**Django Payment Service Report**

Following on from the first coursework for this module, I have implemented a Django web service which functions as a Payment API. This proved to be a relatively simple task thanks to the detailed OpenAPI3 specification which I defined as part of the first coursework. The specification outlined each of the API endpoints which would be required for a payment service, and provided exact details of the request and response bodies which the endpoint would use. It also outlined the different objects which would be used, which was used to easily define the database schema.

To start, the Django project and app were created which automatically created the base files which would be needed for the web service. The next step was to then define the database schema, which in our case stores Transactions, Billing Address’s, and Card Details (Card Details is foreign key of Transaction, Billing Address is foreign key of Card Details). The database also stores Payment objects, which do not have any actual purpose, but serve as a log of payments made because we will not actually be taking real money from cards.

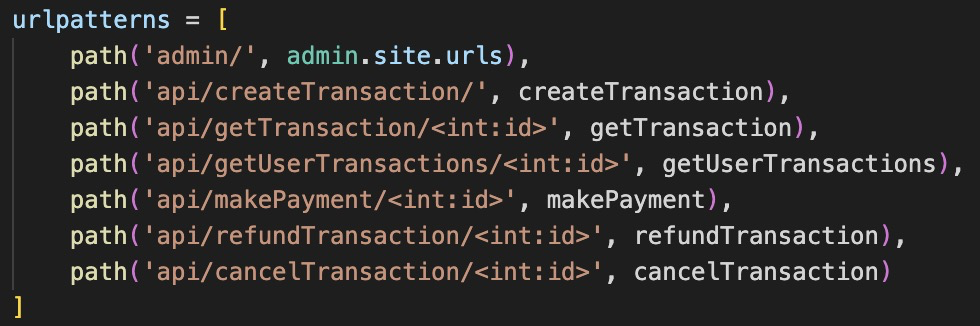
The schema is as follows:





The database was then set up using the makemigrations and migrate commands from the project’s manage.py file.

The next step was to define each of the endpoints and URL’s for the app. Each URL was added in line with the initial specification and the corresponding function was created, initially checking that the right type of request (GET or POST) was being used, and then returning a simple HTTP response.



After testing that all of the API routes were working and provided a sample response, it was then time to build them to incorporate the required functionality. These were incorporated in the same order as above, as it made sense to be able to create transactions first so that the other functions could be tested.