API Guide

(*Song Store App*)

Version	Date	Author	Description
1.0	14-Sept-2017	Ting Ting Lin	Initial draft
1.1	21-Sept-2017	Ting Ting Lin	Complete details

Index

Methods

1. Login

Authenticate the user with the system and obtain the authToken

Request

Method	URL
POST	api/login/

Type	Params	Required	Values
HEAD	apiKey	Required	string
POST	username	Required	string
POST	password	Required	string

apiKey

`apiKey` must be sent with all client requests. The apiKey helps the server to validate the request source.

Response

Status	Response

200	{
	"apiKey": <authKey>
	}
	
	authKey (**string**) - all further API calls must have this key in header

403	{"error":"API key is missing."}
400	{"error":"Please provide username."}
400	{"error":"Please provide password."}

401	{"error":"Invalid API key."}
401	{"error":"Incorrect username or password."}
500	{"error":"Something went wrong. Please try again later."}

2. Retrieve all songs

Get the list of all songs

Request

Method	URL
GET	api/songs

Type	Params	Required	Values
HEAD GET GET GET GET GET GET	apiKey songName artists album genre songwriters releasedDate	Required Optional Optional Optional Optional Optional Optional	string [“”|string] [[“”]|string array] [{“Album Name”:“”}|Dictionary] [[“”]|string array] [[“”]|string array] [“”|string]

authKey

The `authKey` that was given in response to `/api/login`

songName

The `songName` is an optional string parameter that will return a list of all songs with matching song name if given

artists

The `artists` is an optional array parameter that will return a list of all songs with matching artists if given

album

The `album` is an optional dictionary parameter that will return a list of all songs with matching album if given

genre

The `genre` is an optional array parameter that will return a list of all songs with matching genre type if given

songwriters

The `songwriters` is an optional array parameter that will return a list of all songs with matching songwriters if given

releasedDate

The `releasedDate` is an optional string parameter that will return a list of all songs with matching released date if given

Response

Status	Response

200	**Response will be an object containing the list of songs(array). Each iter in the song has the following structure.**
	{
	    "songID": 10,
	    "songName": "Hello",
	    "artists": [
	        "Adele"
	    ],
	    "album": {
	        "Album Name": "25"
	    },
	    "genre": [
	        "Pop",
	        "Soul"
	    ],
	    "songWriters": [
	        "Adele Adkins",
	        "Greg Kurstin"
	    ],
	    "releasedDate": "23 October 2015"
	}
	An example response is:-
	{
	    "songs": [
	        {
	            "songID": 10,
	            "songName": "Hello",

	"artists": [
	                "Adele"
	            ],
	            "album": {
	                "Album Name": "25"
	            },
	            "genre": [
	                "Pop",
	                "Soul"
	            ],
	            "songWriters": [
	                "Adele Adkins",
	                "Greg Kurstin"
	            ],
	            "releasedDate": "23 October 2015"
	        }
	    ],
	}

401	{"error":"Invalid API key."}
404	{"error":"Not Found."}
500	{"error":"Something went wrong. Please try again later."}

3. Retrieve song Details

Get detail info of one specific song

Request

Method	URL
GET	api/songs/{songID}

Type	Params	Required	Values
HEAD GET	apiKey songID	Required Required	string number

authKey

The `authKey` that was given in response to `/api/login`

songID

The `songID` is the only unique identifier used to retrieve one specific song

Response

Status	Response

200	**Response will be an object containing the detail info of one song. Eac item in the song has the following structure.**
	
	{
	    "songID": 10,
	    "songName": "Hello",
	    "artists": [
	        "Adele"
	    ],
	    "album": {
	        "Album Name": "25"
	    },

	"genre": [
	        "Pop",
	        "Soul"
	    ],
	    "songWriters": [
	        "Adele Adkins",
	        "Greg Kurstin"
	    ],
	    "releasedDate": "23 October 2015"
	}
	
	An example response is:-
	{
	    "songID": 10,
	    "songName": "Hello",
	    "artists": [
	        "Adele"
	    ],
	    "album": {
	        "Album Name": "25"
	    },
	    "genre": [
	        "Pop",
	        "Soul"
	    ],
	    "songWriters": [
	        "Adele Adkins",
	        "Greg Kurstin"
	    ],
	    "releasedDate": "23 October 2015"
	}

