About the final assessment

In this course, you will be assessed in two ways: you will complete the peer-reviewed Little Lemon restaurant API project and a final graded quiz.

The purpose of the assessments

The primary purpose of the assessments is to check your knowledge and understanding of the key learning objectives of the course.

Most importantly, they help you establish which topics you have mastered and which topics require further focus before completing the course. Ultimately, the graded assessments are designed to help you make sure that you can apply what you have learned.

What to expect from the graded quiz

The quiz tests only the topics covered throughout the course so there shouldn’t be any surprises. Please review the feedback on your answers and where necessary go back and work through the topics that you feel require your further attention.

What to expect from the Little Lemon API project

You have encountered exercises, knowledge checks, in-video questions and other assessments throughout this course. Nothing in the API project will be outside of the scope of what you have already learned, so you should be well-placed to succeed.

You will combine the skills and knowledge that you applied throughout the course and use them to build an API for the Little Lemon restaurant. Little Lemon's management wants to develop an online-based order management system and mobile application. You will need to create the back-end API that allows customers to browse food items, view the item of the day and place orders. Managers need to be able to update the item of the day and monitor orders and assign deliveries. And the delivery crew should be able to check the orders assigned to them and update an order once it’s delivered.

How will I get feedback on my API project?

This is a peer-review project which means that your API will be evaluated by your fellow learners. Of course, this means that you will also need to give feedback to your peers and grade their APIs. In this way, you can learn from other learners' ideas and the issues that they may have encountered.

The rest of this lesson will guide you on how best to approach the development of your API project.

Good luck!

Congratulations on making it this far. You've learned all the basics of REST API development, the good, the bad, the do's and don'ts. Finally, you are here to use that knowledge to build an API project for the little lemon restaurant, this will be fun and a little challenging at the same time. Let's dive into a demonstration of what you're going to do. In this project you will deal with three types of users, managers, customers, and the delivery crew. You can use the knowledge of user roles you learned earlier in this course. You'll start with the user registration and authentication process that each user will use, then you will create API endpoints that can be used to assign users to a group like manager or a delivery person. This endpoint can be used to remove them from the groups too. You will need to think carefully about what type of token is required for this endpoint. Now, first is the manager role. Only managers can use some API endpoints to add, edit, and remove menu items. Managers should also be able to update any user to a delivery person. Next up is the customer rule, if a user doesn't belong to any specific group you should consider them a customer. Customers should be able to browse menu items, filter them by categories, and price ranges, and search menu items. You need to create APIs that will allow customers to add menu items to their cart and place an order. But here's the catch, the cart must be emptied when the order is successfully created. Does this sound a little bit tricky? Don't worry you can do it. You can also add an API endpoints to flush the cart at any point. One customer should only be able to have one cart at a time, and one cart should be able to contain multiple menu items. Last, there are APIs related to the delivery process. First, you need API endpoints for the managers to browse the orders and assign them to a delivery person. Managers should also be able to filter orders by their status, like delivered and not delivered. After successful authentication, delivery people should be able to browse orders assigned to them by using your API endpoints and mark them as delivered. Customers can always come to the orders endpoint to see their orders, including the status of that order and the total price. Finally, you will need to implement some throttling for the APIs you built, limit it to five API calls per minute. While working on this project, please ensure that your API endpoints support only the required HTTP methods, and always return the appropriate status code with the response. Let's discuss the tools you'll need. You should use VS Code for writing the code, you will need to use Django with Django REST framework or DRF, and everything should work in a virtual environment. For testing and debugging you can use insomnia, use only the packages and libraries you learned about earlier in this course. Just one last thing, if you use a session authentication class during the development, make sure to comment that out before submission. The finished project should only support token-based authentication. It's time to implement the knowledge you gained in this video. Good luck.

Project structure and API routes

Introduction

This reading is an overview of the scope of the project, all the necessary endpoints, and notes that you will have to implement in the final project. This reading will help you to successfully complete the project so read it carefully and reference it while developing your API project to help you keep on track.

Scope

You will create a fully functioning API project for the Little Lemon restaurant so that the client application developers can use the APIs to develop web and mobile applications. People with different roles will be able to browse, add and edit menu items, place orders, browse orders, assign delivery crew to orders and finally deliver the orders.

The next section will walk you through the required endpoints with an authorization level and other helpful notes. Your task is to create these endpoints by following the instructions.

Structure

You will create one single Django app called LittleLemonAPI and implement all API endpoints in it. Use pipenv to manage the dependencies in the virtual environment. Review the video about

Creating a Django Project using pipenv.

Function or class-based views

You can use function- or class-based views or both in this project. Follow the proper API naming convention throughout the project. Review the video about

Function- and class-based views

as well as the video about

Naming conventions

.

User groups

Create the following two user groups and then create some random users and assign them to these groups from the Django admin panel.

Manager

Delivery crew

Users not assigned to a group will be considered customers. Review the video about

User roles

.

Error check and proper status codes

You are required to display error messages with appropriate HTTP status codes for specific errors. These include when someone requests a non-existing item, makes unauthorized API requests, or sends invalid data in a POST, PUT or PATCH request. Here is a full list.

API endpoints

Here are all the required API routes for this project grouped into several categories.

User registration and token generation endpoints

You can use Djoser in your project to automatically create the following endpoints and functionalities for you.

When you include Djoser endpoints, Djoser will create other useful endpoints as discussed in the

Introduction to Djoser library for better authentication video.

Menu-items endpoints

User group management endpoints

Cart management endpoints

Order management endpoints

Additional step

Implement proper filtering, pagination and sorting capabilities for /api/menu-items and /api/orders endpoints. Review the videos about

Filtering and searching

and

Pagination

as well as the reading

More on filtering and pagination

.

