No-Code App: Break that ETF!

Daniele Boschetti, Paula Palermo, Pascal Bobbià

DOWNLOAD IT HERE: https://daniele-boschettis-team.adalo.com/break-that-etf

Goal: Our objective was to create an app with ADALO able to display information about different selected ETFs.

Selection: The selection criterion was based on a python package that was able to display the first 15 components of these 7 ETFs; due to our limited budget, it was not possible to gain further information.

Structure: The App is built in the following way: once entering in it, the user can log in if already owning an account or sign up.

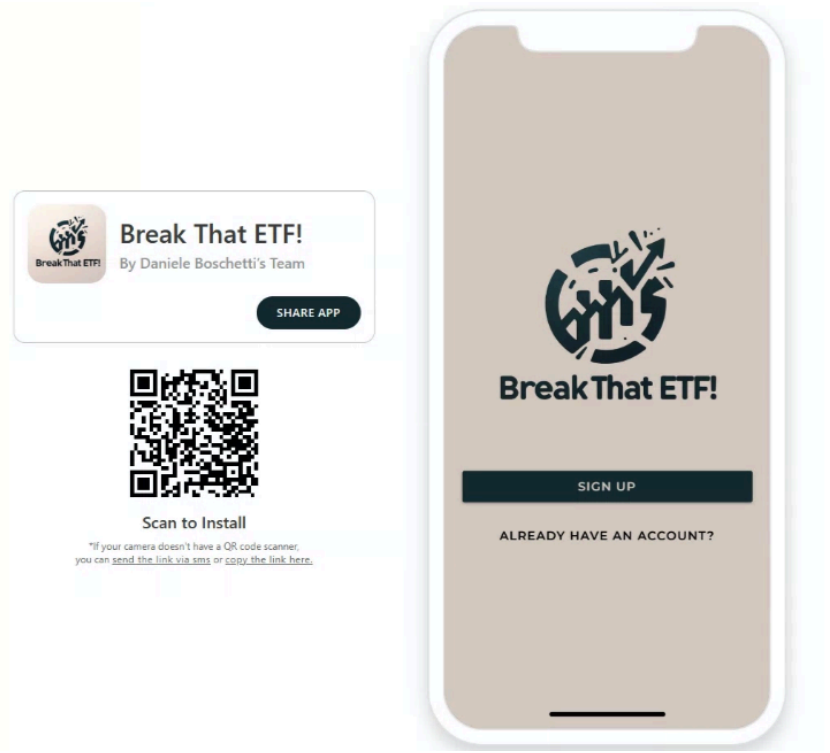
Once entered, the user will be directed to the **Home page**, which displays a search tab used for knowing if the app contains a certain ETF (the app actually contains only 7, but by having more it would be more practical), a card list containing the different ETFs (displayed by name and ticker) sorted alphabetically (for practicality causes); by clicking on "**Individual holdings**", the app will load another page containing the ETF holdings, while by clicking on "**More details**", the app will bring the user to a page containing a description of the ETF. Under the "**Description**" section, the user can see the ETF name, its, family fund logo, its category and a description on the way it operates (with this information being taken from the respective ETF Yahoo Finance page). The "**Performance**" page will show the name, logo and category too as well as a dashboard containing:

On the bottom of the page there will be a navigation panel displaying the pages "Home", "Favorites", "Search", ...

Favorites: The app features the option to select individual holdings and add them into the "**Favorites**" list. This is done by clicking on the "**Add to favorites**" button on the respective holding description page.

Profile: the user will be able to see its name, its email, a "**Log out**" button executing the action by clicking on it.

API: The API is proprietary and created with Airtable



INDEX

Section 1 - Overview

The market for ESG ETFs has experienced significant growth as more investors seek to align their investments with their values. However, there is increasing criticism that many ESG ETFs do not truly adhere to the highest standards of environmental, social, and governance criteria. Often, these funds include holdings that may be counterproductive to the sustainability goals they purport to support.

Key issues raised include:

- Lack of Transparency: Information about the specific holdings within ESG ETFs is not always readily available to the public, making it challenging for investors to make informed decisions.
- Greenwashing: Some ESG ETFs include companies engaged in harmful environmental or social practices, questioning the integrity and authenticity of the funds.
- Inconsistent Standards: The criteria for what constitutes an ESG investment can vary widely, leading to inconsistencies and potential misrepresentation of true sustainable investments.

The Solution: Break that ETF

Break that ETF addresses these issues by providing users with detailed insights into the individual holdings of various ESG ETFs. Our app enables users to:

1. View and analyze the specific ESG ETFs, including information about:
    - Market information, and price
    - Methodology
    - Issuer
    - ESG Score
    - Weighted Average Carbon Intensity (Tons of CO2e / $M Sales)

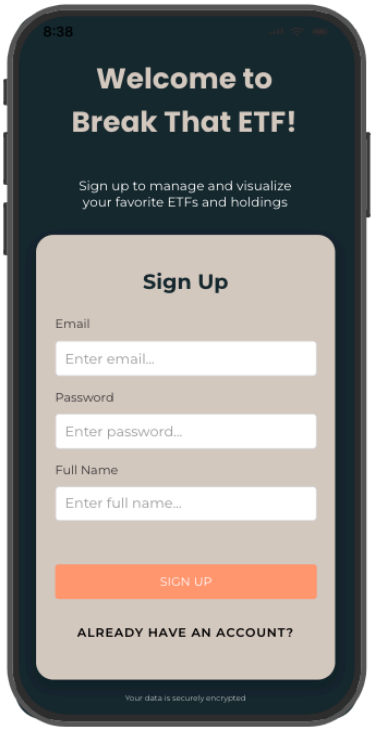
2. View and analyze the specific companies included in ESG ETFs
    - View top 15 holdings of the ESG ETF
    - Percentage of allocation of the specific stock
    - Market and price information about the individual holdings

3. Assess the alignment of these companies with the user's personal ESG criteria
    - Identify the holdings that are a better fit for user's specific ESG criteria and ballance with returns and market analysis
    - Create alternative portfolios based on their values and investment goals.

