Admin Access

Mobile Backend Services (MBS) admin access allows application admin users to execute some batch operations and make MBS API calls on behalf of another user.

Create an admin user

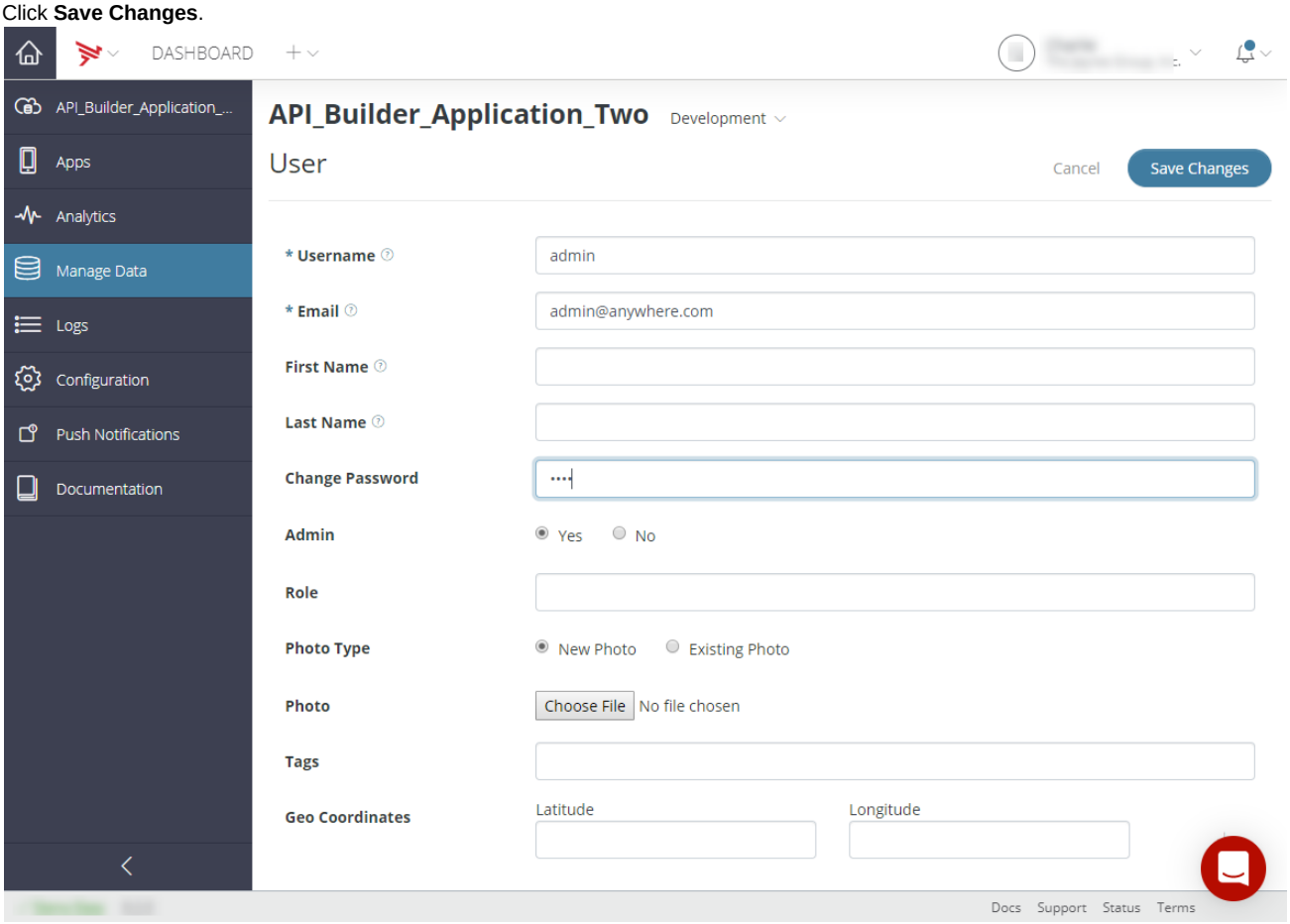
Before creating an admin user, log in to the Dashboard and select your application.

1. Log into the AMPLIFY Platform.
2. Select **Dashboard** on the Dashboard tile.
3. Select an application from the **Projects** tab.

Then, either create a new admin user or add admin access to an existing user.

Create a new admin user

1. In the left navigation bar, click **Manage Data**.
2. In the main pane, click **Users**.
3. Click **+ Create User**. A dialog appears.
4. Under the **Admin** section, click the **Yes** radio button.
5. At a minimum, enter a username, email, and password.
6. Click **Save Changes**.



Mobile Backend Services creates a new user with admin access.

Add admin access to an existing user

1. In the left navigation bar, click **Manage Data**.
2. In the main pane, click **Users**.
3. Locate the user you want to give admin access to and click the username to edit the user.
4. Locate the **Admin** section and click the **Yes** radio button.
5. Scroll down and click **Save Changes**.

