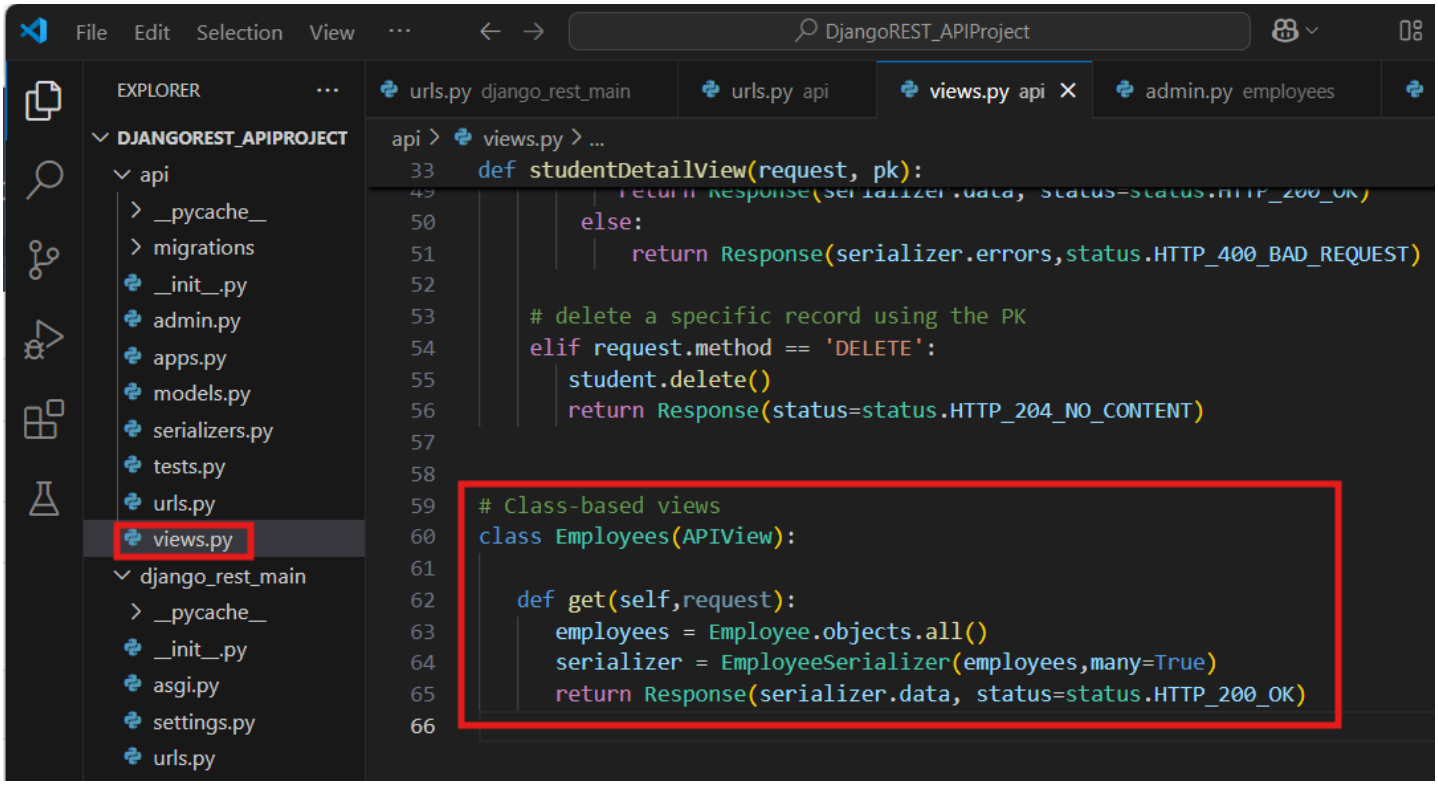
Topic: 8. Updating the Models using Class-Based Views