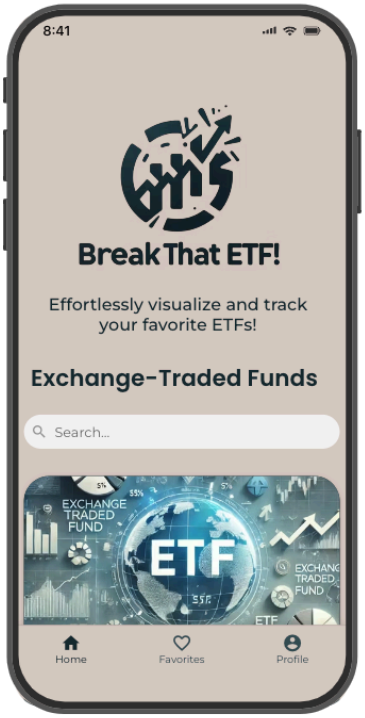
With Break that ETF, users are no longer at the mercy of broad and often opaque ESG ratings. Instead, they have the tools to make transparent and informed investment choices that truly reflect their commitment to sustainability and ethical governance.

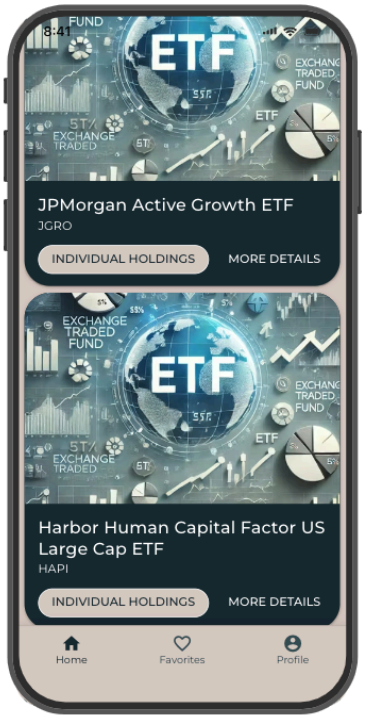
Section 2- Structure of the App

1. User Authentication: Upon launching the app, users are given the option to either log in if they already possess an account or sign up to create a new account.

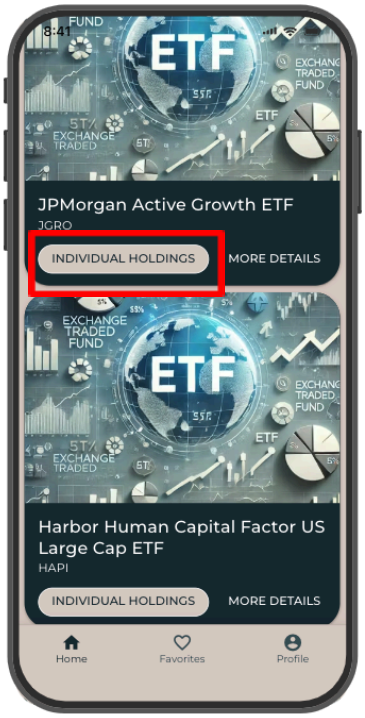


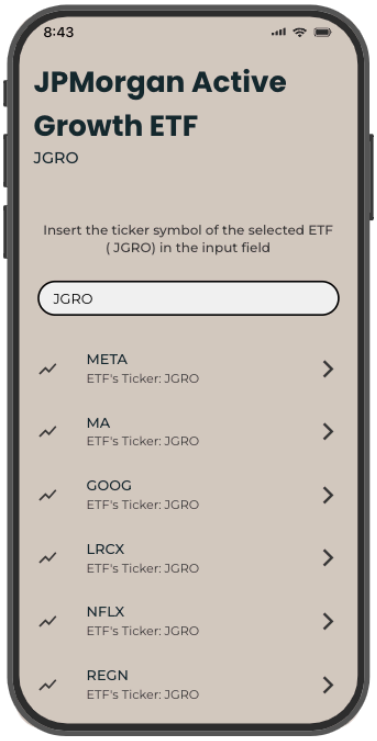
2. Home Page: Once authenticated, users are directed to the Home page, which contains the following features:
    - Search Tab: Allows users to search if the app contains a specific ETF (currently limited to 7 ETFs).
    - Card List: Displays the different ETFs available within the app, sorted alphabetically by name and ticker for practicality.
    - ETF Details



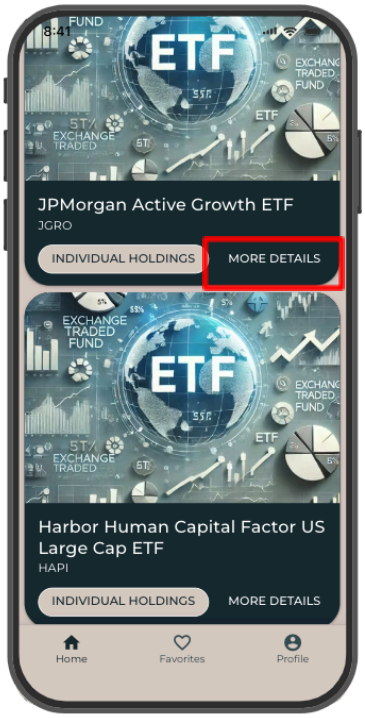


3. <u>Individual Holdings:</u> By clicking on the "Individual holdings" button, users are taken to a page displaying the individual holdings of the selected ETF.

