User Authentication | Coursera

Important Information
It is especially important to submit this assignment before the deadline, April 3, 11:59 PM PDT, because it must be graded by others. If you submit late, there may not be enough classmates around to review your work. This makes it difficult - and in some cases, impossible - to produce a grade. Submit on time to avoid these risks.

Instructions

Assignment Overview
In this assignment you will continue the exploration of user authentication. We have already set up the REST API server to validate an ordinary user. Now, you will extend this to verify an Admin and grant appropriate privileges to an Admin. At the end of this assignment, you would have completed the following:

- Check if a verified ordinary user also has admin privileges.
- Allow an ordinary user to only perform GET operatioins
- Allow only an Admin to perform POST, PUT and DELETE operations
- Allow an Admin to be able to GET all the registered users' information from the database

Assignment RequirementsThis assignment is divided into three tasks as detailed below:Task 1In this task you will implement a new function named *verifyAdmin()* in *verify.js* file. This function will check an ordinary user to see if s/he has Admin privileges. In order to perform this check, note that all users have an additional field stored in their records named *admin*, that is a boolean flag, set to *false* by default. Furthermore, when the user's token is checked in *verifyOrdinaryUser()* function, it will load a new property named *decoded* to the *request* object. This will be available to you if the *verifyAdmin()* follows *verifyOrdinaryUser()* in the middleware order in Express. From this *req* object, you can obtain the admin flag of the user's information by using the following expression:

```
    req.decoded._doc.admin
```

You can use this to decide if the user is an administrator. The *verifyAdmin()* function will return *Next();* if the user is an admin, otherwise it will return Next(err);

Task2
In this task you will update all the routes in the REST API to ensure that ordinary users are restricted only to perform GET operations. On the other hand, Admins can perform POST, PUT and DELETE operations. Update the code for all the routers to support this.

Task 3

In this task you will now activate the /users REST API end point. When an Admin sends a GET request to http://localhost:3000/users you will return the details of all the users. If an ordinary user performs this operation, you should return a reply with the status of 403, and a message "You are not authorized to perform this operation!".

Review criterialess
Your assignment will be graded based on the following criteria:

Task 1

- You have implemented the *verifyAdmin()* function in *verify.js*.
- The *verifyAdmin()* function will allow you to proceed forward along the normal path of middleware execution if you are an Admin
- The *verifyAdmin()* function will prevent you from proceeding further if you do not have Admin privileges, and will send an error message to you in the reply.

Task 2

- An ordinary user is restricted only to perform the GET operation on the resources/REST API end points.
- An Admin (who must be first checked to make sure is an ordinary user), can perform the GET, PUT, POST and DELETE operations on any of the resources/ REST API end points.

Task 3

- A GET operation on http://localhost:3000/users by an Admin will return the details of the registered users
- An ordinary user (without Admin privileges) cannot perform the GET operation on http://localhost:3000/users.