: “Minnesota”,  “capacity”: 1,  “aircrafts”: [],  “self”: “https://appspot.com/hangars/5678”  },  {  “id”: 9012,  “name”: “Hangar 1”,  “city”: “Los Angeles”,  “state”: “California”,  “capacity”: 5,  “aircrafts”: [  { “id”: 0987, “self”: “https://appspot.com/aircrafts/0987” },  { “id”: 6543, “self”: “https://appspot.com/aircrafts/0987” },  { “id”: 2109, “self”: “https://appspot.com/aircrafts/2109” }  ],  “self”: “https://appspot.com/hangars/9012”  },  {  “id”: 3456,  “name”: “Wingfoot Lake Airship Hangar”,  “city”: “Suffield”,  “state”: “Ohio”,  “capacity”: 3,  “aircrafts”: [],  “self”: “https://appspot.com/hangars/3456”  },  {  ““id”: 7890,  “name”: “Gulfport Army Airfield Hangar”,  “city”: “Gulfport”,  “state”: “Mississippi”,  “capacity”: 4,  “aircrafts”: [],  “self”: “https://appspot.com/hangars/7890”  },  ],  “next”: “https://appspot.com/hangars?cursor=CjYSMGoWZH5jczQ5My1maW5hbC1mcmFuY2ljb3IWCxIJQWlyY3JhZnRzGICAgJiH2JAKDBgAIAA%3D”  } |

#### Success

|  |
| --- |
| Status: 200 OK  Content-Type: application/json  {  “totalHangars”: 2,  “hangars”: [  {  “id”: 1234,  “name”: “Arch Hangar”,  “city”: “Limestone”,  “state”: “Maine”,  “capacity”: 2,  “aircrafts”: [  { “id”: 8765, “self”: “https://appspot.com/aircrafts/8765” },  { “id”: 4321, “self”: “https://appspot.com/aircrafts/4321” }  ],  “self”: “https://appspot.com/hangars/1234”  },  {  “id”: 5678,  “name”: “Riverside Hangar”,  “city”: “St. Paul”,  “state”: “Minnesota”,  “capacity”: 1,  “aircrafts”: [],  “self”: “https://appspot.com/hangars/5678”  },  ]  } |

#### Failure

|  |
| --- |
| Status: 406 Not Acceptable  Content-Type: application/json  {  “Error”: “This endpoint only serves application/json”  } |

#### Failure

|  |
| --- |
| Status: 500 Internal Server Error  Content-Type: application/json  {  “Error”: “An internal server error has occurred”  } |

# Get a Hangar

Allows you to fetch a specific hangar.

|  |
| --- |
| GET /hangars/:hangarId |

## Request

### Media Type(s)

None

### Authorization

None

### Path Parameters

|  |  |
| --- | --- |
| **Name** | **Description** |
| hangarId | The ID of the hangar |

### Request Body

None

## Response

Media Type(s)

application/json

### Response Body Format

JSON

### Response Statuses

|  |  |  |
| --- | --- | --- |
| **Outcome** | **Status Code** | **Notes** |
| Success | 200 OK |  |
| Failure | 406 Not Acceptable | If the Accept header in the request is not application/json, the request will fail and a 406 status code will be returned.  Note: If the client has an Accept header set to \*/\*, i.e., any MIME type, the server will serve application/json in the response. The server assumes the client is indifferent to a specific media type as one was not explicitly set in the Accept header. |
| Failure | 404 Not Found | No hangar with this hangarId exists. |
| Failure | 500 Internal Server Error | An internal server error has occurred. |

### Response Examples

#### Success

|  |
| --- |
| Status: 200 OK  Content-Type: application/json  {  “id”: 1234,  “name”: “Arch Hangar”,  “city”: “Limestone”,  “state”: “Maine”,  “capacity”: 2,  “aircrafts”: [],  “self”: “https://appspot.com/hangars/1234”  } |

#### Failure

|  |
| --- |
| Status: 406 Not Acceptable  Content-Type: application/json  {  “Error”: “This endpoint only serves application/json”  } |