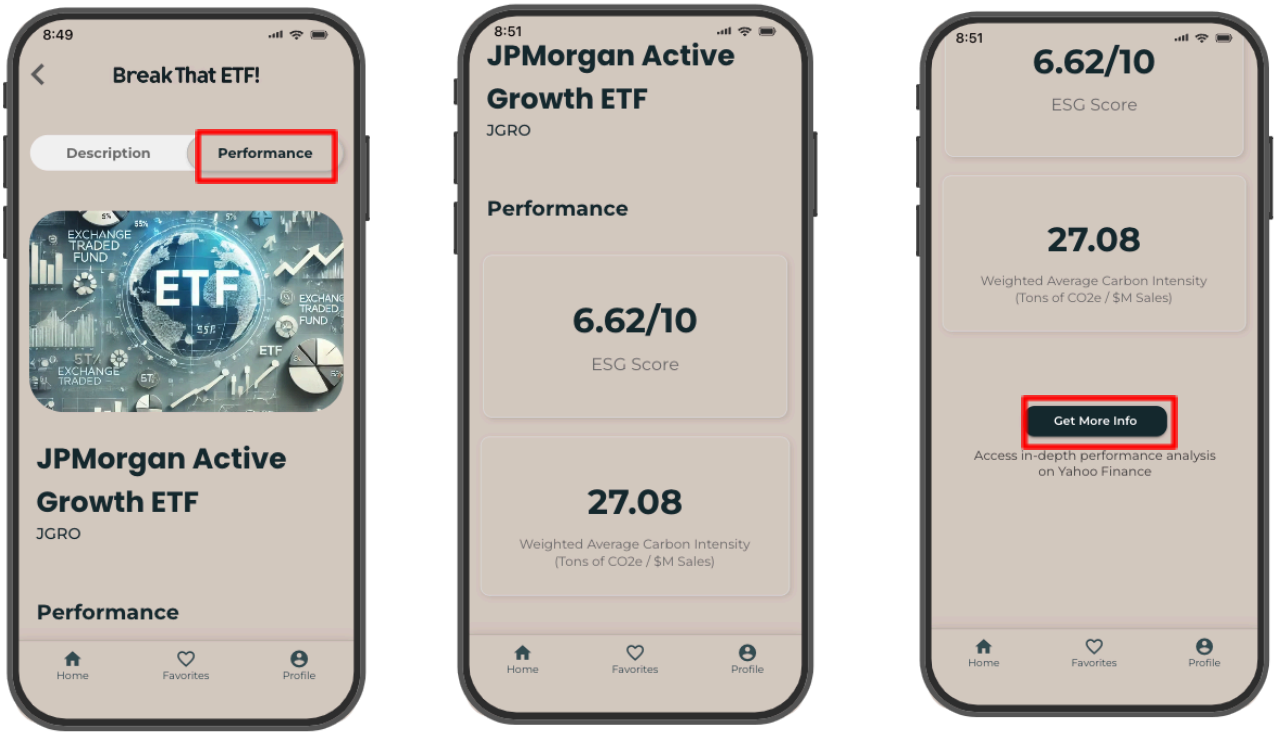




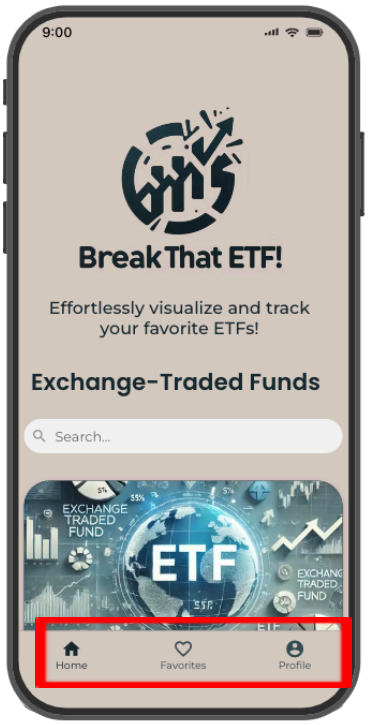
4. <u>More Details:</u> By clicking on the "More details" button, users are redirected to a page with a comprehensive description of the ETF, including:
    - ETF Name
    - Fund Family Logo
    - Category
    - Description: This section pulls information from the respective ETF's Yahoo Finance page.
    -



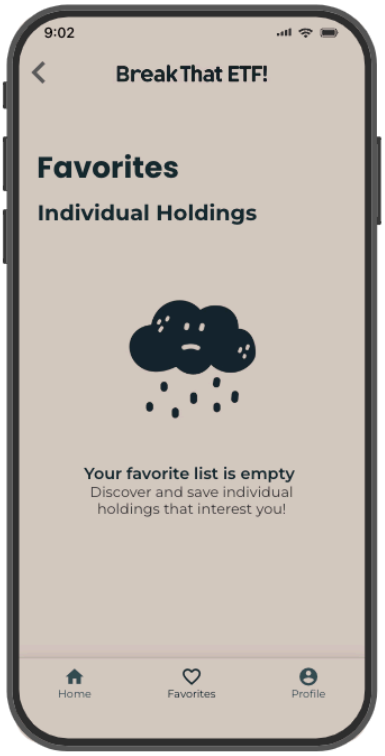
5. Performance Page: The “Performance” page includes:
    - Name, Logo, and Category: Basic details of the ETF.
    - ESG score and Weighted Average Carbon Intensity Index
    - Dashboard: Contains graphs and charts illustrating the performance metrics of the ETF.



