1. Framework yang pernah saya gunakan yaitu Echo, Fiber dan Gin.
    - Echo (Payment gateway) : Alasan utamanya sendiri karena existing framework yang menjadi standard di backend tim dan pada saat itu echo memiliki dokumentasi yang memadai. Banyak function built-in untuk middleware dan lain-lain yang mempermudah untuk proses development.
    - Gin (Payment gateway, POS, E-Commerce) : Dipilih untuk menghandle GraphQl karena dokumentasi dan komunitas yang besar sehingga lebih mudah mencari solusi jika terjadi sesuatu.
    - Fiber (Payment gateway, POS, E-Commerce, Event Management, Bid-App): Secara benchmark lebih cepat dibanding framework yang lain
2. Di golang terdapat fitur concurrency yang disebut goroutine dan channel. Goroutine sebagai worker dan channel digunakan untuk komunikasi antar goroutine. Concorrency digunakan untuk menjalankan sebuah perintah secara parallel.
3. Penggunaan waitgroup, channel atau kombinasi
    - waitgroup : jika membutuhkan concurrency tanpa saling komunikasi antar goroutine akan tetapi membutuhkan proses concurrency ini selesai baru bisa lanjut ke proses berikutnya.
    - channel : jika membutuhkan komunikasi data.
    - kombinasi: jika membutuhkan concurrency dan komunikasi data antar goroutine
4. Goroutines adalah salah satu fitur di bahasa Go yang memungkinkan concurrent programming yang mudah digunakan. Dalam beberapa project yang pernah saya kerjakan, goroutine sering saya gunakan untuk proses pengiriman email secara terpisah dengan proses yang sekarang jika ada pembayaran yang success dan jika get beberapa data menggunakan parameter yang sama, maka saya biasanya menggunakan goroutine dengan waitgroup untuk get data. Dan juga
5. Queueing adalah pola desain untuk mengelola dan memproses tugas-tugas secara berurutan. Queueing di Go biasanya memanfaatkan channels dan goroutines untuk menciptakan prose yang efisien.
6. Sebenarnya tergantung kebutuhan, akan tetapi untuk proses yang synchronous biasanya saya menggunakan gRPC atau RestfulAPI, dan untuk proses asynchronous biasanya menggunakan GCP pub/sub atau rabbitMQ. Untuk gRPC sendiri digunakan karena ringan, akan tetapi challengenya yaitu memastikan proto harus punya standard dan versioning yang musti dijaga. Resful sendiri lebih mudah untuk didevelop akan tetapi untuk response kadang lebih lambat daripada gRPC.
7. Hasilnya ada di folder code/answer_7.go
8. Yang saya lakukan adalah print error tersebut untuk mengetahui apa pesan errornya dan selanjutnya dilakukan pengecekan untuk menyelesaikan error tersebut.
9. Saya pernah menggunakan third party seperti sentry webhook teams/slack untuk mengirim allert bahwa terjadi error beserta error messagenya dan nama service tempat terjadinya error. Ini mempermudah untuk melakukan pengecekan dan penyesaian error tersebut.
10. Hasilnya ada di folder code/answer_10.go
11. Pernah, ketika mengerjakan project bidding_apps saya harus memastikan ketika ada beberapa user yang melakukan penawaran, hanya 1 request user yang valid dan tidak terjadi race condition. Yang saya lakukan untuk mengatasi ini yaitu menggunakan salah satu feature redis yaitu redis lock. Ketika mengerjakan project PG, POS dan juga E-Commerce, saya harus menggunakan GraphQl dimana itu

adalah pertama kali menggunakan itu. Jadi saya harus research, development dan untuk performance test dibantu oleh tim QA.