#### Failure

|  |
| --- |
| Status: 404 Not Found  Content-Type: application/json  {  “Error”: “No hangar with this hangarId exists”  } |

#### Failure

|  |
| --- |
| Status: 500 Internal Server Error  Content-Type: application/json  {  “Error”: “An internal server error has occurred”  } |

# Add an Aircraft to a Hangar

Allows you to add an aircraft that you own to a hangar.

|  |
| --- |
| PUT /hangars/:hangarId/aircrafts/:aircraftId |

## Request

### Media Type(s)

None

### Authorization

Required

### Authorization Type

Bearer Token

### Path Parameters

|  |  |
| --- | --- |
| **Name** | **Description** |
| hangarId | The ID of the hangar |
| aircraftId | The ID of the aircraft |

### Request Body

None

## Response

### Media Type(s)

application/json

### Response Body Format

Success: No Body

Failure: JSON

### Response Statuses

|  |  |  |
| --- | --- | --- |
| **Outcome** | **Status Code** | **Description** |
| Success | 204 No Content |  |
| Failure | 401 Unauthorized | If the request is missing a JSON web token or does not contain a valid JSON web token as a Bearer Token the request will fail and a 401 status code will be returned. |
| Failure | 403 Forbidden | If the user making the request does not own the aircraft that they are trying to add to a hangar, the request will fail and a 403 status will be returned. |
| Failure | 400 Bad Request | If the hangar is at capacity or if the aircraft is already parked in another hangar, then the request will fail and a 400 status will be returned. |
| Failure | 404 Not Found | No hangar with this hangarId exists or no aircraft with this aircraftId exists. |
| Failure | 500 Internal Server Error | An internal server error has occurred. |

### Response Examples

#### Success

|  |
| --- |
| Status: 204 No Content |

#### Failure

|  |
| --- |
| Status: 401 Unauthorized  Content-Type: application/json  {  “Error”: “Bearer token is missing or invalid”  } |

#### Failure

|  |
| --- |
| Status: 404 Not Found  Content-Type: application/json  {  “Error”: “No hangar with this hangarId exists”  } |

#### Failure

|  |
| --- |
| Status: 404 Not found  Content-Type: application/json  {  “Error”: “No aircraft with this aircraftId exists”  } |

#### Failure

|  |
| --- |
| Status: 400 Bad Request  Content-Type: application/json  {  “Error”: “This hangar is at capacity”  } |

#### Failure

|  |
| --- |
| Status: 400 Bad Request  Content-Type: application/json  {  “Error”: “This aircraft is already parked in a hangar”  } |

#### Failure

|  |
| --- |
| Status: 403 Forbidden  Content-Type: application/json  {  “Error”: “You are not authorized to perform this action”  } |

#### Failure

|  |
| --- |
| Status: 500 Internal Server Error  Content-Type: application/json  {  “Error”: “An internal server error has occurred”  } |

# Remove an Aircraft from a Hangar

Allows you to remove an aircraft that you own from a hangar.

|  |
| --- |
| DELETE /hangars/:hangarId/aircrafts/:aircraftId |

## Request

### Media Type(s)

None

### Authorization

Required

### Authorization Type

Bearer Token

### Path Parameters

|  |  |
| --- | --- |
| **Name** | **Description** |
| hangarId | The ID of the hangar |
| aircraftId | The ID of the aircraft |

### Request Body

None

## Response

### Media Type(s)

application/json

### Response Body Format

Success: No Body

Failure: JSON

### Response Statuses

|  |  |  |
| --- | --- | --- |
| **Outcome** | **Status Code** | **Description** |
| Success | 204 No Content |  |
| Failure | 401 Unauthorized | If the request is missing a JSON web token or does not contain a valid JSON web token as a Bearer Token the request will fail and a 401 status code will be returned. |
| Failure | 403 Forbidden | If the user making the request does not own the aircraft that they are trying to add to a hangar, then the request will fail and a 403 status will be returned. |
| Failure | 400 Bad Request | If the hangar is empty or if the aircraft is parked in a different hangar, then the request will fail and a 400 status will be returned. |
| Failure | 404 Not Found | No hangar with this hangarId exists or no aircraft with this aircraftId exists. |
| Failure | 500 Internal Server Error | An internal server error has occurred. |

