CS 4345 Software Engineering Principles
Sample Exam

Instructor: Dr. Jia Zhang

Here is a simplified scenario of a software system supporting uber-like car sharing services. Your company developed a taxi reservation system before, and you are expected to design the new system as a software architect.

[16] 1. As an architect, you should provide a 4+1 View design for the system using the corresponding UML diagrams, e.g., scenario view (use case diagram); logical view (component diagram); development view (class diagram); progress view (sequence diagram); and physical view (deployment diagram). [Hint: we had in-class exercise on this.]

[10] 2. Your system will offer three basic service levels. carPOOL is the least expensive level of service, in which the customer may share the ride with another passenger going in the same general direction. carX is a level of service in which the rider will get a private ride. carBlack is a level of service in which the rider is provided a black luxury car. Each such service will be handled by dedicated subsystems, but you surely will not let customers know all such details. Instead, you would like to provide an interface for customers to select from, and based on their selections, you will direct them to corresponding modules to calculate their transportation fees. Two design patterns may be used here together. Briefly explain your recommendation, and draw a class diagram to explain your design (maybe with short descriptions if appropriate). [Hint: Façade, strategy]

[6] 3. Your system may offer more service levels at some cities or countries. For example, you may want to offer a service carGo in India, which provides for a ride in a hatchback. For another example, carEATS will allow users to have meals delivered from participating restaurants by your registered drivers. You want your system to be designed to provide such car sharing services in many cities and countries, meaning that although all cities will provide the basic levels of services, each city may configure her own customized services. Which design pattern will you recommend to use? Briefly justify your suggestion, and draw a class diagram to explain your design (maybe with short descriptions if appropriate). [Hint: template]

[6] 4. Your system will take payment through credit cards. But you do not want to implement your own credit card payment system; instead, you would leverage PayPal service for now. You decide to implement a handler locally in your code which prepares PayPal-compatible data format and forward the call to PayPal. Any design pattern you see appropriate here? Briefly justify your suggestion, and draw a class diagram to explain your design (maybe with short descriptions if appropriate). [Hint: proxy]

[6] 5. You decide to adopt a dynamic pricing model. The same route costs different amounts at different times as a result of factors such as the supply and demand for drivers at the time the ride is requested. When rides are in high demand in a certain area and there are not enough drivers in such area, fares increase to get more drivers to that area and to reduce demand for rides in that area. Therefore, you will calculate the transportation fee for a rider who will provide detailed information for you to calculate. Any design pattern you see appropriate here? Briefly justify your suggestion, and draw a class diagram to explain your design (maybe with short descriptions if appropriate). [Hint: strtegy]

[6] 6. Your system may periodically send coupons to all registered riders automatically. Will you recommend to use any design pattern to realize this push-mode notification? Briefly justify your suggestion, and draw a class diagram to explain your design (maybe with short descriptions if appropriate). [Hint: observer]

[6] 7. You want to let the system serve as a social network for riders. A rider can post a message to other riders, sometimes broadcast to other riders. Any design pattern you see appropriate here? Briefly justify your suggestion, and draw a class diagram to explain your design (maybe with short descriptions if appropriate). [Hint: mediator]

[6] 8. You plan to implement a sharableCar object in your system, which could be categorized into normal car, luxury black car, SUV, wheelchair accessible transport, etc. For each category of car, it may carry and display different features, as well as different transportation fee calculation methods. To keep the code consistent with extensibility, you would like to create such sharableCar object without exposing the creation logic and refer to newly created object using a common interface. Any design pattern you see appropriate here? Briefly justify your suggestion, and draw a class diagram to explain your design (maybe with short descriptions if appropriate). [Hint: factory]

[6] 9. You plan to leverage some legacy code in your company that implements online car reservation function. That means you want to turn those legacy code as reusable components in your new system but with different interfaces. Any design pattern you see appropriate here? Briefly justify your suggestion, and draw a class diagram to explain your design (maybe with short descriptions if appropriate). [Hint: adapter]

[6] 10. As you are expanding your services into new cities, you may need to add some new features/options to some services. You don't want to change the code structure (because some cities may need the original version) while you add new functions. Any design pattern you see appropriate here? Briefly justify your suggestion, and draw a class diagram to explain your design (maybe with short descriptions if appropriate). [Hint: decorator]

[6] 11. Like many startups, your software may begin its journey with a monolithic architecture, built for a single offering in a single city. Having one codebase seemed "clean" at the time, and could solve your core business problems, which included connecting drivers with riders, billing, and payments. It was reasonable back then to have all of your business logic in one place. As you rapidly expand into more cities and introduced new products, this architectural style should be changed. As a software architect, which architectural style(s) will you recommend to? Briefly explain why you recommend such architectural styles, any advantages? Disadvantages? Draw a component diagram to explain your design (maybe with short descriptions if you see appropriate).

[6] 12. To be practical, your software will require the drivers to have a smartphone and the users must have access to either a smartphone or the mobile website. While drivers and users may use different types of devices to view your app, which architectural style/design pattern would you see appropriate here? Briefly justify your suggestion, and draw a class diagram to explain your design (maybe with short descriptions if appropriate).

[14] 13. Your app is becoming very popular. In order to support rapidly increasing user base (i.e., traffic) while quickly developing new functions and deploying it, you decide to deploy and run your app in the cloud at a data center. In order to smooth and facilitate this process, you have decided to enforce DevOps in your project team.
[4] (1) Briefly explain your motivation of using DevOps.
[6] (2) You want to use container management technique. Briefly explain your reasons and briefly explain how you will do it using the latest container management technique.
[4] (3) In order to save budget, you plan to use multitenancy architectural style. Briefly explain your reasons and how you will make it work.