Middle East Technical University

Department of Computer Engineering

CENG 495

Cloud Computing
Spring 2021–2022
HW - 2

Due date: 2022-05-18 23:59

1 Introduction

In this homework, you will use NoSQL database MongoDB[1] on the Database-as-a-Service (DBaaS) cloud platform MongoDB Atlas with serverless application platform MongoDB Realm to develop and deploy an application.

The application will be a book manager that lets users keep track of their books, rate and review them.

2 MongoDB Platform, Atlas and Realm - Start Here

- Go over NoSQL and MongoDB concepts. Make sure you understand how dynamic schema differs from what you are used to from DBMS.
- Create a free account on MongoDB Atlas https://www.mongodb.com/atlas/database
- Create a new project and build a new database
- Use MongoDB Realm to prepare the backend of your application and deploy your HTML & JavaScript code
- Follow a tutorial! We recommend https://www.mongodb.com/developer/quickstart/realm-web-sdk/, but there are plenty available for MongoDB, developing applications in MongoDB cloud platform etc.

3 Book Tracker

When it comes to book aggregation, Goodreads is the most popular option. However, the site looks (and have always looked) old and clunky. This is your chance to make something better!

Your application will be implemented in HTML and JavaScript.[2] It should communicate with your MongoDB database that you had deployed on Atlas using Realm Web SDK. Let's talk about the required pages and collections to have for your application.

[1] https://www.mongodb.com/

[2] Yes, you can use *that* framework as long as it deploys to Realm, no need to mail me

3.1 Home Page

This is the start page of your application, it is accessed through `index.html`. The required actions for the home page are:

Add Book Create & insert a new book into the database, read below for the required attributes of books.

Remove Book Remove the book from the application, deleting ratings and reviews of the book with it. Any user effected by this removal should have their relevant fields updated as well.

Add User Create a new user, refer to the user attributes (3.2.1) below.

Remove User Remove the user and update relevant books that are effected by this change.

Login as a User Akin to Linux `su` command, start using the application as the selected user. You are not required to implement user authorization. There is an Anonymous Users functionality in MongoDB Realm for this exact purpose. This action will be different if you are going for the authorization bonus (3.1.1).

~~3.1.1 User Authorization Bonus~~

~~You can get this bonus by implementing user authentication and authorization. Make sure to follow a *best practices guide* to learn how to implement this (e.g. handling cookies).~~

3.2 User Page

This is the *profile* page of the user. Integrate & display every user attribute given below on this page, in a manner that makes sense with usability in mind. Refer below (3.2.4) for the actions offered to the users.

3.2.1 User Attributes

Username The username of the user, required field

Number of Books Read The number of books read by the user. This is initialized as zero. A book is read when the user rates it

Favorite Books A list of user's favorite books. Initialized as zero

Average Rating The average of every rating given by this user. Initialized as zero

Reviews Every review written by this user. The order to display the reviews are explained below (3.2.2)

If you feel like you need additional attributes, you are welcome to implement them.

Allow authors to sign up to your application as users as well.[3] Authors cannot rate or review books though they can still display their favorite books. Instead, authors can display the books they have authored on their profile pages.

3.2.2 Review Sorting

The reviews of the user should be sorted as:

- The user's favorite books on top, in the same order as user's favorite books list
- The rest of the books reviewed by the user, sorted by the given rating. The highest rated books should be on top.

[3] Hint: you can use a dynamic schema here and on other places too!

3.2.3 Sorting Bonus

There is a bonus if you implement *date and timestamps* and display the most recently read book on top.

3.2.4 User Actions

Users should be able to;

Rate Book Rates a book on a scale of 1 to 5. The average rating of the book and the average rating given by the user should be updated. A user can rate a book multiple times, the latter overwriting the prior rating. The first time user rates a book also marks the book as *read* by the user, updating relevant fields.

Review Book Allows a user to write a text review on the book. The user must have rated the book before or during the review process to add the review. This review should appear on the book's reviews and the user's reviews. Reviews can be updated, and the newer review overwrites the older one.

Favorite Book Places this book on user's favorite books list. You can use a first-in-first-out queue here and kick the oldest favorite book when a new one appears and the queue is full. You can pick any reasonable queue size.

3.3 Books Page

Every book in the system is shown here. How to display them and in what order is left up to you.

3.3.1 Pagination Bonus

You can employ *pagination* and display less than ten books per page with an option to navigate between the pages.

3.3.2 Book Attributes

The following is the list of attributes of books. Note that not every field is always present, for every book.

~~**Name** The name of the book~~

~~**Author** The author of the book. Books can have multiple authors~~

~~**Translator** If the book is a translation then this field is for the translator. It's not used for books in their original language.~~

~~**Editor** The editor of the book, if any. Sometimes for books with multiple authors an editor is required to compile the whole book together, this field is useful for those cases.~~

~~**Cover** The cover image of the book. You do not need to implement image upload, just hyperlink to an image file on the Internet and display it here~~

~~**Fiction or Non-fiction** Whether the book is fiction or non-fiction. This is important for the genre~~

~~**Publisher** The publisher of the book~~

~~**Genre** The genre of the book. The presence of genres are different for fiction and non-fiction books. See below (3.3.3) for more info.~~

Year Published The publishing year (in YYYY format) of this book

Rating Average rating this book has got by all the users

All Reviews List of reviews on the book written by all users

You are free to add additional attributes as you see fit. For instance, to keep track of the average rating, you can also have a **number of readers**.

3.3.3 Handling Genres

If the book is fiction; identify 6 of your preferred genres and ensure that application's genres are limited only to them. Support this enforcement by drop-down menus or autocompletion in your application. On the other hand, non-fiction books do not have genres (in the scope of this application).

4 Before Submission

Populate your application by at least 12 books and 6 users. Have your users rate and review every book between them so that every book has at least 3 ratings and 2 reviews.

Run MongoDB's `findOne( )` function to show the following documents;

- a fiction book
- a non-fiction book with multiple authors
- a translated book
- a book with an editor and multiple authors
- a regular user
- an author

Pretty-print & add these to your `README` file.

5 Submission

- Deploy your application to MongoDB cloud platform and submit the source code of your application to our ODTUClass page. Archive your project as a `.tar.gz` file and name it as "firstname_lastname.tar.gz".
- Your submission must include a `README` file. Fill it with your design decisions, required MongoDB documents (mentioned in §4), the URL of your application and specify which bonuses should be considered in your submission.
- Make sure that your deployed application is accessible to anyone on the Internet.
- This is an individual assignment. You can discuss your ideas with your peers but using implementation specific code that is not your own is strictly forbidden and constitutes as cheating. This includes but not limited to friends, any previous homework, CENG homework repositories on GitHub, or the Internet in general. The violators will get no grade from this assignment and will be punished according to the department regulations.

6 Grading

80% of your grade will be on the usability of your web applications and the correctness of your MongoDB usage. Your `README` file will make up the 20% of your grade. Any bonus you get will be +7 to your final grade, with a maximum of 100, *after* late submission deductions. Black-box testing will not (cannot) be employed for this homework, so I will use each application and read every submission in its entirety for grading.