Oppgaver

Vi ønsker at du løser disse oppgavene så godt du klarer. Hvis du f.eks ikke klarer å lage en fungerende løsning ønsker vi at du har tenkt ut hvordan det kan løses. Her er det noen spørsmål som er rettet mer mot backend og noen som er rettet mot frontend. Hos oss jobber sjelden samme person med begge deler, selv om det ikke er en regel.

Vi ønsker korte og konsise svar tilsendt på mail til [idar.borlaug@ks.no](mailto:idar.borlaug@ks.no) før 2. gangs intervjuet.

**1**. Hva er en connection leak og hvordan kan en unngå det?

**2.** SQL database

CREATE TABLE `posts` (

`id` INT NOT NULL AUTO\_INCREMENT,

`text` TEXT,

`user\_id` INT unsigned NOT NULL,

`updated\_at` TIMESTAMP NOT NULL,

`created\_at` TIMESTAMP NOT NULL,

PRIMARY KEY (`id`)

);

CREATE TABLE `post\_likes` (

`post\_id` INT unsigned NOT NULL,

`user\_id` INT unsigned NOT NULL,

`created\_at` TIMESTAMP NOT NULL

);

Lag en spørring som returnerer alle posts for en gitt user\_id som også tar med antallet likes per post.

**3.** Hva er fordelene og ulempene ved å benytte http2 vs http 1.1?  
**4.** Hva er en Content Security Policy?

**5.** Hva er forskjellen på en cookie, sessionstorage og localstorage?

**6.** Nevn noen tiltak man kan gjøre for å sørge for at en nettside er universelt utformet.

**7. Valgfri oppgave**

Du skal velg **én av oppgavene** under.

**Valgoppgave A)** Lag en applikasjon som krypterer hvert ord i teksten under, 5000 ganger. Velg krypteringsalgoritme sjølv. Samle de krypterte ordene i en array med index til ordets posisjon. Skriv arrayen til en fil.

"""

Examples

A Non-programming Example

Suppose a simple scheme for a repair centre, involving a manager and a group of technicians. The manager is responsible for receiving articles, and assigning an article to be repaired by a technician. All technicians have similar skills for repairing articles, and each one is responsible to repair one article at a time, independent of the other technicians. When a technician finishes repairing his assignment, he notifies the manager; the manager then assigns him a new article to be repaired, and so on. In general, repairing articles represents an irregular problem: some articles may present a simple fix and take a little amount of time, while others may require a more complex repair. Also, the effectiveness of this scheme relies on the fact that the number of articles that arrive to the centre can be substantially larger than the number of technicians available.

A Programming Example

Consider a real-time ultrasonic imaging system [GSVOM97], designed to acquire, process and display a tomographic image. Data is acquired based on the reflection of an ultrasonic signal that excites an array of 56 ceramic sensors. Data is amplified and digitalised to form a black and white image of 56´ 256 pixels, each one represented by a byte. An interpolation program is required to process the image, enlarging it to make it clearer to the observer. The image is displayed on a standard resolution monitor (640´ 480 pixels) in real-time, this is, at least 25 frames per second. In accordance with these requirements, an interpolation by a factor 3 between columns was chosen, enlarging the information of each image three times. A calculation shows the volume of data to be processes per second: each frame is represented as 168´ 256´ 1 bytes, and using 25 frames per second, makes a total of 1.075200 Mbytes per second. Using a manager-worker system for the cubic interpolation, the image is received a stream of pixels by the manager, which assigns to each worker a couple of pixels. Each worker uses each couple of pixels as input data, and calculates the cubic interpolation between them, producing other four interpolated pixels. As the number of workers is less than the total number of pixels, each worker requests for more work to the manager as soon as it finishes its process, and so on.

Problem

A computation is required where independent computations are performed, perhaps repeatedly, on all elements of some ordered data. Each computation can be performed completely, independent of the activity of other elements. Data is distributed among components without a specific order. However, an important feature is to preserve the order of data. Consider, for example, an imaging problem, where the image can be divided into smaller sub-images, and the computation on each sub-image does not require the exchange of messages between components. This can be carried out until completion, and then the partial results gathered. The overall affect is to apply the computation to the whole image. If this computation is carried out serially, it should be executed as a sequence of serial jobs, applying the same computation to each sub-image one after another. Generally, performance as execution time is the feature of interest.

Forces

Using the previous problem description and other elements of parallel design, such as granularity and load balance [Fos94, CT92], the following forces are found:

Preserve the order of data. However, the specific order of data distribution and operation among processing elements is not important

The same computation can be performed independently and simultaneously on different pieces of data.

Data pieces may exhibit different sizes.

Changes in the number of processing elements should be reflected by the execution time.

Improvement in performance is achieved when execution time decreases.

"""

**Valgoppgave B)**  
**Værvarsel**

**Mål:**

I denne oppgaven skal du lage et enkelt værvarsel for Bryggen i Bergen. Du kan bruke det rammeverket du ønsker (eller ikke bruke et rammeverk).

Applikasjonen skal hente værdata fra et åpent API.

**API:**

Vi skal bruke OpenWeatherMap API. For å hente nåværende værvarsel fra Bryggen i Bergen, kan du gjøre en GET-request mot dette endepunktet:

https://api.open-meteo.com/v1/forecast?latitude=60.397076&longitude=5.324383&current=temperature\_2m,is\_day,weather\_code&forecast\_days=1

Endepunktet gir info om nåværende værtype, nåværende temperatur og om det er dag eller natt. Du trenger ikke hente flere data enn dette, da det er bare disse 3 du skal vise.

**Krav:**

Du skal vise nåværende temperatur og et bilde for å illustrere nåværende værtype og om det er dag eller natt.

Dokumentasjon for de ulike værtypene finner du her: https://open-meteo.com/en/docs#weathervariables

Bilder for de ulike værtypene kan du hente herfra: https://gist.github.com/stellasphere/9490c195ed2b53c707087c8c2db4ec0c

Applikasjonen skal være universelt utformet.

**Innlevering**

Send en lenke til et Github repo med løsningen