1. Strategy Pattern

Diagram

Description automatically generated

Implementing a strategy pattern will provide information about storage within the document. This will allow for maintainability within the code as updates are performed when new features and requirements are added.

1. Read Factory

Diagram

Description automatically generated

The code will utilize factories to create more expressiveness and rely on dependencies. In addition, it will allow for the extension as trips get designed and modified without adding code.

1. Write Factory

Diagram

Description automatically generated

The parent class will need to choose the creation of the object, so a factory design pattern is crucial here. This also helps reinforce the ability to leave the code and come back and resume. This would be ideal if travel agents get a phone call during non-working hours and could still assist the customer.

1. Decorator

Diagram

Description automatically generated

To prevent travel agents or clients from being delayed in getting their itinerary, using a decorator will prevent any lag. Many behaviors belong to more than one function and will need to cache function results.

1. State

Diagram, schematic

Description automatically generated

Because premium clients are busy and don't have time to sit down and walk through planning a trip, implementing the state pattern will allow agents to return to an incomplete trip and resume work. Without this function, travel agents would need to manually record or start over at the beginning every time, potentially losing business.

1. Package Singleton

Diagram

Description automatically generated

As premium travel grows a database will need to be implemented to share different parts of the program. Having this singleton set up will disable other creation of objects and have a single instance available to all clients.

1. Person Singleton

Diagram

Description automatically generated

While singletons have their place, they do have a pattern for hidden dependency, even if it's just bad software design. It can be hard to test with singletons because adding a person to the trip it's completed later in the process. When testing is required, it will be more complex and costly for the company to create a mock instance of a singleton.

1. Travel Agent Singleton

Diagram, schematic

Description automatically generated

Because the singleton acts as a global access point, using a singleton to store travel agents can minimize internal staffing issues. If an employee has to leave or is on vacation, another travel agent can quickly resume their work without service disruption.