### Response Examples

#### Success

|  |
| --- |
| Status: 204 No Content |

#### Failure

|  |
| --- |
| Status: 401 Unauthorized  Content-Type: application/json  {  “Error”: “Bearer token is missing or invalid”  } |

#### Failure

|  |
| --- |
| Status: 404 Not Found  Content-Type: application/json  {  “Error”: “No hangar with this hangarId exists”  } |

#### Failure

|  |
| --- |
| Status: 404 Not found  Content-Type: application/json  {  “Error”: “No aircraft with this aircraftId exists”  } |

#### Failure

|  |
| --- |
| Status: 400 Bad Request  Content-Type: application/json  {  “Error”: “This hangar is empty”  } |

#### Failure

|  |
| --- |
| Status: 400 Bad Request  Content-Type: application/json  {  “Error”: “This aircraft is not parked at this hangar”  } |

#### Failure

|  |
| --- |
| Status: 403 Forbidden  Content-Type: application/json  {  “Error”: “You are not authorized to perform this action”  } |

#### Failure

|  |
| --- |
| Status: 500 Internal Server Error  Content-Type: application/json  {  “Error”: “An internal server error has occurred”  } |

# Partially Update a Hangar

Allows you to partially update the attributes of an already created hangar.

|  |
| --- |
| PATCH /hangars/:hangarId |

## Request

### Media Type(s)

application/json

### Authorization

None

### Path Parameters

|  |  |
| --- | --- |
| **Name** | **Description** |
| hangarId | The ID of the hangar |

### Request Body

Required

### Request Body Format

JSON

### Request JSON Attributes

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Description** | **Notes** | **Required?** |
| name | The name of the hangar. | The minimum permissible string length of a hangar name is 1 character. The maximum permissible string length for a hangar name is 255 characters.  All hangar name must use characters that match the regular expression: ^[A-Za-z0-9 ./,-]+$ | No |
| city | The United States city that the hangar is located in. | The minimum permissible string length of a hangar city is 1 character. The maximum permissible string length for a hangar city is 255 characters.  All hangar cities must use characters that match the regular expression: ^[A-Za-z .,]+$ | No |
| state | The United States state that the hangar is located in. | The minimum permissible string length of a hangar state is 1 character. The maximum permissible string length for a hangar state is 255 characters.  All hangar states must use characters that match the regular expression: ^[A-Za-z ]+$ | No |
| capacity | The maximum number of aircrafts that the hangar can hold. | The smallest permissible capacity value is 1. The largest permissible capacity value is 5. | No |

\*Note: Any additional attributes provided in the request body that aren’t name, city, state, or capacity will be ignored.

### Request Body Example

|  |
| --- |
| Content-Type: application/json  {  “model”: “The Arch Hangar”,  “capacity”: 3  } |

## Response

### Media Type(s)

application/json

### Response Body Format

JSON

### Response Statuses

|  |  |  |
| --- | --- | --- |
| **Outcome** | **Status Code** | **Notes** |
| Success | 200 OK |  |
| Failure | 415 Unsupported Media Type | If the media type of the content provided in the request body is not JSON and the associated Content-Type request header is not application/json, the hangar will not be updated and a 415 status code will be returned. |
| Failure | 406 Not Acceptable | If the Accept header in the request is not application/json, the hangar will not be updated and a 406 status code will be returned.  Note: If the client has an Accept header set to \*/\*, i.e., any MIME type, the server will serve application/json in the response. The server assumes the client is indifferent to a specific media type as one was not explicitly set in the Accept header. |
| Failure | 400 Bad Request | If any of the 3 possible attributes in the request are misconfigured, the hangar will not be updated and a 400 status code will be returned.  An attempt to update the capacity to a value less than the current number of aircrafts in a hangar will fail with a 400 status code. |
| Failure | 404 Not Found | No hangar with this hangarId exists. |
| Failure | 500 Internal Server Error | An internal server error has occurred. |