This user now has admin access. To disable access, follow the same steps except click the **No** radio button.

Perform Mobile Backend Services API calls on behalf of another user

An admin user can perform MBS API calls on behalf of another user. For example, when you specify the `su_id` parameter to an ID of another user as part of the create method, the admin user creates an object on behalf of that user. The `user` parameter for the object will be reported as the other user, not the admin user.

This admin operation is supported by any create, update and delete method, as well as the following methods:

- KeyValues.append
- KeyValues.incrby
- KeyValues.set
- PushNotifications.subscribe

For example, the following curl command creates a new status for the specified user:

Expand source

```
curl -b cookies.txt -c cookies.txt -F "su_id=520289441e1ef70b1a0236d2" -F
"message=Hola, Mundo\!"
"api.cloud.appcelerator.com/v1/statuses/create.json?key=APP_API_KEY"
{
  "meta": {
    "code": 200,
    "status": "ok",
    "method_name": "createStatus"
  },
  "response": {
    "statuses": [
      {
        "id": "5202c1ed87173a0afc024524",
        "message": "Hola, Mundo\\!",
        "created_at": "2013-08-07T21:53:49+0000",
        "updated_at": "2013-08-07T21:53:49+0000",
        "user": {
          "id": "520289441e1ef70b1a0236d2",
          "created_at": "2013-08-07T17:52:04+0000",
          "updated_at": "2013-08-07T17:52:04+0000",
          "external_accounts": [

          ],
          "confirmed_at": "2013-08-07T17:52:04+0000",
          "username": "not_an_admin",
          "admin": "false"
        }
      }
    ]
  }
}
```

To verify that the specified user created this status and not the admin user, run the following curl command and compare the user IDs:

Expand source

```
curl -b cookies.txt -c cookies.txt
"https://api.cloud.appcelerator.com/v1/statuses/show.json?key=APP_API_KEY&status_id=52
02c1ed87173a0afc024524"
{
  "meta": {
    "code": 200,
    "status": "ok",
    "method_name": "showStatus"
  },
  "response": {
    "statuses": [
      {
        "id": "5202c1ed87173a0afc024524",
        "message": "Hola, Mundo\\!",
        "created_at": "2013-08-07T21:53:49+0000",
        "updated_at": "2013-08-07T21:53:49+0000",
        "user": {
          "id": "520289441e1ef70b1a0236d2",
          "created_at": "2013-08-07T17:52:04+0000",
          "updated_at": "2013-08-07T17:52:04+0000",
          "external_accounts": [

          ],
          "confirmed_at": "2013-08-07T17:52:04+0000",
          "username": "not_an_admin",
          "admin": "false",
          "stats": {
            "photos": {
              "total_count": 0
            },
            "storage": {
              "used": 0
            }
          }
        }
      }
    ]
  }
}
```

Batch delete

Mobile Backend Services provides an API endpoint named `batch_delete` that allows application admins to delete multiple MBS objects in one operation. The method takes a `where` parameter that constrains the selection of objects to delete. If the `where` parameter is omitted, all objects are deleted. For performance reasons, the number of objects that can be deleted in a single batch delete operation is limited to 100,000. Objects are deleted asynchronously in a separate process, not immediately upon method invocation.

If the `where` parameter is omitted, all objects are deleted. For example, if the `where` parameter is omitted from a batch delete of User objects, all users will be deleted.

Certain MBS objects can have dependencies on other objects. For example, when you create a Checkins object you can specify a Places or Events object to associate with it. In this case, the Checkins object is a dependency of the Places or Events object. If you delete the Places or Events object, the dependent Checkins object is **not** deleted.

For example, the following deletes all Users objects whose `favorite_color` custom field is 'blue'.

```
$curl -b cookies.txt -c cookies.txt -X DELETE -F "where={\"favorite_color\":\"blue\"}"

 api.cloud.appcelerator.com/v1/users/batch_delete.json?key<API_KEY>&pretty_json=true

 {
  "meta": {
    "status": "ok",
    "code": 200,
    "method_name": "adminBatchDelete"
  }
}
```

Note that the method returns an HTTP 200 code (success) even if the query matched no objects.

The following MBS objects support batch delete operations:

- Checkins
- PhotoCollections
- Events
- Files
- CustomObjects
- Photos
- Places
- Posts
- Reviews
- Statuses
- Users

Admin drop custom collection

An application admin user can also drop a Custom Object collection using the `admin_drop_collection` method. When calling the `admin_drop_collection` method, the admin user must specify a class name to indicate which custom collection to drop.

For example, the following drops the `car` collection:

```
$ curl -b c.txt -c c.txt -X DELETE
"api.cloud.appcelerator.com/v1/objects/car/admin_drop_collection.json?key=hPkMYgNozXR8
xegNvWjqBVTcWK8P5fIX"
{
  "meta": {
    "status": "ok",
    "code": 200,
    "method_name": "dropCollection"
  }
}
```

Only Custom Objects support the drop custom collection method.