400	{"error":"Missing songID."}
401	{"error":"Invalid API key."}
404	{"error":"Not Found."}
500	{"error":"Something went wrong. Please try again later."}

4. Create a new song

Add new song to song database

Request

Method	URL
POST	api/songs

Type	Params	Required	Values
HEAD	apiKey	Required	string
POST	songName	Required	string
POST	artists	Required	string array
POST	album	Required	Dictionary
POST	genre	Required	string array
POST	songwriters	Required	string array
POST	releasedDate	Required	string

authKey
The `authKey` that was given in response to `/api/login`

songName
The `songName` is a required string parameter that will create song with song name

artists
The `artists` is a required array parameter that will create song with artists

album
The `album` is a required dictionary parameter that will create song with album

genre
The `genre` is a required array parameter that will create song with genre type

songwriters
The `songwriters` is a required array parameter that will create song with songwriters

releasedDate

The `releasedDate` is a required string parameter that will create song with released date

Response

<table>
<tr><th>Status</th><th>Response</th></tr>
<tr><td>200</td><td>

Response will be an object containing the detail info of the song. Each item in the song has the following structure.

```
{
    "songID": 10,
    "songName": "Hello",
    "artists": [
        "Adele"
    ],
    "album": {
        "Album Name": "25"
    },
    "genre": [
        "Pop",
        "Soul"
    ],
    "songWriters": [
        "Adele Adkins",
        "Greg Kurstin"
    ],
    "releasedDate": "23 October 2015"
}
```

An example response is:-

```
{
    "songID": 10,
    "songName": "Hello",
    "artists": [
        "Adele"
    ],
    "album": {
        "Album Name": "25"
    },
    "genre": [
        "Pop",
```

</td></tr>
</table>

	␣␣␣␣␣␣␣␣"Soul" ␣␣␣␣], ␣␣␣␣"songWriters": [␣␣␣␣␣␣␣␣"Adele Adkins", ␣␣␣␣␣␣␣␣"Greg Kurstin" ␣␣␣␣], ␣␣␣␣"releasedDate": "23 October 2015" }
201	{"success":"Create successfully!"}
400	{"error":"Missing songName."}
400	{"error":"Missing artists."}
400	{"error":"Missing album."}
400	{"error":"Missing genre."}
400	{"error":"Missing songWriters."}
400	{"error":"Missing releasedDate."}
401	{"error":"Invalid API key."}
404	{"error":"Not Found."}
409	{"error":"Conflict."}
500	{"error":"Something went wrong. Please try again later."}

5. Update a song

Update a specific song

Request

Method	URL
PUT	api/songs/{songID}

Type	Params	Required	Values
HEAD	apiKey	Required	string
PUT	songID	Required	number
PUT	songName	Optional	[“”|string]
PUT	artists	Optional	[[“”]|string array]
PUT	album	Optional	[{“Album Name”:“”}|Dictionary
PUT	genre	Optional	[[“”]|string array]
PUT	songwriters	Optional	[[“”]|string array]
PUT	releasedDate	Optional	[“”|string]

authKey

The `authKey` that was given in response to `/api/login`

songID

The `songID` is the only unique identifier used to update one specific song

songName

The `songName` is an optional string parameter that will update song with song name if given

artists

The `artists` is an optional array parameter that will update song with artists if given

album

The `album` is an optional dictionary parameter that will update song with album if given

genre

The `genre` is an optional array parameter that will update song with genre type if given

`songwriters`

The `songwriters` is an optional array parameter that will update song with songwriters if given

`releasedDate`

The `releasedDate` is an optional string parameter that will update song with released date if given

Response

<table>
<tr><th>Status</th><th>Response</th></tr>
<tr><td>200</td><td>

Response will be an object containing the detail info of one song. Eac item in the song has the following structure.

