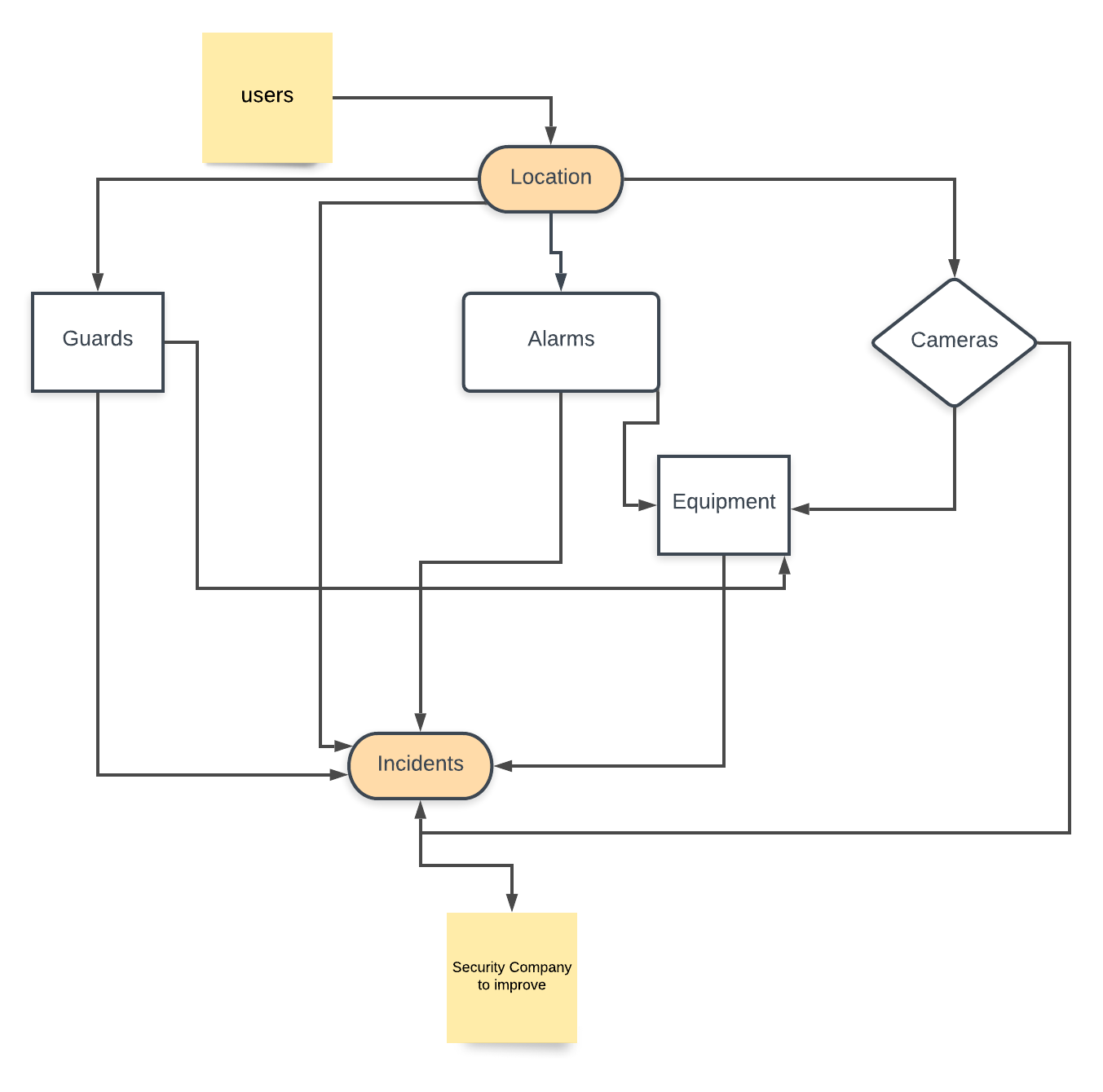
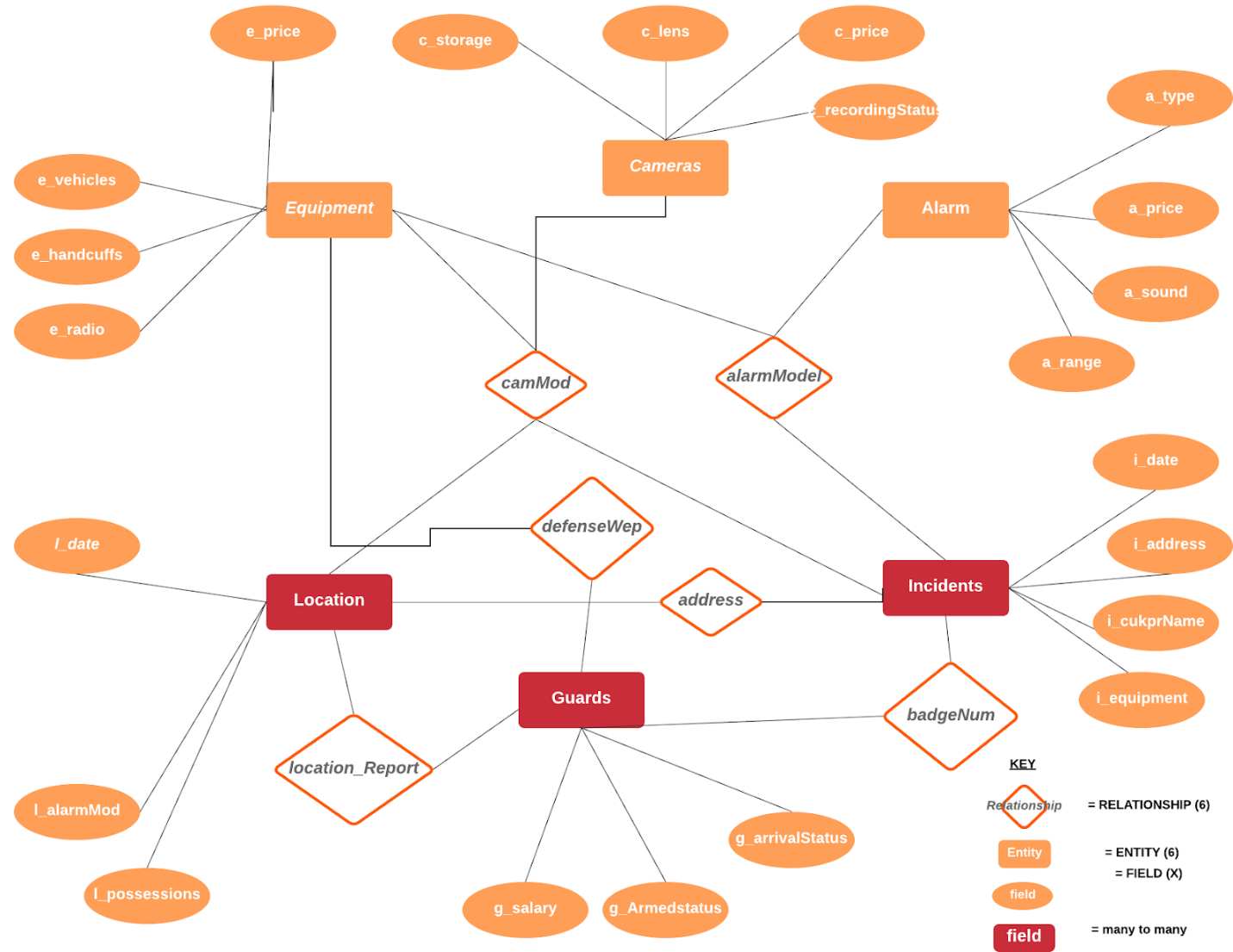
Presentation Outline