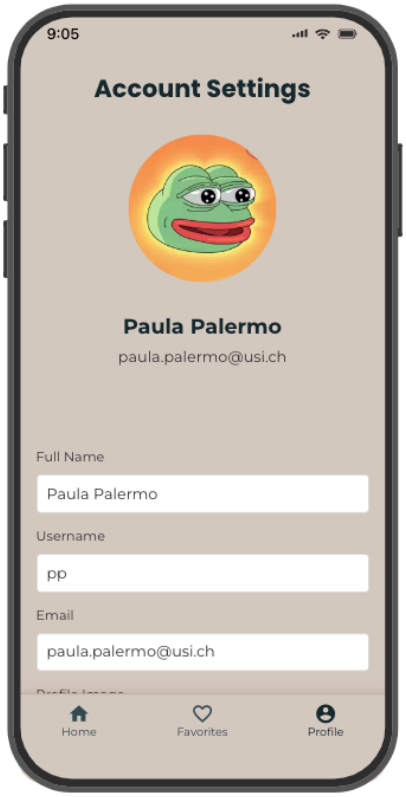
6. Navigation Panel: At the bottom of the page, a navigation panel provides quick access to the following sections:
    - Home
    - Favorites
    - Search
    - Additional sections (to be specified).

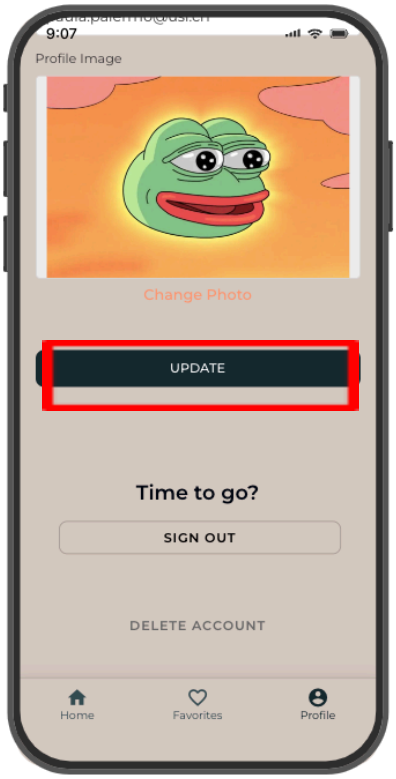


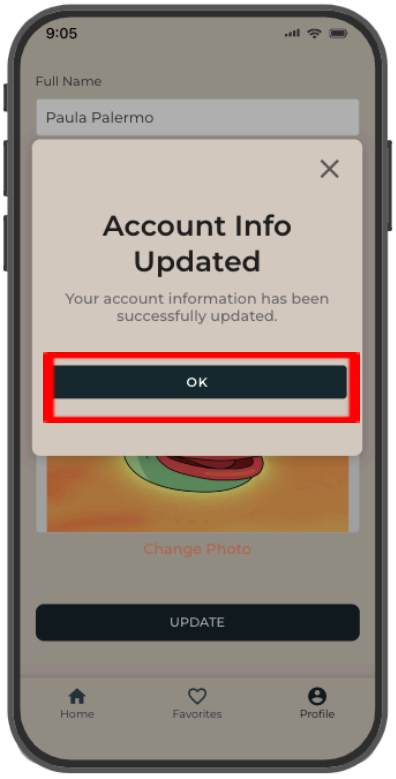
7. Favorites: The app allows users to select individual holdings and add them to a “Favorites” list. This can be done by clicking on the “Add to favorites” button found on the respective holding's description page.



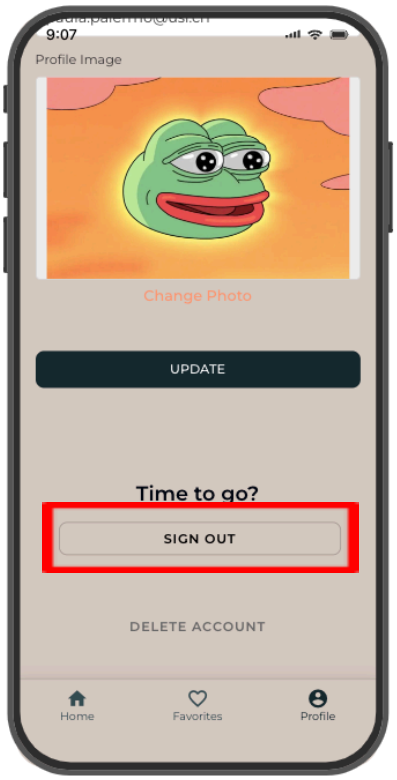
8. Profile: In the Profile section, users can view:
   - Name, Email, Username
   - Update profile picture







9. Log out: Sad to see you go! You can log out clicking in this button



Section 3 - API

The dataset used in the app was created and managed via Airtable, which facilitated the organization and retrieval of ETF holdings data. Let's explore in detail with the specific API documentation.

The ETFs API provides seamless integration of ETF data from Airtable with external systems, using REST semantics and JSON for encoding objects, and standard HTTP codes for signaling outcomes. The API is customized for your Airtable base with ID appWAUPTh7m86iVCo. Note that changes to field names or types in Airtable will alter the API interface correspondingly. Official and community-built API clients are available in various languages including JavaScript, Ruby, .NET, and Python.