Speaker: | Notebook: API Development using Django Framework

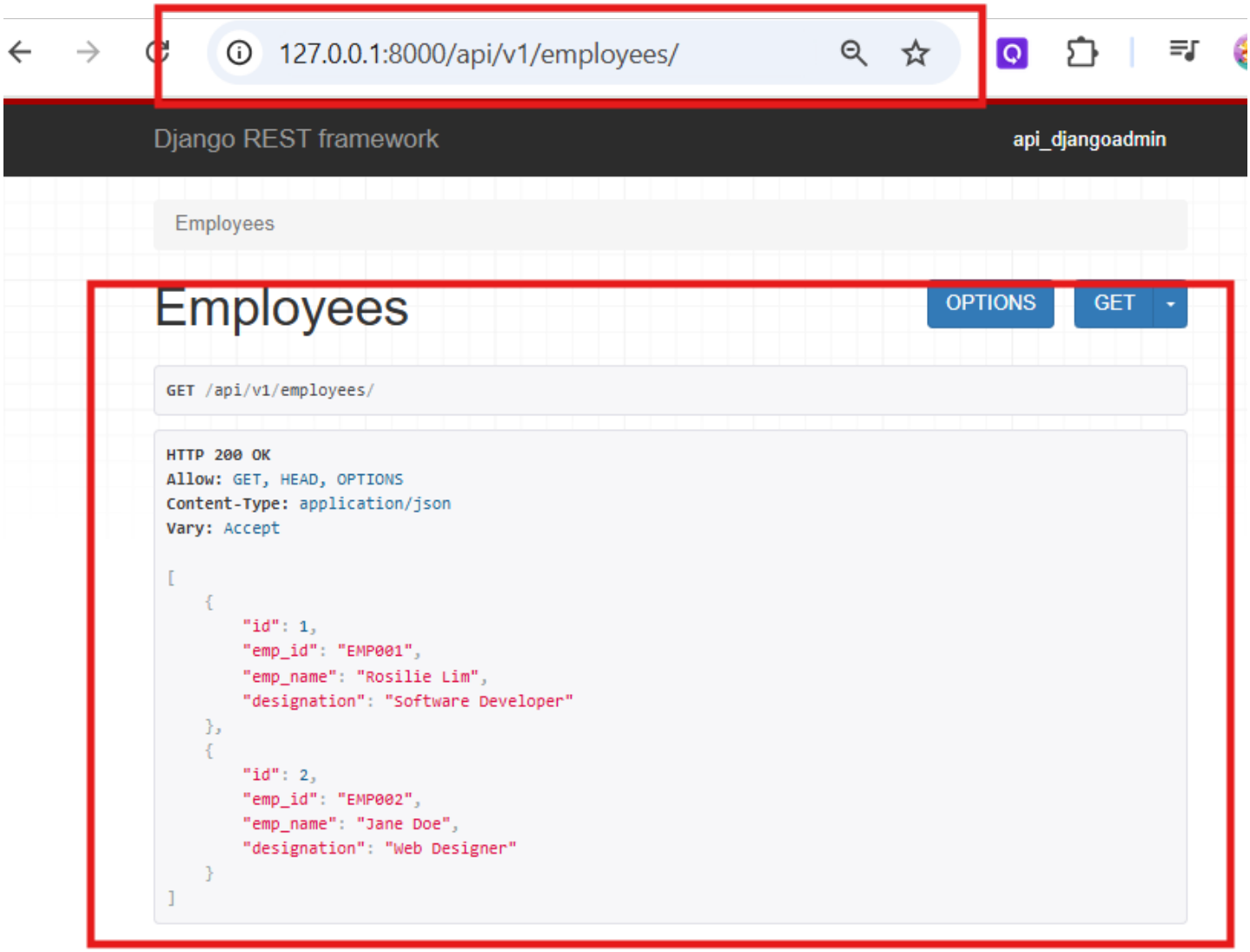


We previously created a new app called EMPLOYEES and a new Employee serializer for our Employee class.

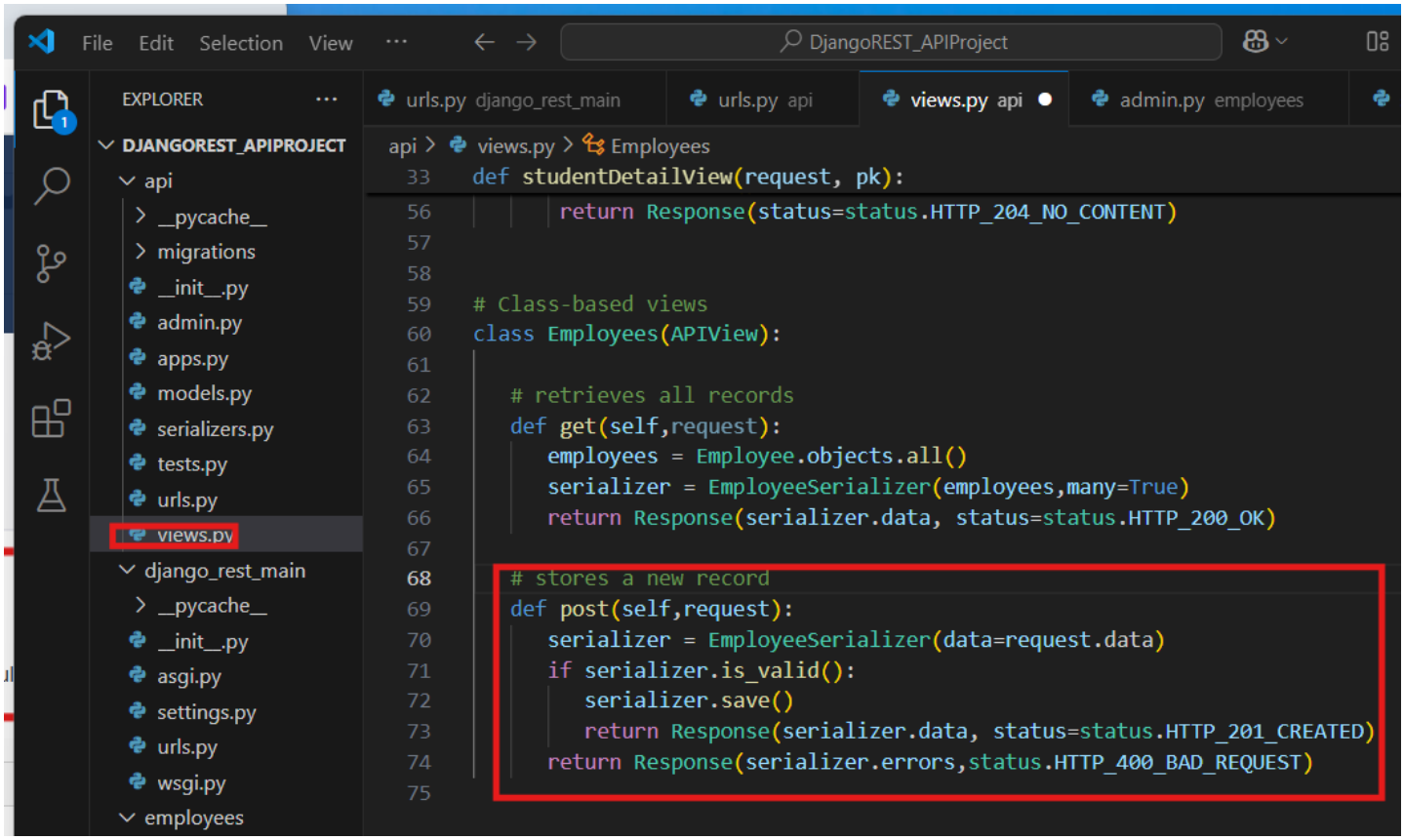
1. Previously, we updated the API\VIEWS.PY to create an EMPLOYEE class and its methods:



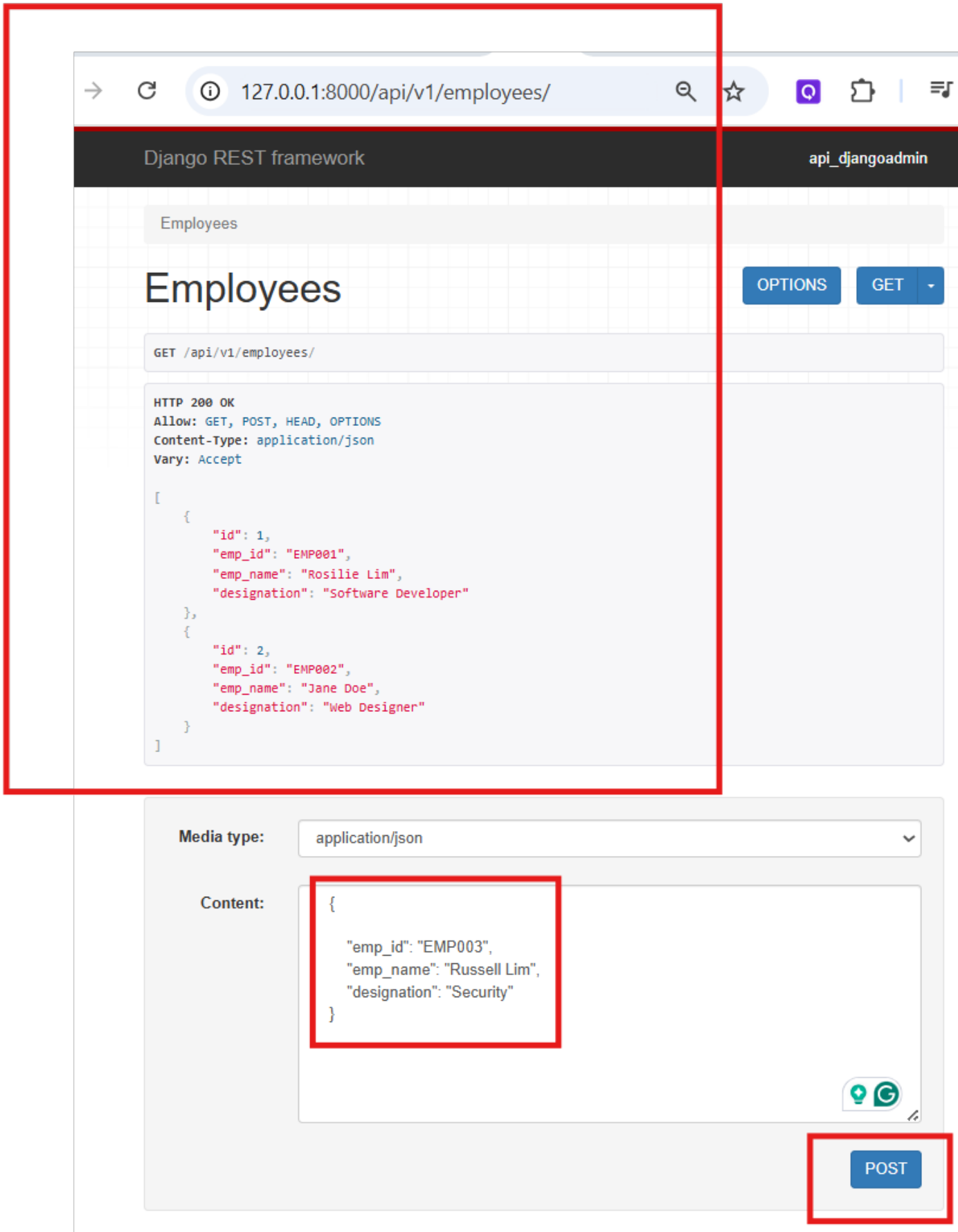
We run the new path:



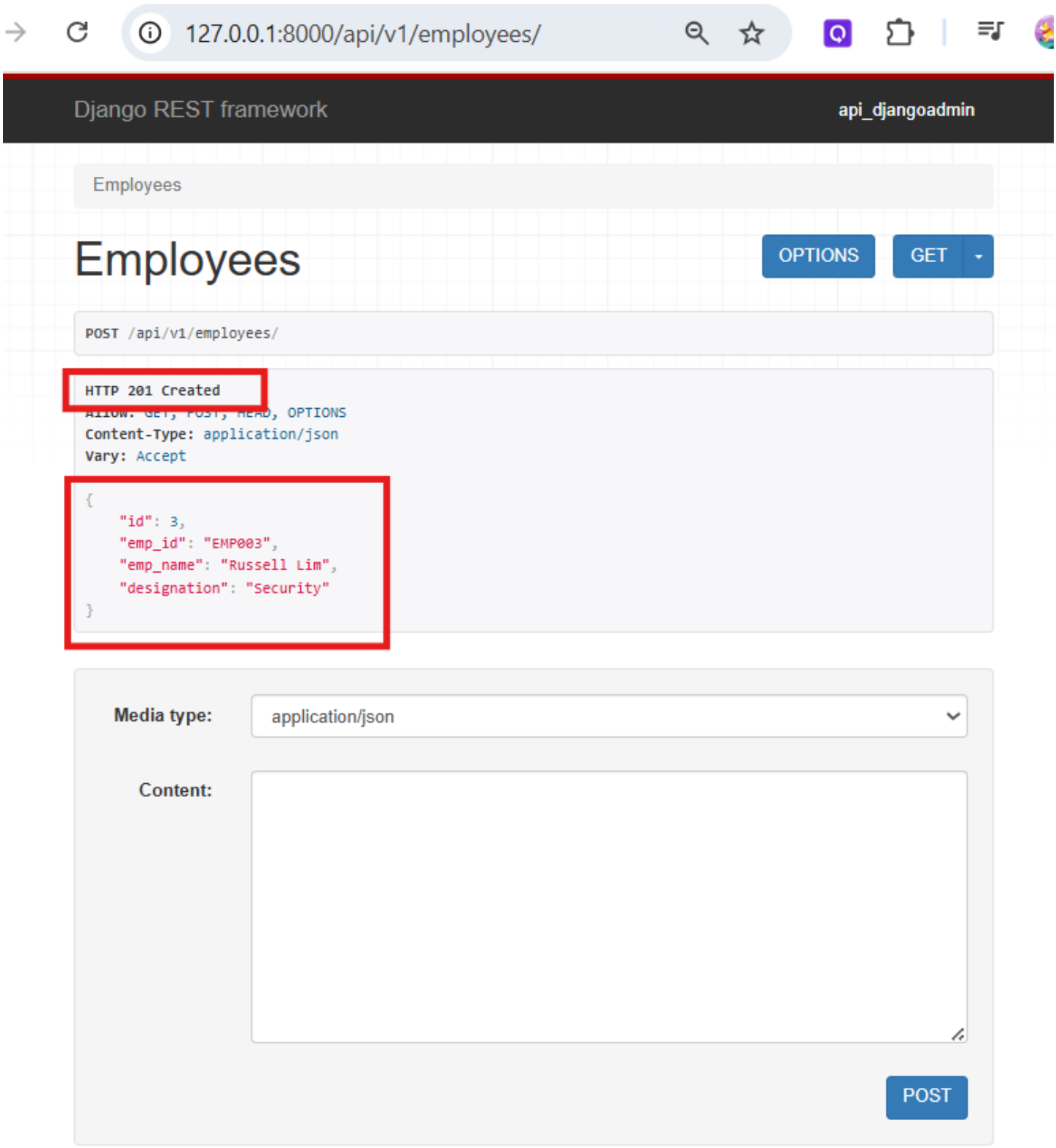
2. To post or add a new record to our model:



