I. Frontend асуултууд

1. CSS-ийн block, inline, inline-block утгуудын ялгаа юу вэ?
   block -> бүтэн мөрийг эзлэнэ 2 block element байх юм бол 2 мөр болно.
   inline -> өөрийн content ийн хэмжээгээр эзэлнэ нэг мөр дотор олон inline element байж болно.
   inline-block -> нэг мөрөнд байрладаг урт өндөрийн хэмжээ өгөж болдог.
2. JavaScript-ийн == болон === хоёрын гол ялгаа юу вэ?
   const a = "1"
   If(a == 1)-> true
   If(a === 1)-> false
   == энэ нь type нь number, string байсан ч valea таарч байгааг л шалгана. харин
   === энэ нь type, value таарч байгааг хоёуланг нь шалгана.
3. Promise гэж юу вэ? async/await ашиглахын давуу тал юу вэ? Мөн error handling-ийн талаар бичнэ үү.
   Promise нь async шууд хийгдэдгүй хугацаа ордог үйлдлүүдийг алдаа гарсан хүлээгдэж байгаа ажилттай болсон гэдгийг мэдэх боломжтой.
   async/await -> жишээ нь server ээс мэдээлэл авчрах үед хүлээгдэж байгаа бас хүсэлт ажилттай дууссанг нь мэдээж болдог. Ямар байдалд байгааг нь мэдсэнээр loader харуулах эсүүл ажиллтай болсон апдаа гарсан гэх рор харуулж болно.
   error handling -> жишээ нь server ээс мэдээлэл авчрах үед алдаа гархад бариж аваж болдог.
4. React-д useMemo болон useCallback hook-үүдийг ямар тохиолдолд ашигладаг вэ? Ашиглахгүй бол ямар асуудал үүсэх вэ?
   useMemo -> их тооцоолол хийгдэж гарсан утга бас filter, sort зэрэгийн утгыг memoize хийж утга өөрчлөгдөх хүртэл render бүрт дахин бодолгүй cached утга ашигладаг. Хэрвээ томоохон тооцоолол хийх, filter, sort зэрэг дээр ашиглахгүй бол жижиг render бүрт тооцоолол хийж web удаан бас гацалттай мэдрэгдэж эхлэнэ. Их тооцоолдог үйлдэл дээр ашиглавал зүгээр жижиг үйлдлүүд дээр ашиглах шаардлагагүй.
   useCallback -> useMemo той адилхан гэхдээ утга биш function memoize хийдэгээр өөр. Render болгонд дахин function үүсгэхгүй. Child component руу function дамжуулах үед ашиглана гэхдээ жижиг handle function бас child component байхгүй бол ашиглах шаардлагүй.
5. React дээр unnecessary re-renders үүсэх нийтлэг шалтгаанууд юу вэ? Үүнийг хэрхэн засах вэ?
   Хэт олон useState ашиглах хэрвээ хоорондоо холбоотой байвал бүх state ажилна render ихсэн гацалт мэдрэгдэнэ. Child comp-д их үйлдэлтэй function дамжуулах юм child render хийж гацталд мэдрэгдэж магадгүй иймэрхий үйлдлүүд дээр useMemo useCallback ашиглавал шийдэж болно.
   useState ийг form дээр ашиглавал. useForm ашиглаж болно.
   Хэт урт мөр кодетой comp. Нэг comp ийн мөр коде нь ихдээ 250 хэтхэргүй байвал зүгээр хоорондоо хамааралгүй бол comp болгож салгах хэргэтэй.
   useEffect, useMemo, useCallback дээр dependency буруу өгөх бас өгөхгүй loop болох эрслэлтэй

III. Infrastructure ☁️

1. Terraform гэж юу вэ? Ашиглахын гол ач холбогдол юу вэ? Ойлгомжтой тайлбарлана уу.
   Terraform гараар хийгддэг байсан ажиллийг зөвхөн code оор удардаж хийх боломжтой IaC tool. Server, cloud, version, database зэргийг гараар үүсгэх биш хэдхэн code оор удардана үүсгэж, өөрчилж, устгах боломжтой автоматжуулсан томоохон tool. Гараар үүсгэн server, cloud, database project той холбох холбох үед алдаа гарах гээд нэлээн цаг ордог байсан ажилийг маш хурдан багахан code ашиглана хийх боломжтой.
2. Terraform State: State файл гэж юу вэ? Remote state болон state locking ашиглахын давуу тал юу вэ?
   Terraform State terraform.tfstate файл нь terraform ой санамж нь apply хийх үед өмнөн байх юм бол дахин үүсгэлгүй. Terraform ударжаж байгаа IaS одоогийн байдлийг хадгалах apply хийх үед өөрчлөлтийг зөв тодорхойлоход ашиглагддаг. Remote state болон state locking багаар ажилхад зориулагдсаан. Remote state нь terraform.tfstate file ийг developers болгоны computer дээр биш server дээр хадгалдаг энгэсээр developer terrafrom apply хийхэд дахин үүсэхгүй server ээс state ийг татаж авна. State locking нь developers зэрэг apply хийхээс сэргийлж өнгө. Зэрэг apply хийх юм бол нэгнийхаа recourse ийг утсгах боломжтой. State locking ашигласанаар эхэлж apply хийсэн хүнийх дуудсасны дараа дараагын хүний үйлдэл эхлэнэ зөрчил гарахгүй.
3. IaaS / PaaS / SaaS гурвын ялгааг тайлбарлана уу.
   Iaas нь өөрөө бүгдийг удирдана хянах болно. Бараг өөрөө бүх зүйлийн хийнэ.
   PaaS нь зөвхөн өөрийн code нд анхаарана server cloud release зэргийг хийхгүй.
   SaaS нь бэлэн platform ашиглана server, code, cloud зэрэгтэй хамааралтай зүйл хийхгүй.
4. Serverless computing гэж юу вэ?
   Server хянаж ударжах шаардлагагүй коде ажиллах үед л төлбөр бодогдон коде ажилаагүй үед төлбөр бодогдохгүй.
5. DDoS халдлагаас хэрхэн хамгаалах вэ?
   Rate limiting -> нэг IP аас хэт их request ирвэл block хийнэ.
   CDN -> server т хүрхийн өмнө шүүж server лүү дамжуулана.
   WAF -> bot болон сэжигтэй хүсэлтүүдийг block хийнэ
   Ачаалал ихсэх үед Load Balancer болон Auto Scaling ашигласнаар сервер уналаас сэргийлнэ. Cloud provider ийн бэлэн DDoS хамгаалалтыг (AWS Shield гэх мэт) давхар ашиглах нь үр дүнтэй.