Exercise: Remote Databases

1. Create "Books" REST Service

Create a collection called **books** where each book have **`title, author, isbn`**.

The following REST services will be created automatically to access your data:

- **List All Books**
  - Endpoint: **`https://baas.kinvey.com/appdata/[:appId]/books`**
  - Method: **`GET`**
  - Headers:
    - Basic Authorization with **user credentials**
  - Returns (JSON)

- **Create a New Book**
  - Endpoint**`: https://baas.kinvey.com/appdata/[:appId]/books`**
  - Method: **`POST`**
  - Headers:
    - Basic Authorization with **user credentials**
    - **`Content-type: application/json`**
  - Request body (JSON): **`{"title":"…", "author":"…", "isbn":"…"}`**

- **Update a Book**
  - Endpoint: **`https://baas.kinvey.com/appdata/[:appId]/books/[:bookId]`**
  - Method: **`PUT`**
  - Headers:
    - Basic Authorization with **user credentials**
    - **`Content-type: application/json`**
  - Request body (JSON): **`{"title":"…", "author":"…", "isbn":"…"}`**

- **Delete a Book**
    - Endpoint: **`https://baas.kinvey.com/appdata/[:appId]/books/[:bookId]`**
    - Method: **`DELETE`**
    - Headers:
        - Basic Authorization with **user credentials**
        - **`Content-type: application/json`**

Test your REST Service, e.g. using **`Postman.`** Try to list all books in **`JSON format`** with an **`HTTP GET request`** to the **`REST API`**.

List All Books

Use the skeleton provided in the resources folder. Add an **`AJAX`** call that takes all books from your application as **`JSON`** object and **`displays`** them when the page loads.

Create a Book

Add an **`HTML form`** with **`[Submit] button`** for adding a new book. When the button is **pressed**, create a **new book** using its **`REST API`** with an **`AJAX request`**.

Edit a Book

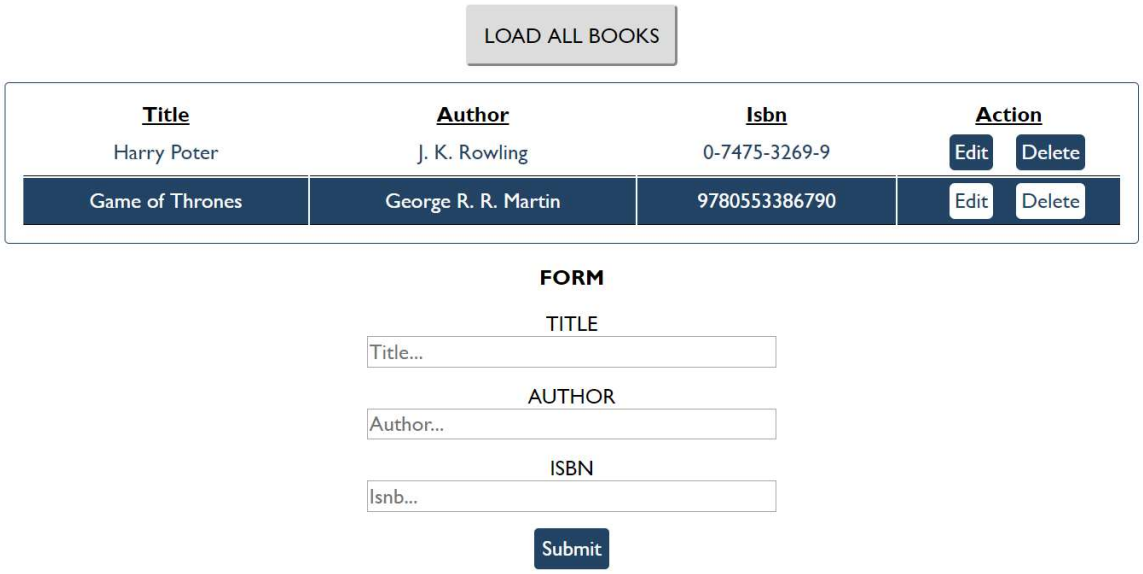
Implement "**`Edit a Book`**" functionality. Clicking on a book should **load its data** in an **`HTML form`**. By **clicking** the **`[Edit]`** button, the **book data** should be **updated** at the **server side** with an **`AJAX request`**.

Delete a Book

Implement "**`Delete a Book`**" functionality. Each book should have a **`[Delete]`** button. Clicking on it should **delete the book at the server side** with an **`AJAX request`**.

* Add Tags for Each Book

Implement **`tags`** for the books. Tags should be stored at kinvey in the Book collection in a column "**`tags`**" as **`array of strings`**. List the tags for each book. Implement **`add`** / **`edit`** / **`delete`** for tags when a book is **`created`** / **`updated`**.



2. Students

Your task is to create functionality for creating and listing students from a database in Kinvey. Create a new collection called "**students**",

Each student has:

- **`ID`** - **`number`**, non-empty
- **`FirstName`** - **`string`**, non-empty
- **`LastName`** - **`string`**, non-empty
- **`FacultyNumber`** - **`string of numbers`**, non-empty
- **`Grade`** - **`number`**, non-empty

You need to write functionality for creating students. When creating a new student, make sure you name the properties accordingly. Create at least one student to test your code.

You will also need to extract students. You will be given an **`HTML template`** with a table in it. Create an **`AJAX request`** that extracts all the students. Upon fetching all students from the database, add them to the table each on a new row, **sorted** in **`ascending order`** by **`ID`**.

Use the skeleton from the provided resources.

Screenshots

ID	First Name	Last Name	Faculty Number	Grade
1	Isaac	Netero	900005878123	4.99
2	George	Serie	900004560603	5.23
3	Nvy	Ose	900001234567	6.00
4	Sunny	Jackson	900003342331	4.40
5	Aina	Haward	900001110011	5.56

ID	First Name	Last Name	Faculty Number	Grade
1	Isaac	Netero	900005878123	4.99
2	George	Serie	900004560603	5.23
3	Nvy	Ose	900001234567	6.00
4	Sunny	Jackson	900003342331	4.40
5	Aina	Haward	900001110011	5.56