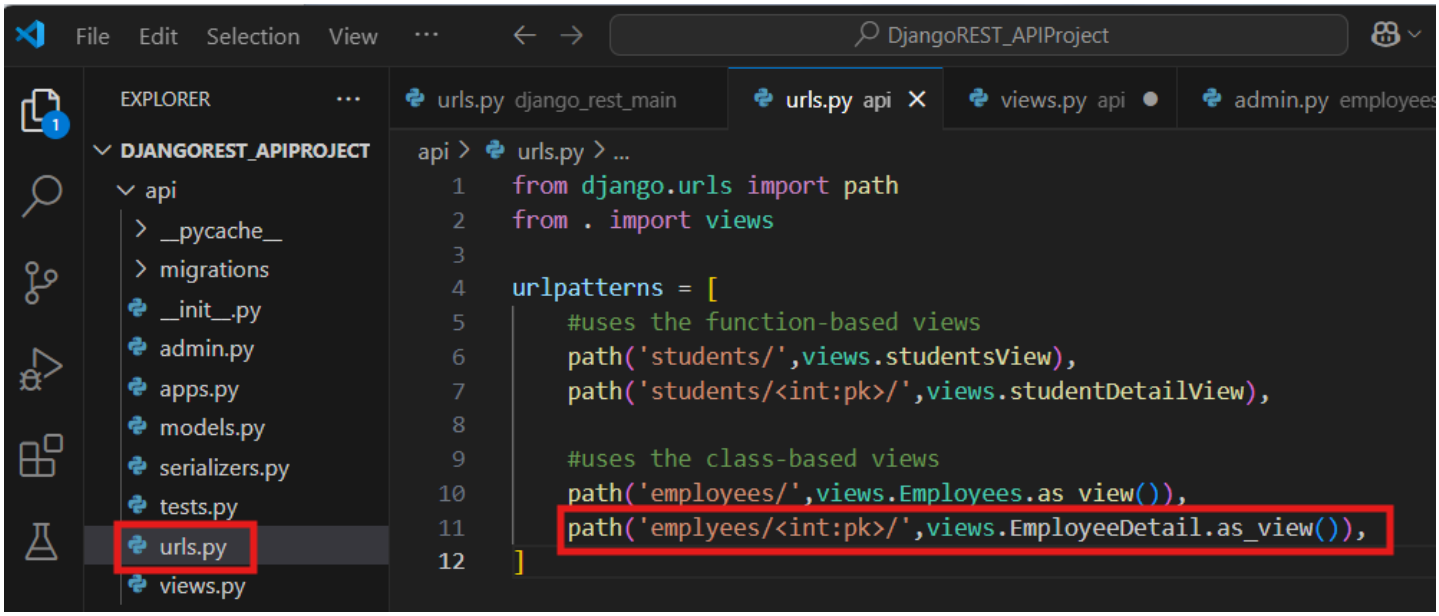
Add the new record. Follow the correct format to be able to save successfully.



This will result to:



3. To allow a single record operation like CRUD on a specific record, we update our API\URLS.PY as:

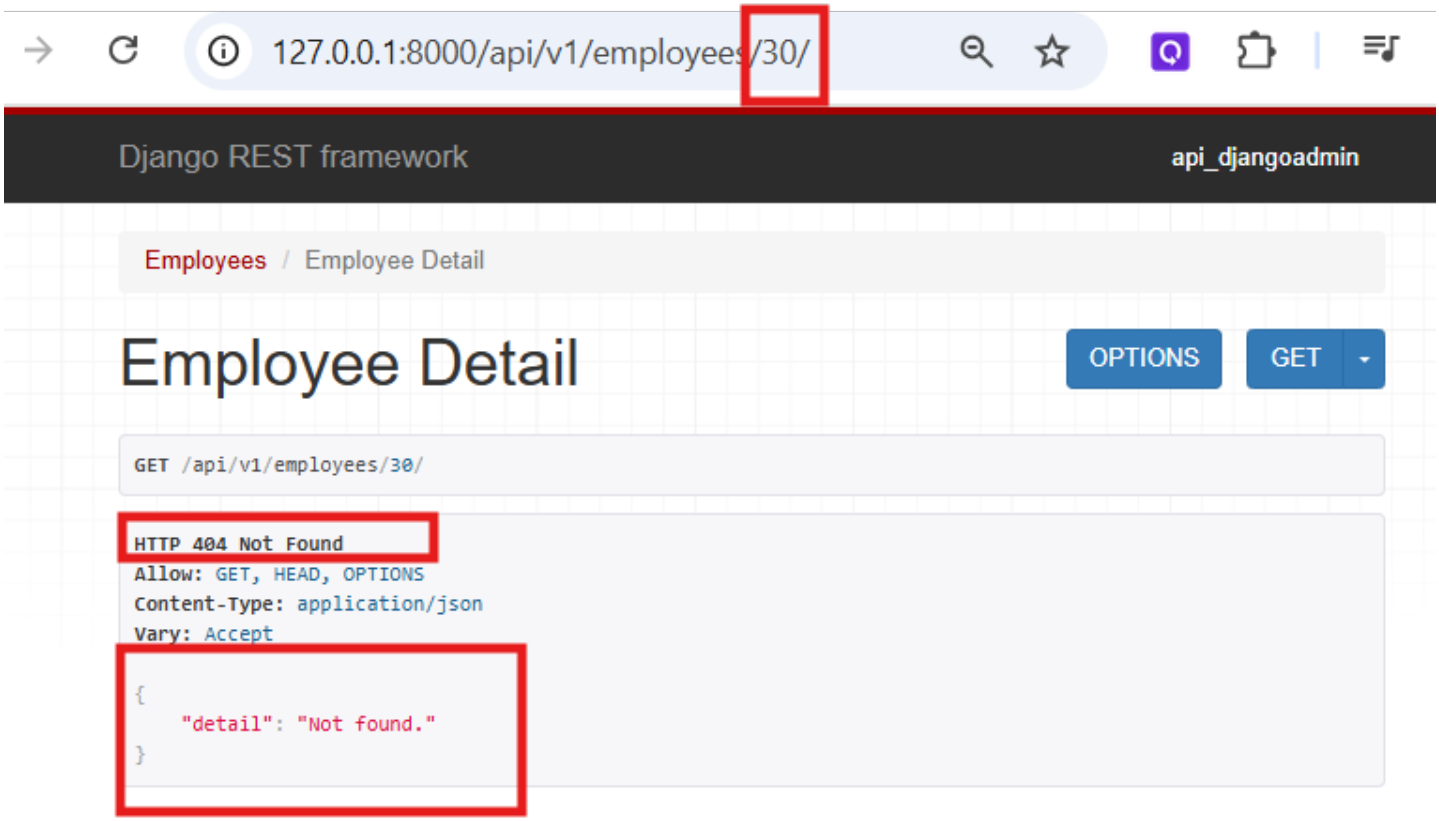


4. Update the API\VIEWS.PY to include EmployeeDetail class:

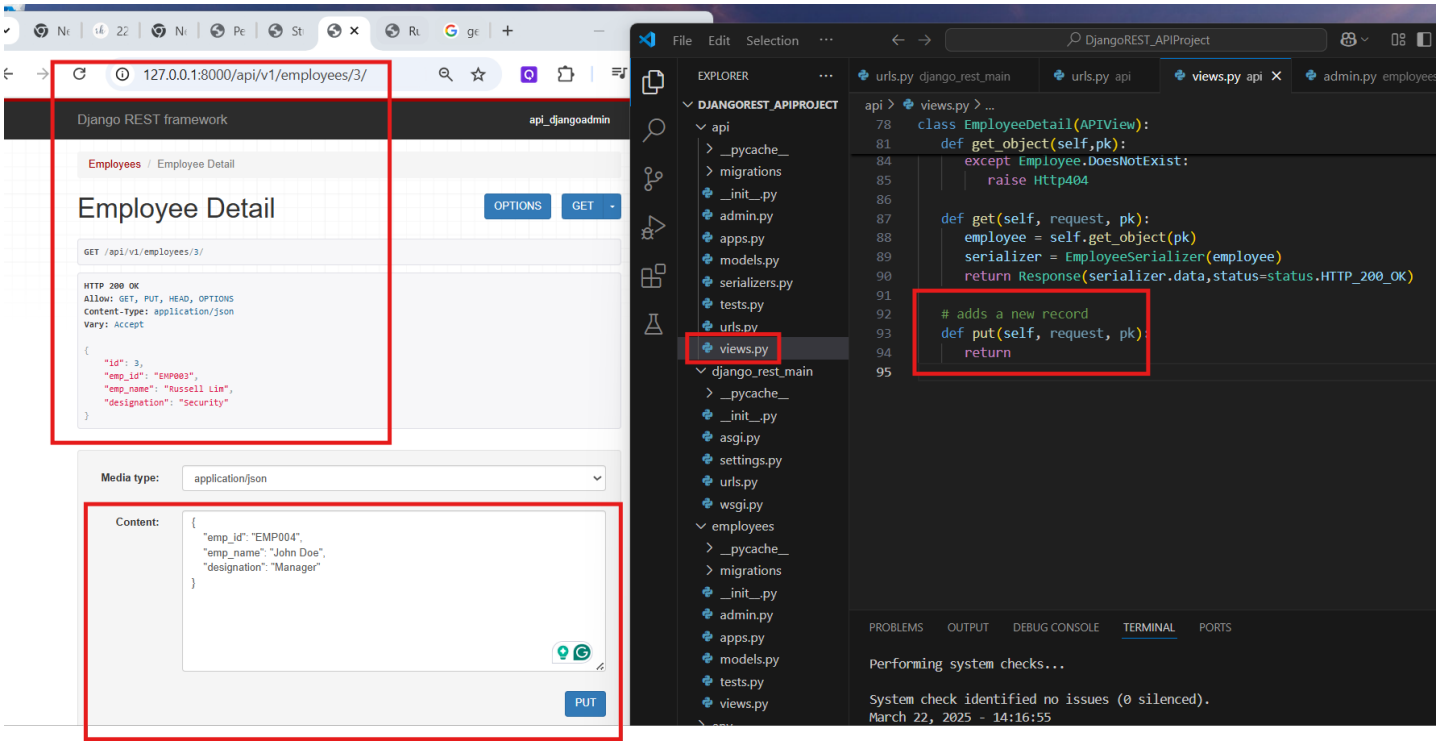
To test, add the path:

`http://127.0.0.1:8000/api/v1/employees/3/`

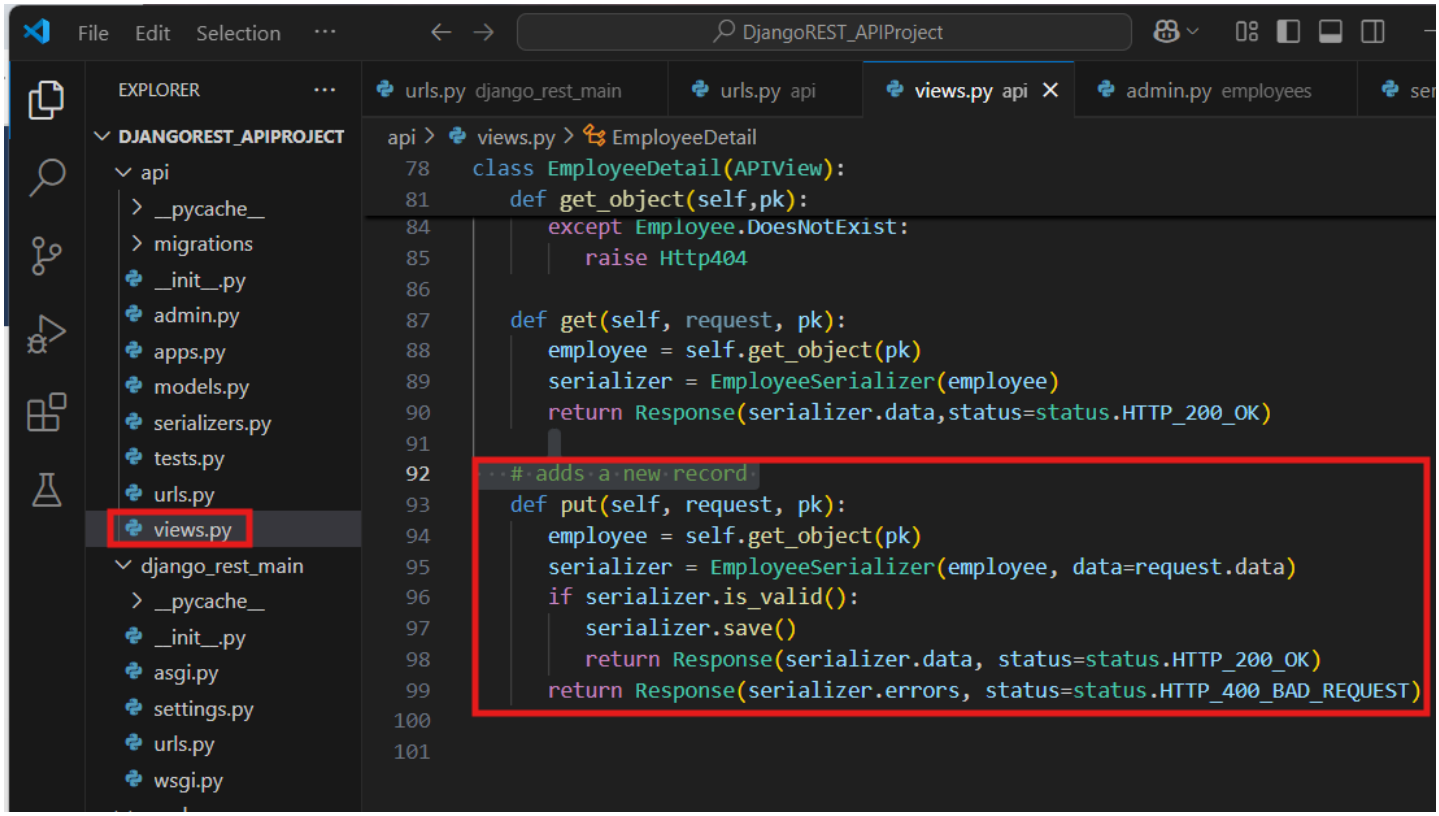
`And if the record is not in the table model:`