Our system was built in two parts; first was the database, titled Security. It has 6 tables: alarm, camera, equipment, guard, incident and location. Each table is filled with its own columns containing appropriate data. The next part of the system was a user interface, coded in Python. This interface allows the user to run common intended uses for our database, such as connect to it, create a table, add data to the existing tables (or even the newly created one), run queries on the database to return crucial/important information for security companies, and disconnect from the database when operations are done.

UML Diagram.

****

E/R Diagram.



Relational schema.

* Location(l\_date, l\_camMod, l\_address, l\_badgeNum, l\_alarmMod, l\_possessions)
* Alarms(a\_model, a\_type, a\_price, a\_sound, a\_range)
* Guards(g\_salary, g\_armedStatus, g\_arrivalStatus, g\_defenseWep, g\_badgeNum)
* Equipment(e\_price, e\_alarmMod, e\_camMod, e\_vehicles, e\_handcuffs, e\_radio, e\_defenseWep)
* Cameras(c\_model, c\_storage, c\_lens, c\_price, c\_recordingStatus)
* Incidents(i\_alarmMod, i\_camMod, i\_date, i\_culpritName, i\_equipment, i\_address, i\_badgeNum)

We used a database called Security, with 6 tables each having one to seven columns for appropriate data. We felt the data we were implementing into the database was appropriate as this data could be useful for security companies looking to improve the security systems put on homes (such as checking for incidents that happen at locations, the cameras and alarms there, equipment used, guards on the scene, etc.). The user interface was coded using Python and followed a very similar structure to Lab 7 and the in-class demonstration for connecting to a database using Python. Following these examples, we mimicked and then expanded upon them to make a user interface that provided more options for the user.