1. API Key and Base URL

API key:

pat6H9F8GCasha2pT.b530e39a0857febab9a02d1815d17e7fd0ae746c8c2d3721ef12793460966a65

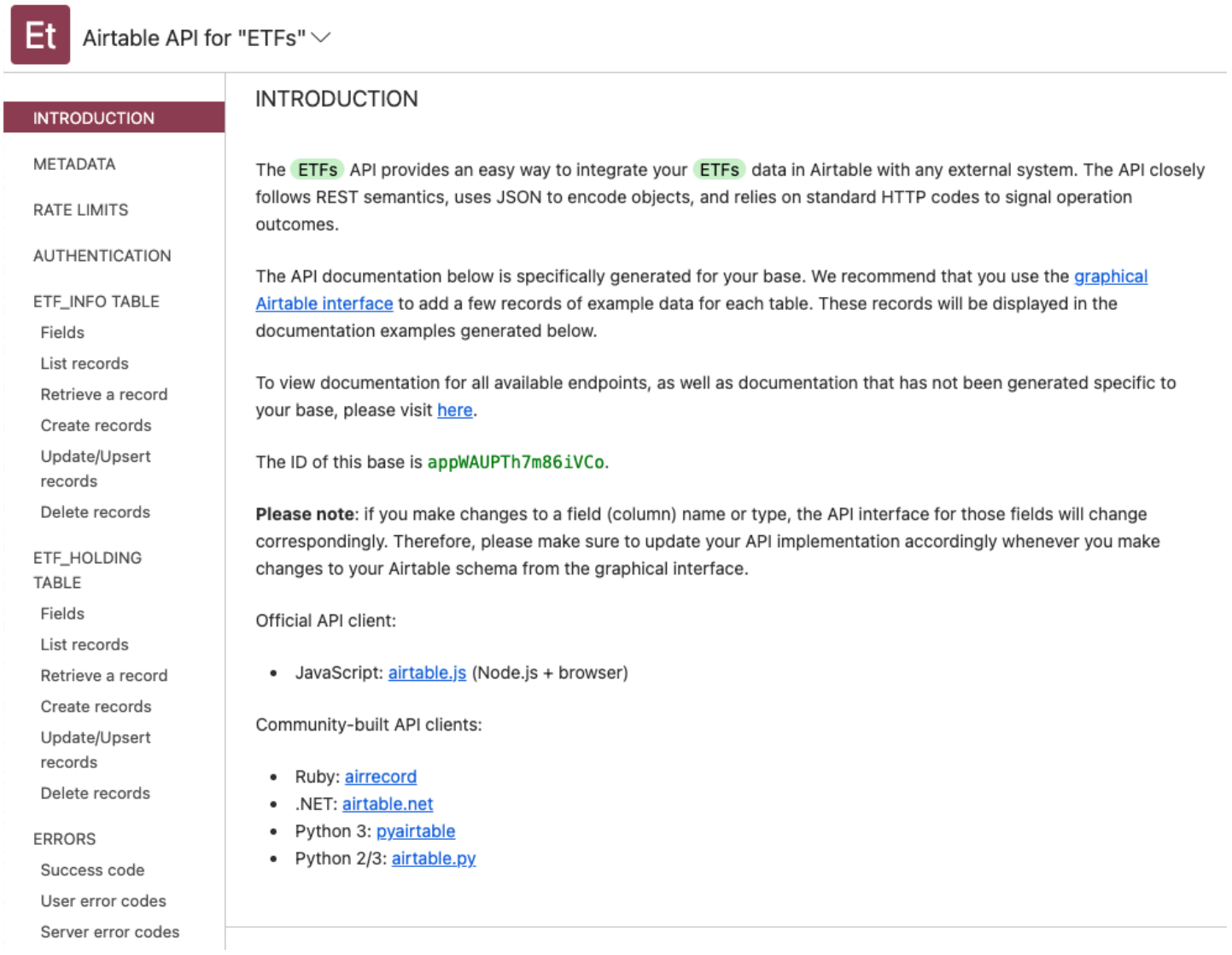
API Base URL:

https://api.airtable.com/v0/appWAUPTh7m86iVCo/ETF_Holding

https://api.airtable.com/v0/appWAUPTh7m86iVCo/ETF_Info

2. Operations

List Records: Issue a GET request to ETF_Info. Customize results with filters, sorting, and formatting via query parameters.
Retrieve Record: Use a GET request to the specific record endpoint. Note: Empty fields are not returned.
Create Records: Issue a POST request to ETF_Info. Include up to 10 record objects in the request body.
Update/Upsert Records: Use PATCH (to update) or PUT (to overwrite). Include the record ID and fields to update.
Delete Records: Use a DELETE request to ETF_Info or the specific record endpoint.



3. ETF_INFO_TABLE

Et Airtable API for "ETFs"

ETF_INFO TABLE

The id for ETF_Info is tblyIpA285iIS7DtR. Table ids and table names can be used interchangeably in API requests. Using table ids means table name changes do not require modifications to your API request.

Fields

Each record in the ETF_Info table contains the following fields:

Field names and field ids can be used interchangeably. Using field ids means field name changes do not require modifications to your API request. We recommend using field ids over field names where possible, to reduce modifications to your API request if the user changes the field name later.

FIELD NAME	FIELD ID	TYPE	DESCRIPTION
ID	fldfUm14o8isJNxyW	Number	number An integer (whole number, e.g. 1, 32, 99). This field allows negative and positive numbers.
ETF_ticker	fldnIAYlHKO6r4Wft	Long text	string Multiple lines of text, which may contain "mention tokens", e.g. <airtable:mention id="menE1i9oBaGX3DseR">@Alex</airtable:mention>
Name	fld522dXp8hHgoBTo	Text	string A single line of text.
Issuer	fldPiOkW86g8V5RjW	Text	string A single line of text.

List ETF_Info records

To list records in `ETF_Info`, issue a **GET** request to the `ETF_Info` endpoint. Note that table names and table ids can be used interchangeably. Using table ids means table name changes do not require modifications to your API request.

Returned records do not include any fields with "empty" values, e.g. `""`, `[]`, or `false`.

You can filter, sort, and format the results with the following query parameters. Note that these parameters need to be URL encoded. You can use our API URL encoder tool to help with this. If you are using a helper library like Airtable.js, these parameters will be automatically encoded.

Note: Airtable's API only accepts request with a URL shorter than 16,000 characters. Encoded formulas may cause your requests to exceed this limit. To fix this issue you can instead make a POST request to `/v0/{baseId}/{tableIdOrName}/listRecords` while passing the parameters within the body of the request instead of the query parameters. See our support article on this for more information.

fields
`array of strings`
optional

Only data for fields whose names are in this list will be included in the result. If you don't need every field, you can use this parameter to reduce the amount of data transferred.

