For this project, we will need to use database management to keep track of various details such as user accounts, orders, reservations, restaurant details and more. However, in order to utilise the MVC model we need to create model objects using OOP. For this reason, we are using sqlite3 which corresponds objects to tables in the database. Objects can be defined with methods referencing the database for things like validation and their attributes can correspond to records in the database. This means that, given a unique key, data can be read out of the database to instantiate an object. A database class can also be defined with methods to create tables and add data to tables. This approach to database design allows us to use relational language to query the database to get the information for objects we instantiate, and then "programs are used to update [database] record[s]" (Banchiihon, 1988).

Though sqlite3 is useful to meet specific requirements of this project, there are also downsides for sqlite3 as compared to other database languages. For example, it "does not support stored procedures or triggers, which are common in other databases" (Julles, 2023). Being that it works with OOP, it only needs to store the information about objects, as when a record is selected from the database, it will be instantiated as an object which contains its own methods. This means that we need less use of triggers in our database. Sqlite also doesn't support user-defined functions, which limits the types of queries that can be run from the database. However, since we only really need a few specific pieces of data to instantiate and refer to objects from the database, less detailed queries are needed.Sqlite3 also allows us to retroactively fill in missing ID numbers, something which mysql connector cannot do, only increasing ID fields even if previous numbers are deleted.

The framework we are using to develop this project will be Python Flask. Not only are we already familiar with this framework, but it is very convenient for website development. Python Flask allows for easy and quick development of RESTful APIs. RESTful APIs are stateless, meaning every request is independent, and are client-server based. This means that the clients and servers are independant and communicate through an interface. This is useful for a website application as "either side can be replaced as long as the interface stays the same" (Chan, Chung, Huang 2019), meaning that the clients and servers interact through the web page and are otherwise independent, meaning better security and scalability (which is good considering the number of clients there will be for out system).

Flask also facilitates the use of sessions which maintain the state between requests. "They allow storing user-specific data…without constantly passing information back and forth between the client and the server" (JetBrains Academy n.d.). This is extremely useful, as we ran into some trouble on our previous assignment with needing a class with a method to set the information (instantiate a class) for the current user. Sessions provide a much more convenient way to store necessary data (as well as the need for less database requests, as we won't need to get all of their data to create an instance of a class).

Compared to other frameworks such as Django, Flask is "ideal for simple, extensible web apps", which makes it easy to edit and add more web pages, which is useful as this project requires a lot of different view screens. Flask also offers a plethora of tools and libraries such as form validation which makes user interaction and acting on user input very easy to create. While Django is more efficient for development time due to Flask's "build from scratch" approach, we have chosen Flask as it is generally a lot easier to use and our familiarity of it puts it at a time advantage over Django, which we would have to learn how to use as

“Django is a more complex web solution that requires extensive expertise” (Bahgat, 2023). Moreover, Flask is an incredibly versatile framework, being small ensures better performance and utility. Flask is “an excellent choice to optimise the creation of a web solution, with the opportunity to build both simple and more sophisticated web applications” (PLANEKS, 2022).

Bibliography:

Banciihon, F. (1988) *Object-oriented database systems*. In: *Proceedings of the seventh ACM SIGACT-SIGMOD-SIGART symposium on Principles of database systems*. New York, NY, USA, ACM.

https://www.linkedin.com/pulse/part-3-introduction-differences-between-sqlite-other-sql-julles#:~:text=While%20SQLite%20is%20a%20powerful,executed%20on%20the%20database%20server.

https://uwe.primo.exlibrisgroup.com/discovery/fulldisplay?docid=cdi_proquest_ebookcentral_EBC5984824&context=PC&vid=44UWE_INST:44UWE_INST&lang=en&search_scope=MyInst_and_CI&adaptor=Primo%20Central&tab=Everything&query=any,contains,python%20flask

https://hyperskill.org/learn/step/33905

https://kinsta.com/blog/flask-vs-django/#:~:text=Flask%20is%20a%20lightweight%20micro,in%20tools%20for%20rapid%20development.

https://www.planeks.net/why-use-flask/

SQLAlchemy
https://www.shiksha.com/online-courses/articles/move-beyond-flat-files-sqlite-and-sqlalchemy-in-python/#:~:text=SQLAlchemy%2C%20on%20the%20other%20hand,data%20and%20write%20complex%20queries.