### Response Examples

#### Success

|  |
| --- |
| Status: 200 OK  Content-Type: application/json  {  “id”: 1234,  “name”: “The Arch Hangar”,  “city”: “Limestone”,  “state”: “Maine”,  “capacity”: 3,  “aircrafts”: [],  “self”: “https://appspot.com/hangars/1234”  } |

#### Failure

|  |
| --- |
| Status: 415 Unsupported Media Type  Content-Type: application/json  {  “Error”: “This endpoint only accepts application/json”  } |

#### Failure

|  |
| --- |
| Status: 406 Not Acceptable  Content-Type: application/json  {  “Error”: “This endpoint only serves application/json”  } |

#### Failure

|  |
| --- |
| Status: 400 Bad Request  Content-Type: application/json  {  “Error”: “Name attribute is invalid”  } |

#### Failure

|  |
| --- |
| Status: 400 Bad Request  Content-Type: application/json  {  “Error”: “City attribute is invalid”  } |

#### Failure

|  |
| --- |
| Status: 400 Bad Request  Content-Type: application/json  {  “Error”: “State attribute is invalid”  } |

#### Failure

|  |
| --- |
| Status: 400 Bad Request  Content-Type: application/json  {  “Error”: “Capacity attribute is invalid”  } |

#### Failure

|  |
| --- |
| Status: 400 Bad Request  Content-Type: application/json  {  “Error”: “Hangar capacity cannot be less than the current number of aircrafts in the hangar”  } |

#### Failure

|  |
| --- |
| Status: 404 Not Found  Content-Type: application/json  {  “Error”: “No hangar with this hangarId exists”  } |

#### Failure

|  |
| --- |
| Status: 500 Internal Server Error  Content-Type: application/json  {  “Error”: “An internal server error has occurred”  } |

# Fully Update a Hangar

Allows you to fully update the attributes of an already created hangar.

|  |
| --- |
| PUT /hangars/:hangarId |

## Request

### Media Type(s)

application/json

### Authorization

None

### Path Parameters

|  |  |
| --- | --- |
| **Name** | **Description** |
| hangarId | The ID of the hangar |

### Request Body

Required

### Request Body Format

JSON

### Request JSON Attributes

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Description** | **Notes** | **Required?** |
| name | The name of the hangar. | The minimum permissible string length of a hangar name is 1 character. The maximum permissible string length for a hangar name is 255 characters.  All hangar name must use characters that match the regular expression: ^[A-Za-z0-9 ./,-]+$ | Yes |
| city | The United States city that the hangar is located in. | The minimum permissible string length of a hangar city is 1 character. The maximum permissible string length for a hangar city is 255 characters.  All hangar cities must use characters that match the regular expression: ^[A-Za-z .,]+$ | Yes |
| state | The United States state that the hangar is located in. | The minimum permissible string length of a hangar state is 1 character. The maximum permissible string length for a hangar state is 255 characters.  All hangar states must use characters that match the regular expression: ^[A-Za-z ]+$ | Yes |
| capacity | The maximum number of aircrafts that the hangar can hold. | The smallest permissible capacity value is 1. The largest permissible capacity value is 5. | Yes |

\*Note: Any additional attributes provided in the request body that aren’t name, city, state, or capacity will be ignored.