For example, to only return data from `ID` and `ETF_ticker`, send these two query parameters:

```
fields%5B%5D=%20ID&fields%5B%5D=ETF_ticker
```

You can also perform the same action with field ids (they can be found in the fields section):

```
fields%5B%5D=fldfUm14o8isJNxyW&fields%5B%5D=fldnIAYlHKO6r4Wft
```

Note: `%5B%5D` may be omitted when specifying multiple fields, but must always be included when specifying only a single field.

filterByFormula
`string`
optional

A formula used to filter records. The formula will be evaluated for each record, and if the result is not `0`, `false`, `""`, `NaN`, `[]`, or `#Error!` the record will be included in the response. We recommend testing your formula in the Formula field UI before using it in your API request.

If combined with the `view` parameter, only records in that view which satisfy the formula will be returned.

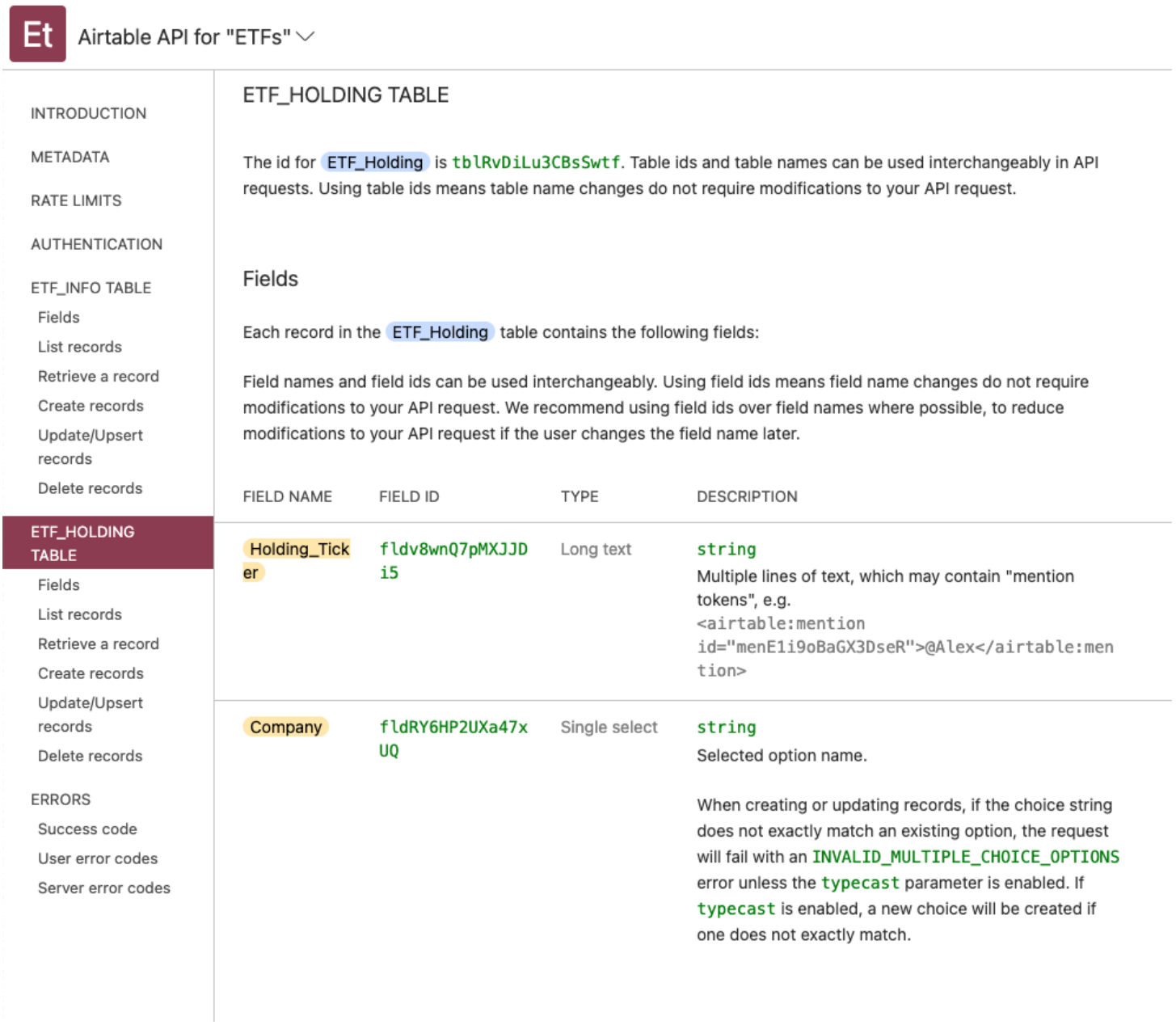
EXAMPLE REQUEST

```
curl "https://api.airtable.com/v0/appWAUPTh7m86iVCo/ETF_Holding?maxRecords=3&view=Grid%20view" \
  -H "Authorization: Bearer YOUR_SECRET_API_TOKEN"
```

