Question 1

**Using the airline reservation system as an example, under what conditions would a transparent**

**façade be more appropriate to use than an opaque façade?**

From the article: https://msdn.microsoft.com/en-us/library/orm-9780596527730-01-04

The façade described in the preceding example is *opaque*, in that the subsystems cannot be accessed except via the Facade object. This requirement might be too stringent. Suppose some users want to get at the individual operations of particular subsystems. We can change all the internal modifiers to public, which will make the façade optional, or *transparent*. That is, as well as being able to go through the Facade, the client will be able to instantiate SubsystemA directly, for example, and then call A1.

So using a transparent façade might make the most sense when you need to complete a full operation and save data, such as actually creating a ticket and reservation object. This is different from performing an operation just for informatin gathering purposes, like displaying a list of current flights and rates. In cases like that, an opaque façade could be used because you only need to invoke certain methods to gather this information.

Also, security could be a concern, maybe opaque facades make sense to use for items available to the general public online whereas a transparent façade is used by employees logged in to a system, or possible different employees could have different facades of different levels of “opacity”, i.e. you make some methods public versus private as appropriate.