.NET 5/core

web Api

TDD

X unit

NOT
======
Database

front end

" Uncle" Bob — Leave tech decisions
until the last moment

Prag Prog — Tracer bullets

O k X Y Features
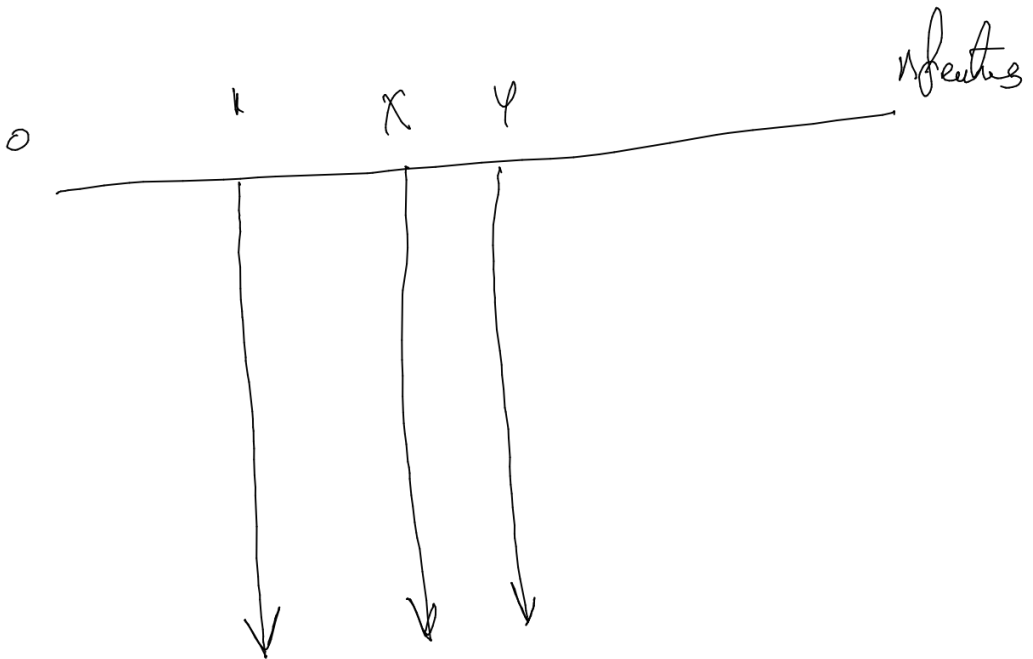
- NET 5 or Core
  - web Api
  - no Auth
  - Open Api / swagger
  - Tests
  - Rest ful (GET / POST / PUT / etc)

for next time. Vs code remote
                      sessions

Controller
  ↓
Services
  ↓
Data Store

App Run

↓

$\left(\text{App settings . json}\right)$

↓

environment name = Development

↓

$\left(\text{App settings . Development . json}\right)$

↓

Do i have user Secrets?
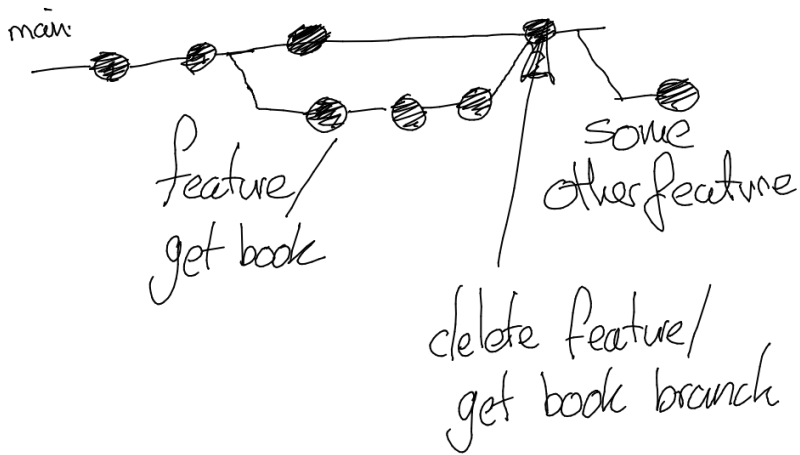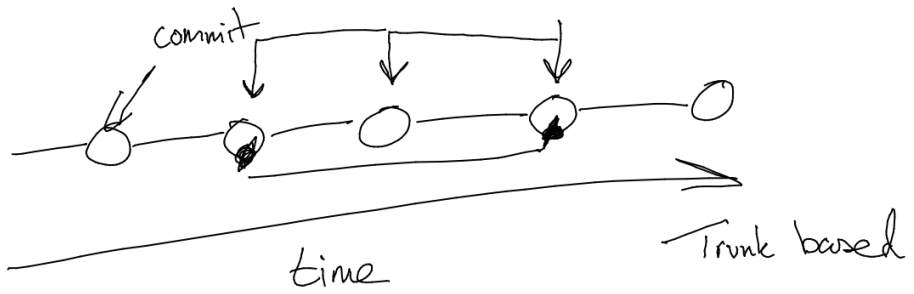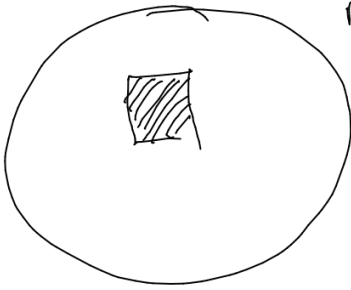
↓