EXAMPLE RESPONSE


```
{
    "records": [
        {
            "id": "recd2T4NSsaKkkbmH",
            "createdTime": "2024-06-21T12:38:41.000Z",
            "fields": {
                "Assets%": 8.1,
                "Company": "NVIDIA Corporation",
                "Holding_Ticker": "NVDA",
                "ETF_Ticker": "HAPI",
                "ETF_Name": "Harbor Human Capital Factor US Large Cap ETF",
                "Issuer": "Harbor",
                "Brand": "Harbor"
            }
        },
        {
            "id": "recUkUFObXaLGNQI6",
            "createdTime": "2024-06-21T12:38:41.000Z",
            "fields": {
                "Assets%": 5.38,
                "Company": "Apple Inc.",
                "Holding_Ticker": "AAPL",
                "ETF_Ticker": "HAPI",
                "ETF_Name": "Harbor Human Capital Factor US Large Cap ETF",
                "Issuer": "Harbor",
                "Brand": "Harbor"
            }
        },
        {
            "id": "rec0Ze6EeZAVCP0v7",
            "createdTime": "2024-06-21T12:38:41.000Z",
            "fields": {
                "Assets%": 5.01,
                "Company": "Microsoft Corporation",
                "Holding_Ticker": "MSFT",
                "ETF_Ticker": "HAPI",
                "ETF_Name": "Harbor Human Capital Factor US Large Cap ETF",
                "Issuer": "Harbor",
                "Brand": "Harbor"
            }
        }
```

4. ETF_HOLDINGS_TABLE



Airtable API for "ETFs"

INTRODUCTION
METADATA
RATE LIMITS
AUTHENTICATION
ETF_INFO TABLE
Fields
List records
Retrieve a record
Create records
Update/Upsert records
Delete records
ETF_HOLDING TABLE
Fields
List records
Retrieve a record
Create records
Update/Upsert records
Delete records
ERRORS
Success code
User error codes
Server error codes

ETF_HOLDING TABLE

The id for ETF_Holding is tblRvDiLu3CBsSwtf. Table ids and table names can be used interchangeably in API requests. Using table ids means table name changes do not require modifications to your API request.

Fields

Each record in the ETF_Holding table contains the following fields:

Field names and field ids can be used interchangeably. Using field ids means field name changes do not require modifications to your API request. We recommend using field ids over field names where possible, to reduce modifications to your API request if the user changes the field name later.

FIELD NAME	FIELD ID	TYPE	DESCRIPTION
Holding_Ticker	fldv8wnQ7pMXJJDi5	Long text	string Multiple lines of text, which may contain "mention tokens", e.g. <airtable:mention id="menE1i9oBaGX3DseR">@Alex</airtable:mention>
Company	fldRY6HP2UXa47xUQ	Single select	string Selected option name. When creating or updating records, if the choice string does not exactly match an existing option, the request will fail with an INVALID_MULTIPLE_CHOICE_OPTIONS error unless the typecast parameter is enabled. If typecast is enabled, a new choice will be created if one does not exactly match.

List ETF_Holding records

To list records in `ETF_Holding`, issue a **GET** request to the `ETF_Holding` endpoint. Note that table names and table ids can be used interchangeably. Using table ids means table name changes do not require modifications to your API request.

Returned records do not include any fields with "empty" values, e.g. `""`, `[]`, or `false`.

You can filter, sort, and format the results with the following query parameters. Note that these parameters need to be URL encoded. You can use our API URL encoder tool to help with this. If you are using a helper library like Airtable.js, these parameters will be automatically encoded.

Note: Airtable's API only accepts request with a URL shorter than 16,000 characters. Encoded formulas may cause your requests to exceed this limit. To fix this issue you can instead make a POST request to `/v0/{baseId}/{tableIdOrName}/listRecords` while passing the parameters within the body of the request instead of the query parameters. See our support article on this for more information.

fields
`array of strings`
optional

Only data for fields whose names are in this list will be included in the result. If you don't need every field, you can use this parameter to reduce the amount of data transferred.

For example, to only return data from `Holding_Ticker` and `Company`, send these two query parameters:

```
fields%5B%5D=Holding_Ticker&fields%5B%5D=Company
```

You can also perform the same action with field ids (they can be found in the fields section):

```
fields%5B%5D=fldv8wnQ7pMXJJDi5&fields%5B%5D=fldRY6HP2UXa47xUQ
```

Note: `%5B%5D` may be omitted when specifying multiple fields, but must always be included when specifying only a single field.

filterByFormula
`string`
optional

A formula used to filter records. The formula will be evaluated for each record, and if the result is not `0`, `false`, `""`, `NaN`, `[]`, or `#Error!` the record will be included in the response. We recommend testing your formula in the Formula field UI before using it in your API request.

If combined with the `view` parameter, only records in that view which satisfy the formula will be returned.

The formula must be encoded first before passing it as a value. You can use this tool to not only encode the formula but also create the entire url you need. For example, to only include records where `Holding_Ticker` isn't empty, pass in

EXAMPLE REQUEST

```
curl "https://api.airtable.com/v0/appWAUPTh7m86iVCo/ETF_Holding?maxRecords=3&view=Grid%20view" \
  -H "Authorization: Bearer YOUR_SECRET_API_TOKEN"
```

EXAMPLE RESPONSE


```
{
    "records": [
        {
            "id": "recd2T4NSsaKkkbmH",
            "createdTime": "2024-06-21T12:38:41.000Z",
            "fields": {
                "Holding_Ticker": "NVDA",
                "Company": "NVIDIA Corporation",
                "Assets%": 8.1,
                "ETF_Ticker": "HAPI",
                "ETF_Name": "Harbor Human Capital Factor US Large Cap ETF",
                "Issuer": "Harbor",
                "Brand": "Harbor"
            }
        },
        {
            "id": "recUkUFObXaLGNQI6",
            "createdTime": "2024-06-21T12:38:41.000Z",
            "fields": {
                "Holding_Ticker": "AAPL",
                "Company": "Apple Inc.",
                "Assets%": 5.38,
                "ETF_Ticker": "HAPI",
                "ETF_Name": "Harbor Human Capital Factor US Large Cap ETF",
                "Issuer": "Harbor",
                "Brand": "Harbor"
            }
        },
        {
            "id": "rec0Ze6EeZAVCP0v7",
            "createdTime": "2024-06-21T12:38:41.000Z",
            "fields": {
                "Holding_Ticker": "MSFT",
                "Company": "Microsoft Corporation",
                "Assets%": 5.01,
                "ETF_Ticker": "HAPI",
                "ETF_Name": "Harbor Human Capital Factor US Large Cap ETF",
                "Issuer": "Harbor",
                "Brand": "Harbor"
            }
        }
```

5. ERRORS

Et Airtable API for "ETFs"

ERRORS

The `ETFs` API follows HTTP status code semantics. 2xx codes signify success, 4xx mostly represent user error, 5xx generally correspond to a server error. The error messages will return a JSON-encoded body that contains `error` and `message` fields. Those will provide specific error condition and human-readable message to identify what caused the error.

