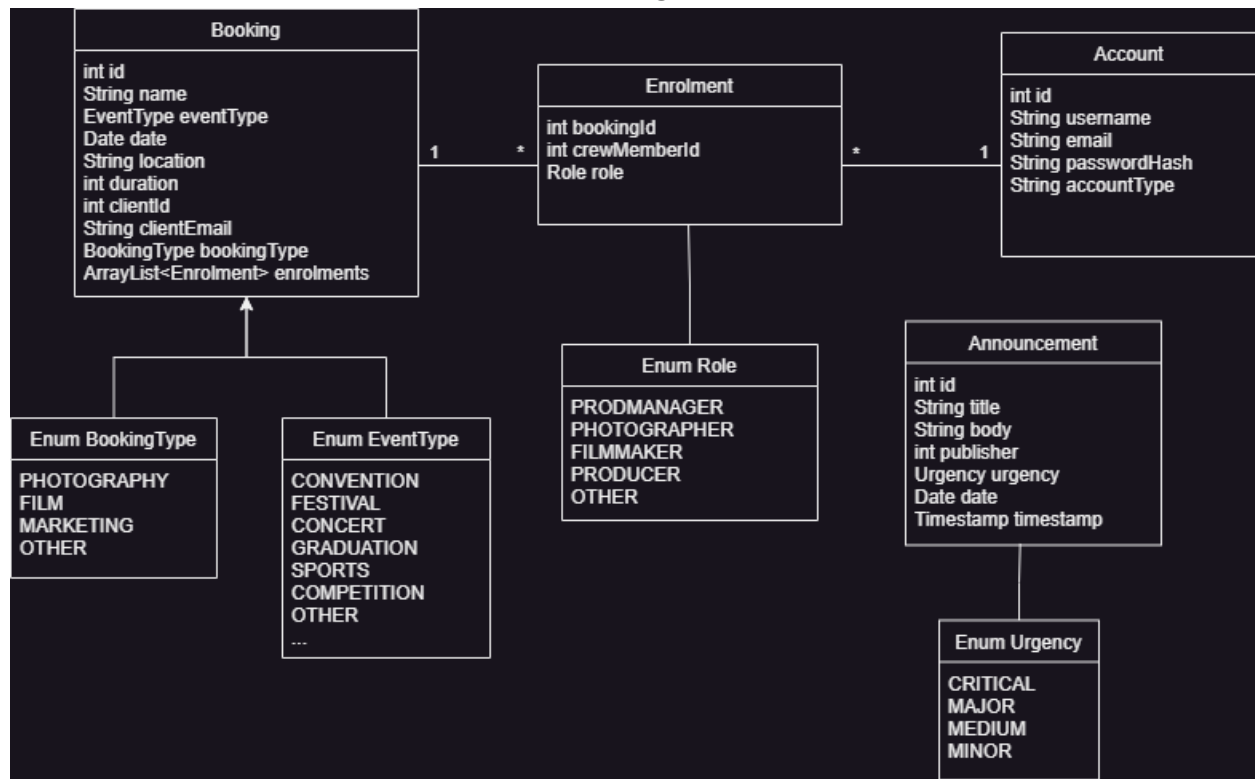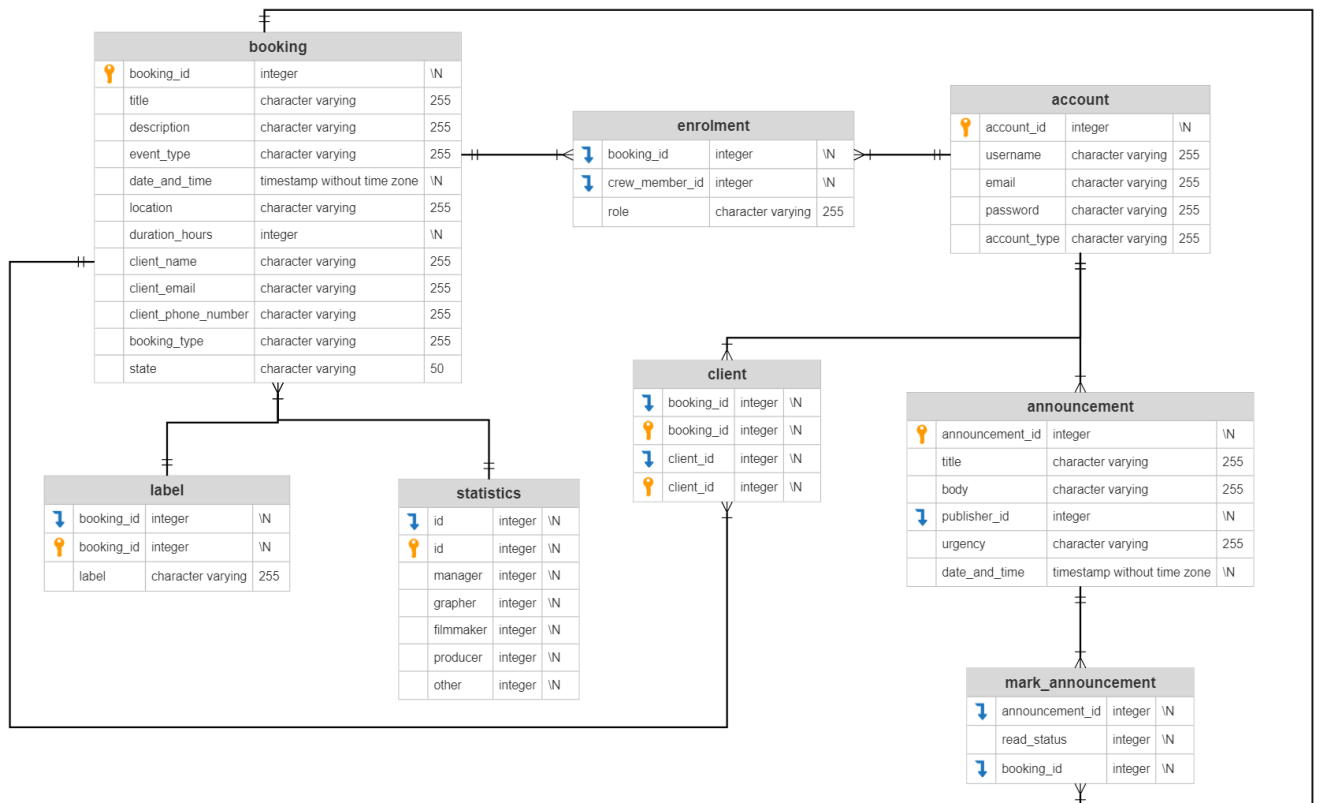# Assignment 2