{guid3 . json

↓

loads those

↓

Cli appsettings.

off stream

- ✓ - fix rider (upgrade + add correct
      licence details)
- - document what i did
- ✓ - create a readme
- ✓ - git commit && git push.

commit

time

Trunk based

main:

feature/
get book

delete feature/
get book branch

some
other feature

Mock dependency

 ← dependency

Moq proxy

proxy when
i call get on
the dependency, actually
do the following

Red, Green, Refactor

$$\left(\text{Test} + \text{Commit} \right) \text{ or Revert}$$

Red = write a test or make a
change THEN run the tests

Green = fix any broken test or
compilation issues

Refactor = revisit affected code and
refactor where neccessar

Feb 5th 2021

Paged results

- Subset of results
- Page number
- total results
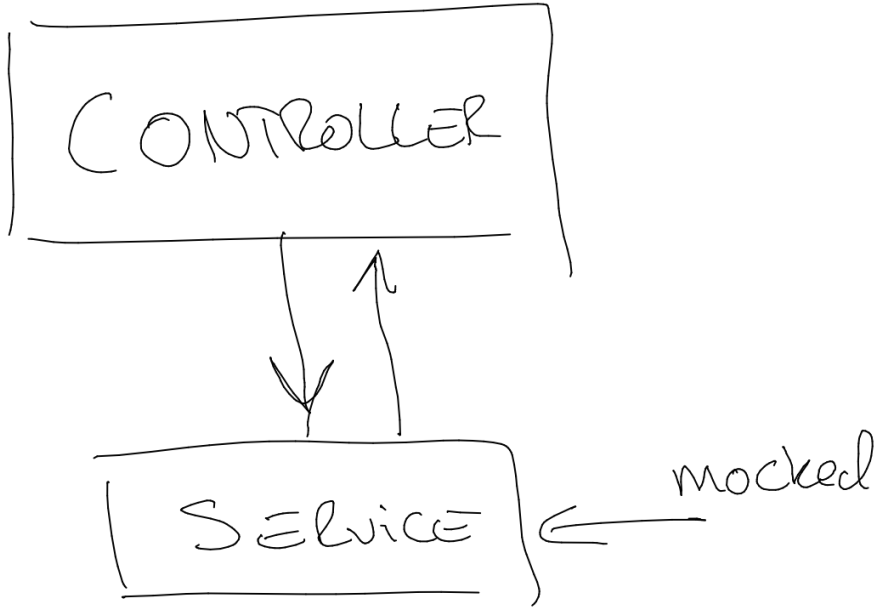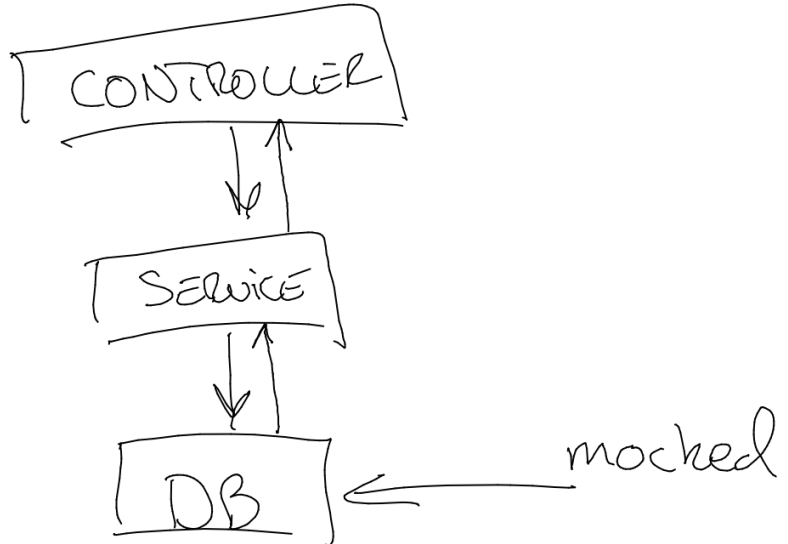- Supported results per page
  (10, 25, 50, 100, etc.)

