



To-do App: Create a Stitch Backend

Author: Stitch Documentation Team

When completed, you will have built a Stitch backend app that handles the basic requirements of a To-do app.

Time required: 15 minutes

What You'll Need

There are no prerequisites for this guide.

Procedure

A. Create a MongoDB Stitch App

Estimated Time to Complete: ~8 minutes

1 Log into Atlas [↗](#).

To use MongoDB Stitch, you must be logged into MongoDB Atlas [↗](#). If you do not have an Atlas account, follow the instructions in the [Atlas documentation](#) to create an account [↗](#).

2 Create an Atlas cluster.

If you do not already have an Atlas cluster for use with MongoDB Stitch, create a cluster [↗](#).

Atlas provides a Free Tier M0 replica set as well as paid M10+ clusters. Free Tier deployments have restrictions as compared to paid M10+ deployments but will work for the purposes of this tutorial. For complete documentation on these restrictions, see [Atlas M0 \(Free Tier\), M2, and M5 Limitations](#) [↗](#).

3 Add a MongoDB Stitch application.

- In Atlas, click **Stitch Apps** in the left-hand navigation.
- Click the **Create New Application** button.
- Under **Application Name**, enter a name for your application. The name can only contain ASCII letters, numbers, underscores, and hyphens.

- d. Under **Link To Cluster**, select the MongoDB Atlas cluster that you'd like to use for your application.
- e. Click **Create**.

After you click **Create**, a new Stitch application will be created for you. The application will include a MongoDB service named `mongodb-atlas` that is linked to your cluster. This process may take several minutes.

Once your application has been created, you will be automatically redirected to the Stitch application console.

B. Define a Rule and Create a Schema

Estimated Time to Complete: ~6 minutes

MongoDB Stitch rules specify the read and write access for the fields in your collections. Schemas define the shape of the data in your collections.

1

Create a namespace for the `todo.items` collection

- a. In the left navigation pane, under **MongoDB Clusters**, click **Rules**.
- b. Next to your Atlas service name (typically `mongodb-atlas`), click the plus button.
- c. In the Add New Collection screen, provide the following values:

Property	Value
Database Name	<code>todo</code>
Collection Name	<code>items</code>
Select a Template	Users can only read and write their own data
Field Name For User ID	<code>owner_id</code>


Stitch creates a new rule (a combination of a role and permissions) based on the template. This rule allows an authenticated user to only read and write their own documents and to insert new documents.

d. Click **Add Collection**.

MONGODB AUTHORIZATION:

MongoDB Stitch rules do not override the read and write access (i.e. authorization) that may have been set up separately in MongoDB. That is, MongoDB Stitch rules determine whether the fields are readable or writable; not whether the client has authorization to read or write to a particular database or collection.

Similarly, MongoDB Stitch validation rules do not override document validation rules set up separately in MongoDB.

To view the rule associated with the template, expand **todo**, and then click on **items**. Click on the edit button () to open the role editor.

By default, the collection has the following **Apply When** rule:

```
{
  "owner_id": "%user.id"
}
```

With this rule, read and write operations can access all fields in a document if the document contains an `owner_id` field whose value equals the user ID (`%user.id`) of the client issuing the read. If the `owner_id` field in the document does not equal the client user id, the document is not readable.

2

Add a Schema for **todo.items**.

IMPORTANT:

This step is only necessary if you're using GraphQL to query your data. GraphQL requires a database schema to structure queries and mutations. Creating a schema will not impact your MongoDB queries if you are following the Web, Android, or iOS tutorials.

A schema is a JSON document that defines the shape and contents of your collection. Stitch allows you to generate a schema based on the data in your collection or create one manually. Since we don't have any data in our `todo` collection yet, we'll choose the second option.

To add a new schema, click on the **Schema** tab. Paste the following schema into the **Stitch Schema** text box:

```
{
  "title": "item",
  "properties": {
    "_id": {
      "bsonType": "objectId"
    },
    "owner_id": {
      "bsonType": "string"
    },
    "task": {
      "bsonType": "string"
    },
    "checked": {
      "bsonType": "boolean"
    }
  }
}
```

The schema defines the shape of the data in the `todo` collection. Each `item` in the collection has four fields: `_id`, `owner_id`, `task`, and `checked`. We also define the types for each of these fields.

After you paste the schema into the text box, click **Save** on the top right corner.

HOW STITCH USES YOUR SCHEMA:

Although MongoDB collections do not require a schema to be specified, Stitch can use schemas to enforce the shape of the data in your collection. When a schema is enforced, users will receive an error if they attempt to insert or update documents that do not match the fields and types in the schema.

Stitch requires a schema be defined when you use GraphQL in your application. The GraphQL query language uses your schema to create query and mutation functions.

You can view the GraphQL operations generated from your schema by clicking **GraphQL** under **Data Access & Security** in the left-hand navigation bar. The **Documentation Explorer** on the right-hand side shows all the GraphQL operations you can execute based on your collection schema. Click **Query** under **Root Types** to view the generated queries:

```
item(query: ItemQueryInput): Item
items(
  query: ItemQueryInput
  sortBy: ItemSortByInput
  limit: Int
): [Item]!
```

The `item` function takes in an `ItemQueryInput` and returns an `Item`; this is similar to the `findOne` method that finds one document. `items` works similarly, but returns an `Item` array. GraphQL generates similar functions for inserting, updating, and deleting documents in your collection called mutations.

C. Set Up Authentication

Estimated Time to Complete: ~1 minute

To get started, we'll use only anonymous authentication. This allows a user to load your app and try it out, but as soon as they are done using the app, they will no longer be able to access their to-do list. We will add additional authentication providers – and the ability to link accounts – later.

To enable Anonymous Authentication:

- In the left-hand navigation, under **Data Access & Security**, select **Users**.
- Select the **Providers** tab.
- In the authentication provider list, click the row that is labeled **Allow users to log on anonymously**.
- Click the **Disabled** slider so that it turns green and changes to **Enabled**.
- Click **Save**.

D. Deploy Your Application

Estimated Time to Complete: ~1 minute

Stitch saves changes that you make in a draft state that is not immediately available to client applications. To give client applications access, you must deploy your draft changes. To deploy changes, click **Review & Deploy Changes** in the banner at the top of the Stitch UI and then click **Deploy**.

Summary

Congratulations! You have set up and deployed the Stitch backend app and are now ready to build a client application.

What's Next

Now it's time to build one or more client To-do applications. Follow this tutorial to build the client of your choice.