<pre>
{
    "songID": 10,
    "songName": "Hello",
    "artists": [
        "Adele"
    ],
    "album": {
        "Album Name": "25"
    },
    "genre": [
        "Pop",
        "Soul"
    ],
    "songWriters": [
        "Adele Adkins",
        "Greg Kurstin"
    ],
    "releasedDate": "23 October 2015"
}
</pre>

An example response is:-

<pre>
{
    "songID": 10,
    "songName": "Hello",
    "artists": [
        "Adele"
</pre>

</td></tr>
</table>

	],
	"album": {
	"Album Name": "25"
	},
	"genre": [
	"Pop",
	"Soul"
	],
	"songWriters": [
	"Adele Adkins",
	"Greg Kurstin"
	],
	"releasedDate": "23 October 2015"
	}

400	{"error":"Missing songID."}
401	{"error":"Invalid API key."}
404	{"error":"Not Found."}
500	{"error":"Something went wrong. Please try again later."}

6. Delete a song

Delete a song

Request

Method	URL
`DELETE`	`api/songs/{songID}`

Type	Params	Required	Values
HEAD DELETE	auth_key songID	Required Required	string number

`authKey`
The `authKey` that was given in response to `/api/login`

`songID`
The `songID` is the only unique identifier used to delete one specific song

Response

Status	Response

200	**An array containing the ID’s of song to delete is given**
	
	`Example response:-`
	
	`{"deletions":[10,11,40]}`

400	`{"error":"Please specify songID."}`
401	`{"error":"Invalid Auth key."}`

500	{"error":"Something went wrong. Please try again later."}

7. Retrieve all reviews

Get the list of all reviews

Request

Method	URL
GET	api/reviews

Type	Params	Required	Values
HEAD	apiKey	Required	string
GET	linkedSongID	Optional	[null|number]
GET	reviewDate	Optional	[“”|string]
GET	userName	Optional	[“”|string]
GET	comment	Optional	[“”|string]

authKey
The `authKey` that was given in response to `/api/login`

linkSongID
The `linkSongID` is an optional parameter that will return list of reviews with matching song ID if given

reviewDate
The `reviewDate` is an optional string parameter that will return list of reviews with matching date if given

userName
The `userName` is an optional string parameter that will return list of reviews with matching userName if given

comment
The `comment` is an optional string parameter that will return list of reviews with matching comment

Response

Status	Response

200	**Response will be an object containing the list of reviews(array). Each item in the review has the following structure.**
	
	{
	"reviewID": 10,
	"linkedSongID": 10,
	"reviewDate": "2017-05-11T11:30:40.000Z",
	"userName": "User1234",
	"comment": "This song is so good!"
	}
	
	An example response is:
	{
	"reviews": [
	{
	"reviewID": 10,
	"linkedSongID": 10,
	"reviewDate": "2017-05-11T11:30:40.000Z",
	"userName": "User1234",
	"comment": "This song is so good!"
	}
	],
	}

401	{"error":"Invalid API key."}
404	{"error":"Not Found."}
500	{"error":"Something went wrong. Please try again later."}

8. Retrieve review Details

Get detail info of one specific review

Request

Method	URL
GET	api/reviews/{reviewID}

Type	Params	Required	Values
HEAD GET	apiKey reviewID	Required Required	string number

authKey
The `authKey` that was given in response to /api/login

reviewID
The `reviewID` is the only unique identifier used to retrieve one specific review

Response

<table>
<tr><th>Status</th><th>Response</th></tr>
<tr><td>200</td><td><b>Response will be an object containing the detail info of one review. Each item in the review has the following structure.</b>
<pre>
{
    "reviewID": 10,
    "linkedSongID": 10,
    "reviewDate": "2017-05-11T11:30:40.000Z",
    "userName": "User1234",
    "comment": "This song is so good!"
}
</pre>
<b>An example response is:</b></td></tr>
</table>

	{
	"reviewID": 10,
	"linkedSongID": 10,
	"reviewDate": "2017-05-11T11:30:40.000Z",
	"userName": "User1234",
	"comment": "This song is so good!"
	}

400	{"error":"Missing songID."}
401	{"error":"Invalid API key."}
404	{"error":"Not Found."}
500	{"error":"Something went wrong. Please try again later."}

9. Create a new review

Add new review to review database

Request

Method	URL
POST	api/reviews

Type	Params	Required	Values
HEAD POST POST POST POST	apiKey linkedSongID reviewDate userName comment	Required Required Required Required Required	string number string string string

authKey
The `authKey` that was given in response to `/api/login`

linkSongID
The `linkSongID` is a required parameter that will create review linking with specific song

reviewDate
The `reviewDate` is a required string parameter that will create review with review date

userName
The `userName` is a required string parameter that will create review with user name

comment
The `comment` is a required string parameter that will create review with comment