Throttling

Finally, apply some throttling for the authenticated users and anonymous or unauthenticated users. Review the video

Setting up API throttling

and the reading

API throttling for class-based views

for guidance.

Conclusion

Now that you have a better idea of the scope of this project with the essential API endpoints, it’s time to start coding. Good luck!

Now that you know the scope of this project and you have read about database schema, it's time to create the models for this final project. Let's start, you need a few models to implement all the features in your project. Django already comes with the user model. So you don't need to worry about that. Now, open the model start by file and start with the category model. This model only recourse to field a slug and the title for the slug. You'll use the slug field type. The title will be char filled with a maximum length of 255. The client application will search against this title field so you must index it. Next, a menu item Model is required. A menu item can have a title and price which will be char field and decimal field respectively. It also needs a boolean field that indicates if this menu item is set as featured. And finally a menu item will always belong to a category. So you need a category field as a foreign key. Now, let's start with a curved model occurred Model is a temporary storage for users who can add menu items before placing an order. Remember in the project scope it was mentioned that a user can only have one card at a time. This model needs a user field which will be a foreign key to the user model. You need to import this model from the django dot contract dot dot dot models module. The card model also needs a menu item field which will be a foreign key to the men White model. And finally you need a quantity field and to price fields. For easier calculation. One is the unit price of the main item and the other is the price which multiplies the quantity with the unit price and saves the result when someone places the order, two things will happen. An order will be created that holds some important information about this order. So where will the menu items related to that order go from the curd. As soon as the order is placed, all the items will move from the curd to an order item table and link those main items with the ordinary. Let's create these two models. The order model needs a user field linking to the user stable as a foreign key and the bullying field called status to mark. If the order is delivered or not, the default value of this field will be set to zero. Then there is a total field continuing the price of all the menu items in this order and a dead field to mark. When this order was placed in this model, you need another field called a delivery crew which will link to the user model in Django. You cannot create two foreign keys referring to the same field in a foreign table. You must set a related name for it. So the related name of this field was said to delivery underscore crew because both user and the delivery crew were referring the user ID in the user table. And here comes the last model. The order item, all the items from the curd will be moved here with a link to the newly created order ID. And then those card items will be delighted. So after placing the order, the curd will be empty and the user can start adding new items for a new order you need a foreign key order referencing the order model in this model then comes the menu item. Again, another foreign key for the menu item model. You also need a quantity field because someone can add multiple items for a menu item and just like the card model. This model also requires a unit price and price field for easier calculation noted. The use of two meta classes in the card and order item model for two unique indexes In the court model. This line means that there can be only one menu item entry for a specific user. You can only change the quantity of the menu item And in the order item model this unique index means that one order can have only one entry for a specific menu item but the quantity can vary now that you have your models ready. It's time to start writing code

By working through the lessons in this course, you've learned the necessary skills and knowledge to develop back-end APIs for the Little Lemon project. You were provided with code snippets and your task was to use these, plus any of your own code, to complete an API project for the Little Lemon restaurant.

You will now take part in a peer-review exercise in which you will submit your completed APIs for two of your peers to review. You will also be required to review two of your peers' projects.

Detailed criteria are covered in the grading criteria overview below.

Review Criteria

When you submit your assignment, other learners in the course will review and grade your work. These are the criteria they’ll use to evaluate your APIs.

In this project, your APIs need to make it possible for your end-users to perform certain tasks and your reviewer will be looking for the following functionalities.

1. The admin can assign users to the manager group

2. You can access the manager group with an admin token

3. The admin can add menu items

4. The admin can add categories

5. Managers can log in

6. Managers can update the item of the day

7. Managers can assign users to the delivery crew

8. Managers can assign orders to the delivery crew

9. The delivery crew can access orders assigned to them

10. The delivery crew can update an order as delivered

11. Customers can register

12. Customers can log in using their username and password and get access tokens

13. Customers can browse all categories

14. Customers can browse all the menu items at once

15. Customers can browse menu items by category

16. Customers can paginate menu items

17. Customers can sort menu items by price

18. Customers can add menu items to the cart

19. Customers can access previously added items in the cart

20. Customers can place orders

21. Customers can browse their own orders

You'll also need to give feedback on and grade the assignments of two other learners using the same criteria.

Here are some examples to help you understand what your assignment should look like.

Browsing menu-items endpoint using the browser:

Making an HTTP GET call to menu-items endpoint:

Adding a user as a manager with admin token:

The HTTP body content to add a user to the manager group with an admin token:

You need to use pipenv to create the virtual environment and to manage the dependencies. Your APIs should be created using Django, DRF and Djoser. You need to use SQLite as the database for this project. You can develop your APIs using VS Code and test them using your browser or Insomnia.

Name the project directory LittleLemon and the app LittleLemonAPI. Ensure that you follow this naming convention to make the reviewing process easier.

You will be required to submit your APIs by uploading a zipped folder that contains your code.

Important note: Include the sqlite database file db.sqlite3 in your project submission and share the superuser’s username and password of all the other users you have created. Write these usernames and passwords in a notes.txt file and keep it inside the project directory and then zip the project.

To learn more about how to zip and unzip folders visit the

Mac

or

Windows

support page.

How to review you peers

Once you have submitted your file, you are required to review two peer submissions. You can view the peers that you need to review in the “Peers to review” section. You need to download their zipped project folder and unzip it. Then, prepare the virtual environment and install all dependencies using the following commands.

cd LittleLemon

pipenv shell

pipenv install

python manage.py makemigrations

python manage.py migrate

python manage.py runserver

Then log into the Django admin panel and create superuser and user groups and randomly assign them into these groups, the same way you did in your own project.