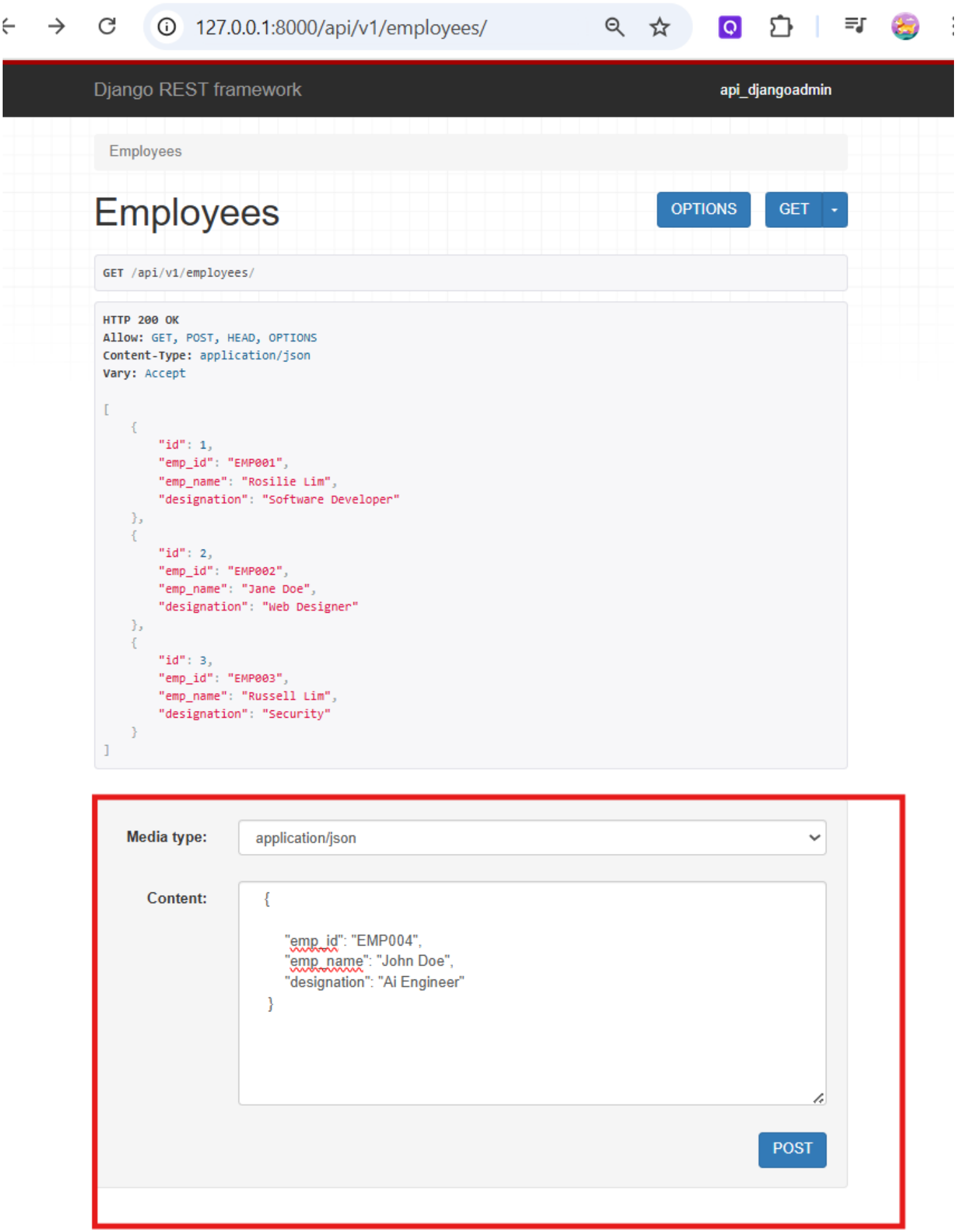
5. To update a single record, update the API\VIEWS.PY as:



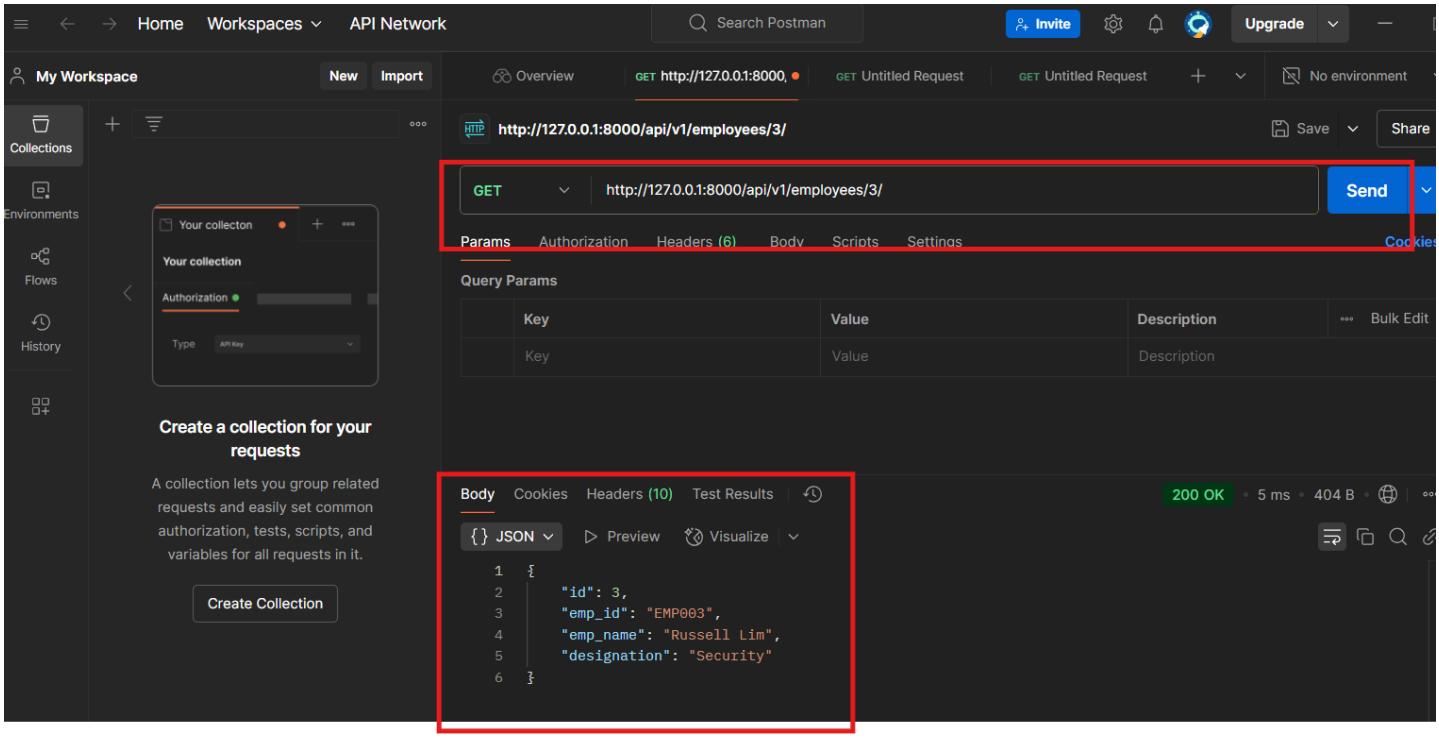
This creates the form immediately but this causes an error when you submit the PUT button. So update it as:



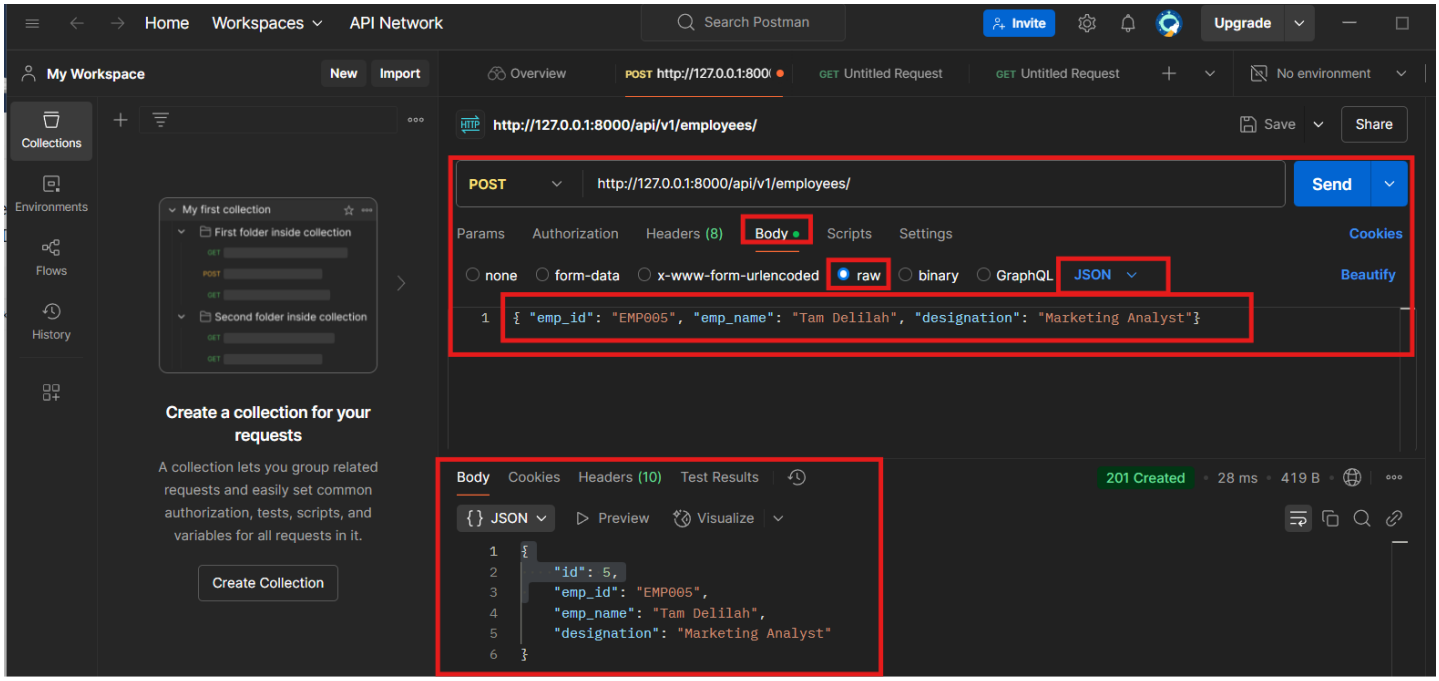
Add the new record:



6. Using POSTMAN:

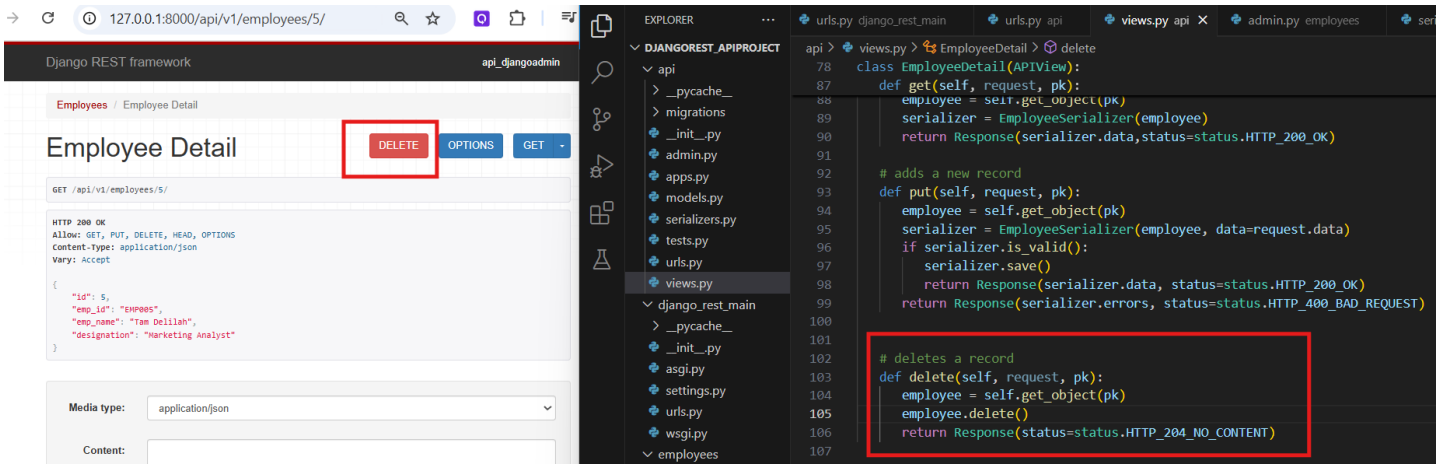


Using POSTMAN to store, choose POST, then select BODY, then RAW, then JSON. Add your records then select the SEND method.