Response

Status	Response

200	**Response will be an object containing the detail info of the review. Each item in the review has the following structure..**
	{
	"reviewID": 10,
	"linkedSongID": 10,
	"reviewDate": "2017-05-11T11:30:40.000Z",
	"userName": "User1234",
	"comment": "This song is so good!"
	}
	An example response is:
	{
	"reviewID": 10,
	"linkedSongID": 10,
	"reviewDate": "2017-05-11T11:30:40.000Z",
	"userName": "User1234",
	"comment": "This song is so good!"
	}

201	`{"success":"Create successfully!"}`
400	`{"error":"Missing linkedSongID."}`
400	`{"error":"Missing reviewDate."}`
400	`{"error":"Missing userName."}`
400	`{"error":"Missing comment."}`
401	`{"error":"Invalid API key."}`
404	`{"error":"Not Found."}`
409	`{"error":"Conflict."}`
500	`{"error":"Something went wrong. Please try again later."}`

10. Update a review

Update a specific review

Request

Method	URL
PUT	`api/reviews/{reviewID}`

Type	Params	Required	Values
HEAD	apiKey	Required	string
PUT	reivewID	Required	number
PUT	linkedSongID	Optional	[null|number]
PUT	reviewDate	Optional	[“”|string]
PUT	userName	Optional	[“”|string]
PUT	comment	Optional	[“”|string]

authKey
The `authKey` that was given in response to `/api/login`

reviewID
The `reviewID` is the only unique identifier used to update one specific review

songID
The `songID` is the only unique identifier used to update one specific song

linkSongID
The `linkSongID` is an optional parameter that will update specific review with matching song ID if given

reviewDate
The `reviewDate` is an optional string parameter that will update specific review with matching date if given

userName
The `userName` is an optional string parameter that will update specific review with matching userName if given

`comment`

The `comment` is an optional string parameter that will update specific review with matching comment

Response

Status	Response

200	**Response will be an object containing the detail info of one review. Each item in the review has the following structure.**
	
	{
	    "reviewID": 10,
	    "linkedSongID": 10,
	    "reviewDate": "2017-05-11T11:30:40.000Z",
	    "userName": "User1234",
	    "comment": "This song is so good!"
	}
	
	An example response is:
	{
	    "reviewID": 10,
	    "linkedSongID": 10,
	    "reviewDate": "2017-05-11T11:30:40.000Z",
	    "userName": "User1234",
	    "comment": "This song is so good!"
	}

400	{"error":"Missing songID."}
401	{"error":"Invalid API key."}
404	{"error":"Not Found."}
500	{"error":"Something went wrong. Please try again later."}

11. Delete a review

Delete a review

Request

Method	URL
DELETE	`api/reviews/{reviewID}`

Type	Params	Required	Values
HEAD DELETE	auth_key reviewID	Required Required	string number

`authKey`
The `authKey` that was given in response to `/api/login`

`reviewID`
The `reviewID` is the only unique identifier used to delete one specific review

Response

Status	Response

200	**An array containing the ID's of review to delete is given**
	`Example response:-`
	`{"deletions":[10,11,40]}`

400	`{"error":"Please specify reviewID."}`
401	`{"error":"Invalid Auth key."}`
500	`{"error":"Something went wrong. Please try again later."}`

Glossary

Conventions

- **Client** - Client application.
- **Status** - HTTP status code of response.
- All the possible responses are listed under ‘Responses’ for each method. Only one of them is issued per request server.
- All response are in JSON format.
- All request parameters are mandatory unless explicitly marked as `[optional]`
- The type of values accepted for a *request* parameter are shown the the values column like this `[10|<any number>]` .The `|` symbol means *OR*. If the parameter is `[optional]`, the default value is shown in blue bold text, as **`10`** is written in `[10|<any number>]`.

Status Codes

All status codes are standard HTTP status codes. The below ones are used in this API.

`2XX` - Success of some kind
`4XX` - Error occurred in client’s part
`5XX` - Error occurred in server’s part

Status Code	Description
`200`	`OK`
`201`	`Created`
`202`	`Accepted (Request accepted, and queued for execution)`
`400`	`Bad request`
`401`	`Authentication failure`
`403`	`Forbidden`
`404`	`Resource not found`

405	Method Not Allowed
409	Conflict
412	Precondition Failed
413	Request Entity Too Large
500	Internal Server Error
501	Not Implemented
503	Service Unavailable