# Coding Exercise No 1

## APIs design

### Sync CRUD tasks / tasks list API

Available both on client and server side

When online: server CRUD API used => create server event log

When offline: client CRUD API used => create/buffer client local event log

Push everything once online through the async api

Propose only :

* Create tasks / tasks list
* Delete tasks / tasks list
* Read tasks / tasks list
* Updates task (merging issue too complex to manage in 3 days):
  + Replace/update or append description to avoid merging issues (only incremental change considered in changeset)
  + replace/update title
  + replace/update priority
* Updates tasks list:
  + replace title

Replace only available to the user who create ? (who is the owner)

No permission for now (only owner auto defined on creation)

### Async API

Receive change update from server/client when online

Init/refresh task list from server (only the ones that the users subscribe => user filter)

## Data model

## Event based synchronization design

## Client development

## AWS deployment model

Alternative

Distributed advanced caching (that works offline)

Check AWS lambda flask conversion (but does not make sense for a client to run on AWS, better to be docker)

Celery for event queueing

Dynamodb for persistence