Moffat\_Bay DB Tables (ERD setup):

* User:
  + UserID (IntField, unique=true, auto increment, primary key)
  + Email: (emailfield, unique=true, not\_null)
  + Password (Hashed password)
  + First\_name (charfield(40), not\_null)
  + last\_name(charfield(40), not\_null)
  + active(Boolean field, default=true)
  + staff\_status(Boolean field, default=false)
  + superuser(boolean field, default=false)
  + last\_login(datetimefield, default=null
  + date\_joined(datetimefield, default=now)
* CustomUser:
  + User (models, onetoonefield, on\_delete=models.CASCADE) \*\*(links to user model)
  + Street(charfield(100), default=null)
  + City(charfield(50), default=null)
  + State(choicefield(dictionary of states/abbreviations), max\_length(50), default=null)
  + Zip(intfield(5) default=null)
  + Phone(phonenumberfield, default=null)
  + Image(imagefield, default=’default.jpg’, upload\_to=’profile\_pics’) \*\* optional profile pic, can be uploaded on update profile page, if not will remain as a default image\*\*\*
* Room\_choices:
  + choiceID(IntField, unique=true, auto increment, primary key)
  + roomsize(charfield(10) not null
* stay\_costs:
  + priceID(intfield, unique=true, auto increment, primary key)
  + guests(intfield, unique=true, max\_value=5, not null)
  + price(intfield, default=115, not null)
* rooms:
  + roomID(IntField, unique=true, auto increment, primary key) \*\* will serve as room number \*\*
  + size(choicefield(dictionary object from room\_choices) not null
  + booked(Boolean, default=false) \*\*this will change to true when booked, so system can filter availability of rooms\*\*\*
* reservations:
  + reservationID(intfield, unique=true, auto increment, primary key) \*\* will serve as confirmation number\*\*\*\*
  + userID(user object)
  + roomID(room object)
  + guests(intfield(1) max\_value=5, default=1, not null)
  + totalprice(intfield, not null) \*\*\* cost will be calculated based on data selected with registration form
  + checkindate(datefield, not null)
  + checkoutdate(datefield, not null)

notes/thoughts:

first, while we need to create the ERD, MySQL code, and populate a MySQL database for this assignment, we really wont need to use all of it once we get going on coding the project. Each Django project comes with a sqlite3 database, which we are using for our development environment. Once we get the project deployed in a few more weeks, then it will migrate to the MySQL, but we really will only need the code to populate the tables. Django handles all of the database stuff behind the scenes, when writing the back end code, all you have to do is create the database table ‘models’ in the models.py files, then run:

*‘python manage.py makemigrations’*

*‘python manage.py migrate’*

The make migrations command looks at the current tables, compares any changes, and creates the sql scripts it needs to make the changes. The migrate command simply applies those changes for you. If you look in the project folder- moffat\_bay/users/migrations/0001\_initial.py you can see how Django created the python code needed to create all of the sql user profile table, this was done with the makemigrations command, then to run, it uses the migrate command to apply it, making the changes with python code. any changes we make to the models within the models.py file, Django can detect and will automatically write the python code to make the changes for us.

What I’d like to do, is yes we have to go through the motions this week for the assignments, and we can use the populate scripts later on for the deployment. We can use the ERD for the overall design, which still will need to be coded in the models.py files.

Also- I tried to simplify the main user model, although in reality it’s a bit more complex. When you create a Django project from scratch, it automatically creates the user model, with multiple different tables that control different aspects from authentication (password hashing, etc) to user access levels (user, staff (like a manager), and superuser (us as admins)). All of these are combined into one admin level page. if you do a pull/fetch from the repo, open the moffat\_bay project folder (notice the lowercase) in a terminal window, run *‘python manage.py runserver’*  it will start the server. Next open a browser, and go to 127.0.0.1:8000/admin \*\*\*notice the unusual port #\*\*\*\*. You can log in with:

Username: bkyncl

Password: Bravo123!

This is the admin page. the levels on the left are the tables of the database, and will show a selection of the data stored within that table. For example, go to the USERS/Profiles tab, and it will show a link for the profiles: ‘jane doe Profile’, yours, then mine. If you click on one, it will open up a populated form, filled in with what it has stored in the database table, but it allows you to change the data and save. If you change it on this page, it will save it to the database table when you hit the save button. If you add another it will create a new record in the database table, etc.

For any models/database tables we want to show up on this admin page – which should be all of them, they just have to be registered in the admin.py file with the line: ‘admin.site.register(modelName)’ and that model will now appear on the admin site where anyone with staff/superuser status can see/edit them. We can also get into different user level permissions, but with it just being us I don’t really think its needed. When a new user registers they will be saved as just a regular user, so it wont be any issue.

That’s all I can think of for now. If you have any questions/thoughts please reach out.