Success code

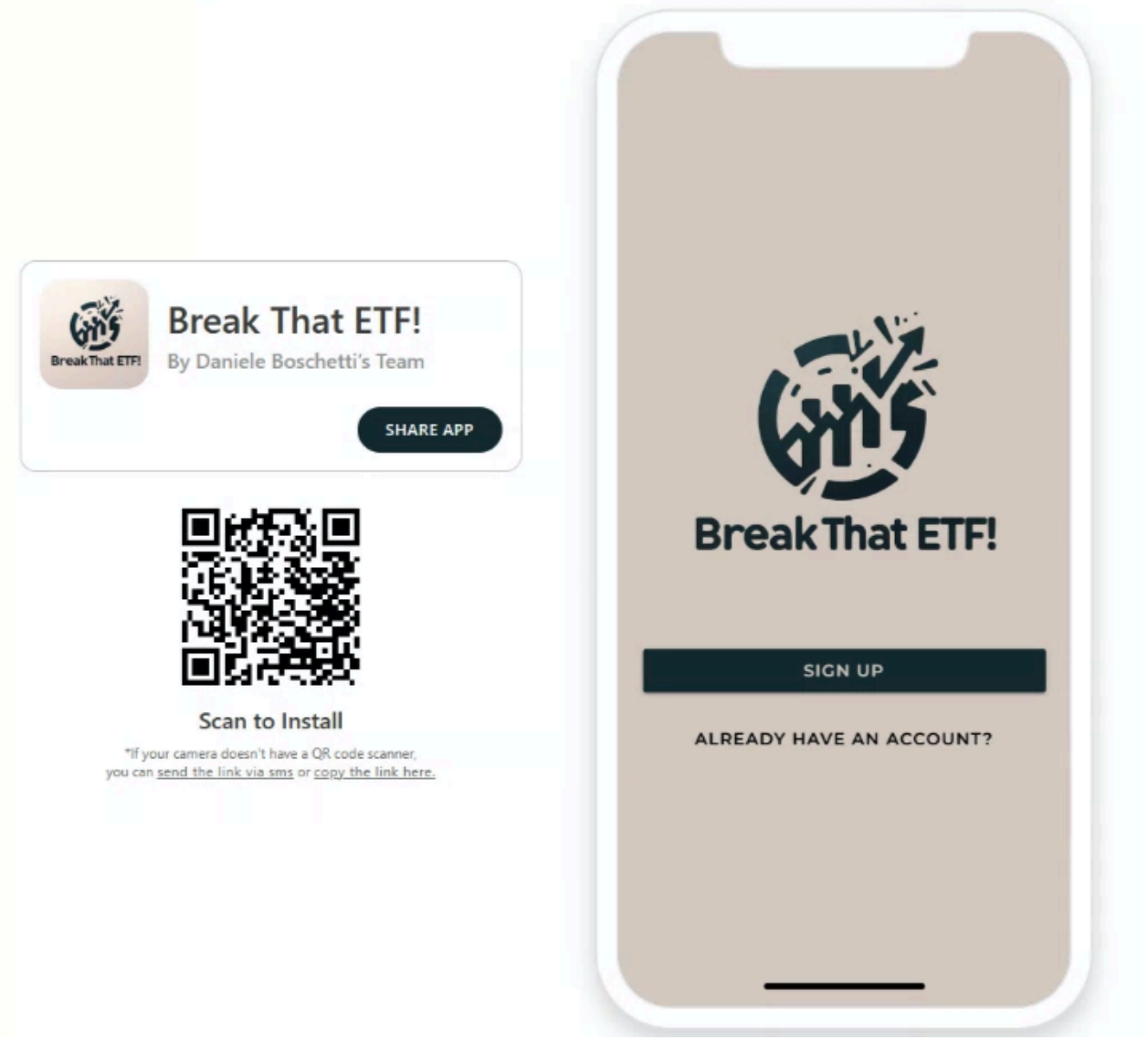
`200`	OK	Request completed successfully.

User error codes

These errors generally indicate a problem on the client side. If you are getting one of these, check your code and the request details.

`400`	Bad Request	The request encoding is invalid; the request can't be parsed as a valid JSON.
`401`	Unauthorized	Accessing a protected resource without authorization or with invalid credentials.
`402`	Payment Required	The account associated with the API key making requests hits a quota that can be increased by upgrading the Airtable account plan.
`403`	Forbidden	Accessing a protected resource with API credentials that don't have access to that resource.
`404`	Not Found	Route or resource is not found. This error is returned when the request hits an undefined route, or if the resource doesn't exist (e.g. has been deleted).
`413`	Request Entity Too Large	The request exceeded the maximum allowed payload size. You shouldn't encounter this under normal use.
`422`	Invalid Request	The request data is invalid. This includes most of the base-specific validations. You will receive a detailed error message and code pointing to the exact issue.

Thank you! 🚀



With love from: Daniele Boschetti, Paula Palermo, Pascal Bobbià

The use of Generative AI was applied to help us with design and also for creating and formatting this document. Chat GPT is the 4th element of this group.