

Schema Validation

The screenshot shows the MongoDB Intro Lab page for the 'Enable Validation for the Users Collection' exercise. The page is titled 'Enable Validation for the Users Collection' and includes a table of contents on the left and a right-hand sidebar with links to 'Database user permissions', 'Explore the JSON schema', 'Explore the script to apply the schema', 'Apply the schema to the users collection', 'Test the schema validation', and 'Summary'. The main content area explains that the exercise involves exploring a pre-written JSON validation schema for the 'users' collection, running a script to apply it, and testing the schema validation by inserting a document that does not match the schema. Below this, the 'Database user permissions' section provides instructions on how to update the validator for any database collection, requiring admin privileges. The instructions are numbered 1 through 5, detailing the steps from opening the Atlas UI to scrolling down to the 'Database User Privileges' section and expanding the 'Built-in Role' dropdown.

Enable Validation for the Users Collection

In this exercise, you will explore the pre-written JSON validation schema for the `users` collection, run a script to apply it to the collection, and test the schema validation by inserting a document that does not match the schema.

Database user permissions

To update the validator for any database collection, your database user must have admin privileges. Follow these steps to ensure your user has the correct permissions:

1. Open the [Atlas UI](#).
2. In the left-hand menu, navigate to **Network Settings** and select "Database Access."
3. Locate your database user in the list. Check the **MongoDB Roles** column for the role of the user you are using for this workshop. If the role is not `atlasAdmin@admin`, you will need to update it.
4. If the role is not `atlasAdmin@admin`, click "Edit" button next to the user.
5. Scroll down to the **Database User Privileges** section and expand the **Built-in Role** dropdown.

The screenshot shows a VS Code editor with a TypeScript script for applying schema validation to the MongoDB Atlas database. The script is located in the `src` directory of the `library-management-system` project. The script imports the `load-env-vars.js` file and the `connectToDatabase` function from the `database.js` file. It then connects to the MongoDB Atlas database and applies the schema validation to the `users` collection. The script defines a `userSchema` object with the following properties:

- `name`: A string, required, with a minimum length of 5. Description: 'must be a string and is required'.
- `isAdmin`: A boolean, required. Description: 'must be a boolean and is required'.

The script also defines a `results` array and a `resultUsers` object. The script is executed using the `node` command.

```
server > src > schema-validation > TS apply-schema.ts > ...
1 import './load-env-vars.js';
2 import { connectToDatabase, databases } from './database.js';
3
4 const { DATABASE_URI } = process.env;
5
6 console.log('Connecting to MongoDB Atlas...');
7 await connectToDatabase(DATABASE_URI);
8 const db = databases.library;
9 console.log('Connected!\n');
10
11 const results = [];
12
13 const userSchema = {
14   bsonType: 'object',
15   required: ['name', 'isAdmin'],
16   properties: {
17     name: {
18       bsonType: 'string',
19       minLength: 5,
20       description: 'must be a string and is required'
21     },
22     isAdmin: {
23       bsonType: 'bool',
24       description: 'must be a boolean and is required'
25     }
26   }
27 };
28
29 console.log('Applying schema validation for users...');
30 const resultUsers = await db.command({
  ...
});
```