Design Patterns

This section discusses which design patterns and best practices were used throughout the design process of this system.

The architecture used for this design was cloud native design. Since the system needs to be connected to a learning database, it made more sense in terms of speed, safety, and scalability to have all resources be stored in the cloud rather than internal servers. It also allows the chatbot to easily perform other services down the line, or even work across multiple dealerships as all the data is stored in the cloud.

Some design patterns that were used were generalization classes, description classes, states, and strategy patterns. Description classes were a useful way for defining what was required to be stored in each component of the database such as cars; each car had tons of details to track, but a description class helped keep the diagrams uncluttered. The strategy and generalization patterns seemed to be very similar to each other. By defining all user behavior as one of 4 intents or strategies it made the system much easier to generalize what actions needed to be accomplished.

Some design principles that were used were KISS and YAGNI. With this design, it would have been easy to let scope creep and have the chatbot do everything for the company, but it seemed more manageable to start the bot off with a few core responsibilities with the potential to add more responsibilities down the line as they were needed. The focus of this design was to have 3 primary workflows the chatbot could complete before listing off every single possible behavior it could help with which would have led to both unnecessary complexity and even behavior that is not required for successful operation.