Plain

Old

C#

Object

Currently

CONTROLLER

SERVICE ← mocked

future

CONTROLLER

SERVICE

DB ← mocked

Paged Service

- get a page of records based on the page number requested
- get the number of records per page

- not going to use other properties for now
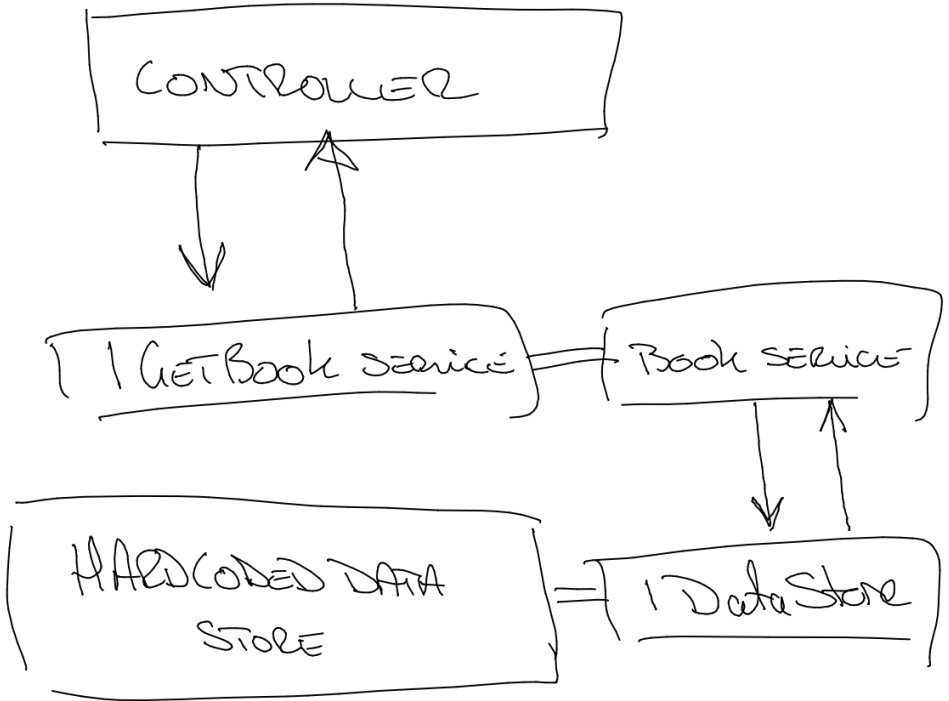
| Skip + Take
| ↑
| Linq operations

Page 1 with 5 results
   Skip nothing, take 5

Page 2 with 5 results
   Skip 5, take 5

Page 3 with 5 results
   Skip 10, take 5

New Design

```
┌─────────────────────────┐
│   CONTROLLER            │
└─────────────────────────┘
     │              ↑
     ↓              │
┌──────────────────────┐    ┌──────────────────┐
│ I GETBook service    │────│ Book service     │
└──────────────────────┘    └──────────────────┘
                                   │       ↑
                                   ↓       │
┌──────────────────────┐    ┌──────────────────┐
│ HARDCODED DATA       │────│ I DataStore      │
│      STORE           │    └──────────────────┘
└──────────────────────┘
```

Issue #3 will be looking
at replacing the HardCoded DataStore
with some kind of Database