### Request Body Example

|  |
| --- |
| Content-Type: application/json  {  “model”: “The Arch Hangar”,  “city”: “Limestone”,  “state”: “Maine”  “capacity”: 2  } |

## Response

### Media Type(s)

application/json

### Response Body Format

JSON

### Response Statuses

|  |  |  |
| --- | --- | --- |
| **Outcome** | **Status Code** | **Notes** |
| Success | 200 OK |  |
| Failure | 415 Unsupported Media Type | If the media type of the content provided in the request body is not JSON and the associated Content-Type request header is not application/json, the hangar will not be updated and a 415 status code will be returned. |
| Failure | 406 Not Acceptable | If the Accept header in the request is not application/json, the hangar will not be updated and a 406 status code will be returned.  Note: If the client has an Accept header set to \*/\*, i.e., any MIME type, the server will serve application/json in the response. The server assumes the client is indifferent to a specific media type as one was not explicitly set in the Accept header. |
| Failure | 400 Bad Request | If any of the 4 required attributes in the request are missing or misconfigured, the hangar will not be updated and a 400 status code will be returned.  An attempt to update the capacity to a value less than the current number of aircrafts in a hangar will fail with a 400 status code. |
| Failure | 404 Not Found | No hangar with this hangarId exists. |
| Failure | 500 Internal Server Error | An internal server error has occurred. |

### Response Examples

#### Success

|  |
| --- |
| Status: 200 OK  Content-Type: application/json  {  “id”: 1234,  “name”: “The Arch Hangar”,  “city”: “Limestone”,  “state”: “Maine”,  “capacity”: 2,  “aircrafts”: [],  “self”: “https://appspot.com/hangars/1234”  } |

#### Failure

|  |
| --- |
| Status: 415 Unsupported Media Type  Content-Type: application/json  {  “Error”: “This endpoint only accepts application/json”  } |

#### Failure

|  |
| --- |
| Status: 406 Not Acceptable  Content-Type: application/json  {  “Error”: “This endpoint only serves application/json”  } |

#### Failure

|  |
| --- |
| Status: 400 Bad Request  Content-Type: application/json  {  “Error”: “Name attribute is missing or invalid”  } |

#### Failure

|  |
| --- |
| Status: 400 Bad Request  Content-Type: application/json  {  “Error”: “City attribute is missing or invalid”  } |

#### Failure

|  |
| --- |
| Status: 400 Bad Request  Content-Type: application/json  {  “Error”: “State attribute is missing or invalid”  } |

#### Failure

|  |
| --- |
| Status: 400 Bad Request  Content-Type: application/json  {  “Error”: “Capacity attribute is missing or invalid”  } |

#### Failure

|  |
| --- |
| Status: 400 Bad Request  Content-Type: application/json  {  “Error”: “Hangar capacity cannot be less than the current number of aircrafts in the hangar”  } |

#### Failure

|  |
| --- |
| Status: 404 Not Found  Content-Type: application/json  {  “Error”: “No hangar with this hangarId exists”  } |

#### Failure

|  |
| --- |
| Status: 500 Internal Server Error  Content-Type: application/json  {  “Error”: “An internal server error has occurred”  } |

# Delete a Hangar

Allows you to delete a Hangar. All aircrafts parked in the hangar will be automatically marked as being in flight when the hangar is deleted.

|  |
| --- |
| DELETE /hangars/:hangarId |

## Request

### Media Type(s)

None

### Path Parameters

|  |  |
| --- | --- |
| **Name** | **Description** |
| hangarId | The ID of the hangar |

### Request Body

None

## Response

### Media Type(s)

Success: None

Failure: application/json

### Response Body Format

Success: No Body

Failure: JSON

### Response Statuses

|  |  |  |
| --- | --- | --- |
| **Outcome** | **Status Code** | **Notes** |
| Success | 204 No Content |  |
| Failure | 404 Not Found | No hangar with this hangarId exists. |
| Failure | 500 Internal Server Error | An internal server error has occurred. |

### Response Examples

#### Success

|  |
| --- |
| Status: 204 No Content |

#### Failure

|  |
| --- |
| Status: 404 Not Found  Content-Type: application/json  {  “Error”: “No hangar with this hangarId exists”  } |

#### Failure

|  |
| --- |
| Status: 500 Internal Server Error  Content-Type: application/json  {  “Error”: “An internal server error has occurred”  } |