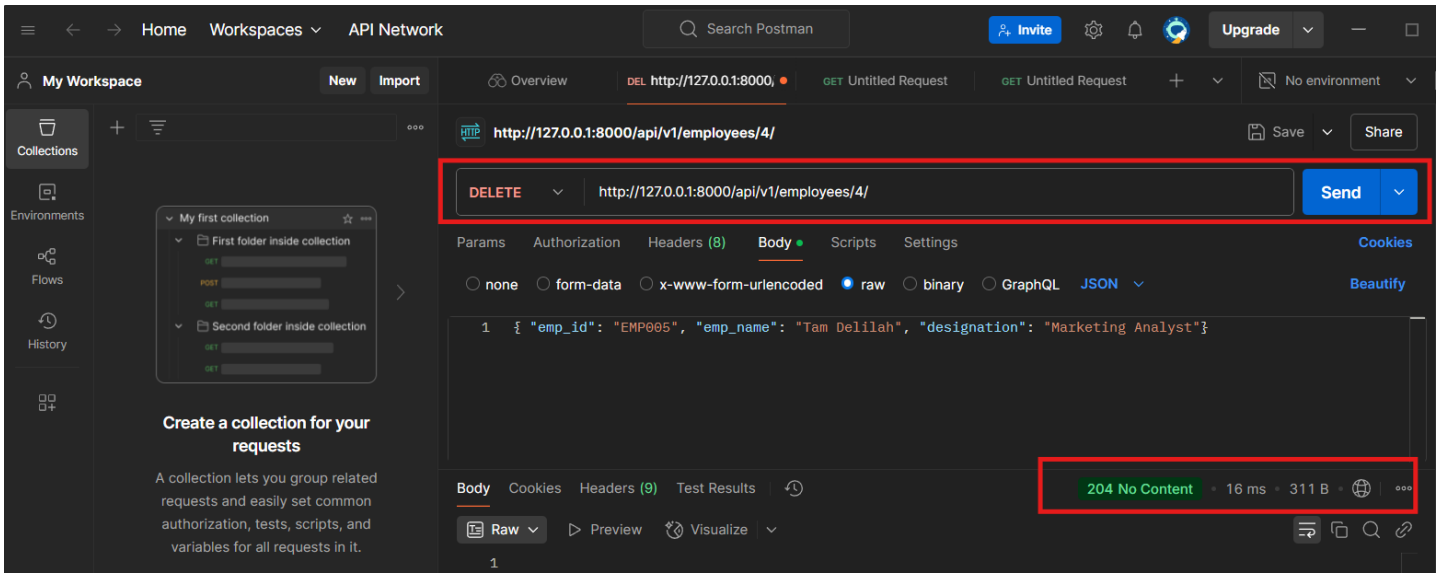
.

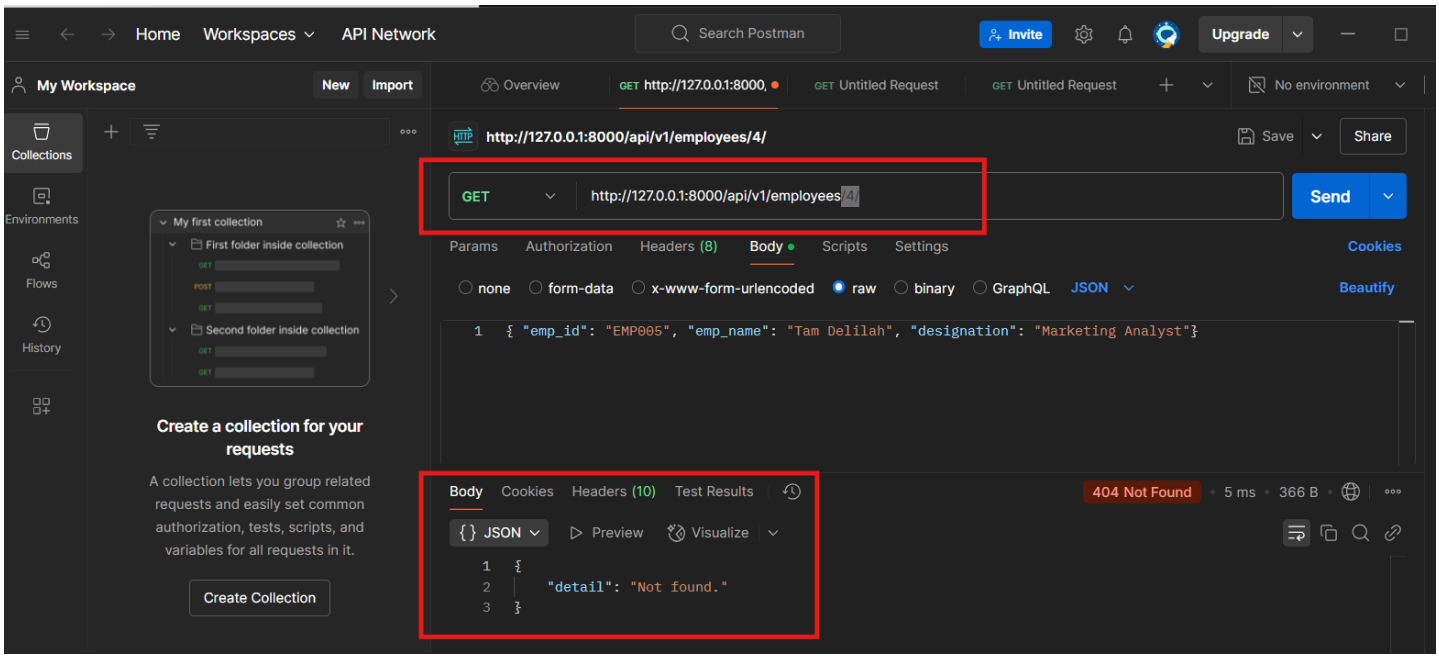
7. To delete a record, create a DELETE method and update as:



We deleted record ID = 5.

8. In POSTMAN, we can also issue a delete option:





Copyright © Personal Digital Notebooks | By Rosilie | Date Printed: Nov. 4, 2026, 7:13 a.m.