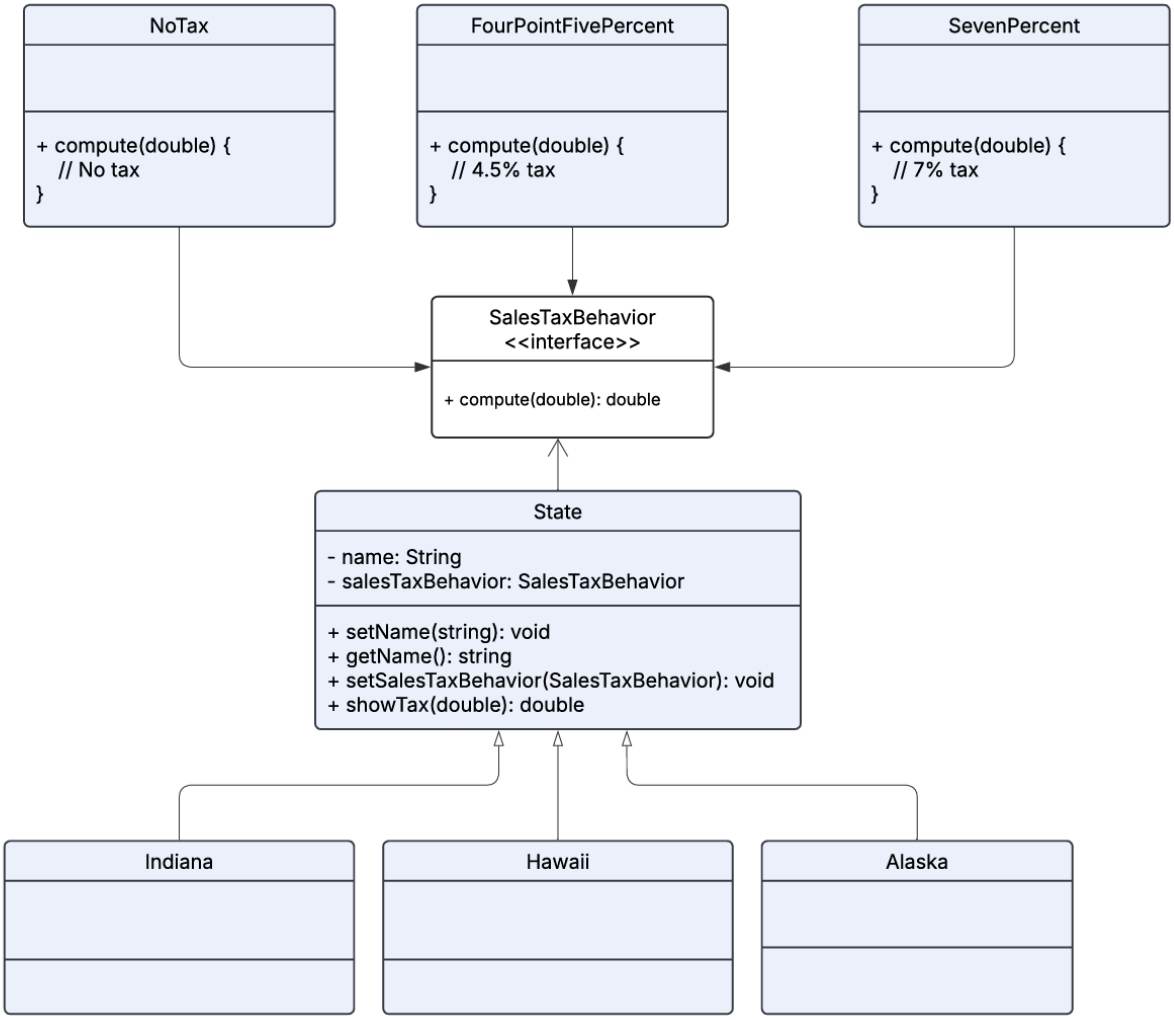
Problem 1:

Insert the UML diagram here



NOTE: Notice that the subclasses for state are essentially empty. This is because you can implement showTax in the abstract class. This is efficient because it's assumed that all states are going to have the same message when showing tax. Moreover, The reason the subclasses still exist is because you can still hard-code the names of the states in the constructors.

Problem 2:

1. IS-A since it inherits from a base class.
2. IS-A since it inherits from a base class.
3. IS-A since it inherits from a base class.
4. IS-A since it inherits from a base class.
5. HAS-A because Duck is being composed by FlyBehavior.
6. HAS-A because Duck is being composed by QuackBehavoir.
7. IS-A relationship because Quack implements QuackBehavior so it's of type QuackBehavior.

8. IS-A relationship because Squeak implements QuackBehavior so it's of type QuackBehavior.
9. IS-A relationship because MuteQuack mplements QuackBehavior so it's of type QuackBehavior.

Problem 3: Insert the UML diagram here for the auctioneering system