UML Diagram

This is the UML diagram we initially came up with. We decided to have enum for the booking type, event type and role in order to better match the specification given to us in the project proposal. In addition, this helps us with data validation since we can check if the data is contained in the enum. This class diagram will definitely change in the future, and according to feedback it is still missing some data such as the client name and booking state in the Booking class.
We will continue to make changes to this basic structure we have outlined here.

Database Schema

booking

booking_id	integer	\N
title	character varying	255
description	character varying	255
event_type	character varying	255
date_and_time	timestamp without time zone	\N
location	character varying	255
duration_hours	integer	\N
client_name	character varying	255
client_email	character varying	255
client_phone_number	character varying	255
booking_type	character varying	255
state	character varying	50

enrolment

booking_id	integer	\N
crew_member_id	integer	\N
role	character varying	255

account

account_id	integer	\N
username	character varying	255
email	character varying	255
password	character varying	255
account_type	character varying	255

label

booking_id	integer	\N
booking_id	integer	\N
label	character varying	255

statistics

id	integer	\N
id	integer	\N
manager	integer	\N
grapher	integer	\N
filmmaker	integer	\N
producer	integer	\N
other	integer	\N

client

booking_id	integer	\N
booking_id	integer	\N
client_id	integer	\N
client_id	integer	\N

announcement

announcement_id	integer	\N
title	character varying	255
body	character varying	255
publisher_id	integer	\N
urgency	character varying	255
date_and_time	timestamp without time zone	\N

mark_announcement

announcement_id	integer	\N
read_status	integer	\N
booking_id	integer	\N

Our database schema is similar to our UML diagram in many ways, however you will notice that we have added some missing information that was not present in our UML diagram such as client name and state in the Booking class. One thing we learned during the sprint review through feedback was that the statistics table is not necessary. These statistics can be calculated directly using a query, or kept in a view to be easily accessed.

It should be noted that to improve security, it will be important to add a salt to the account table. This should protect against rainbow attacks. However, we will also need to implement functions for critical features such as account registration. This would also help secure our database, which is currently one of our weak points.