SalesmanUI klasinn:

* Sá klasi á private breytu sem er tilvik af videoservice klasanum
* Hann á líka main\_menu fall sem birtir aðalvalmyndina og kallar á þær aðgerðir sem valdar eru. T.d. ef valin er add video þá er kallið á videoservice.add video (falll sem bætir við spólu í safnið)
* Ef valið er að birta alla lausa bíla þá er kallað á videoservice sem birtir alla bíla(gæti hugsanlega verið service sem sendir til baka lista sem maður sendir inn í view fall sem birtir alla bíla
* Ef birta þarf undirvalmynd þá er kallað á það fall
* SalesmanUI klasinn á allar þessar valmyndir sem föll hjá sér

VideoService layer-ið

* Service layer-ið á eintak af repo layer-inu
* Þarna er eitthvað „is video valid fall“, til að check-a hvort video-ið sé valid sem verið var að bæta við
* Annað fall sem bætir við video spólu (ef hún er valid þ.e.a.s)
* Önnur föll sem eyða út video spólu
* Get all available
* Get rentals o.s.frv.
* Öll þessi lógík business föll
* Add video fallið hjá Boogie gerir ekkert annað en að kalla á isvalid fallið og kalla svo á repo.addvideo fallið ef svo er
* Ekki viss hvort ég væri með sér CarService, svo RentalService, Svo CustomerService o.s.frv. sennilega betra en einn stór service klasi

Repository layer-ið

* Talar við gagnagrunninn, add, delete, update föll á allar töflurnar, hendir gögnunum í grunninn
* Einnig öll get föll, ná í allt í heilu lagi úr grunninum held ég og miðju layerið vinnur svo úr því til að birta í efsta layer-inu
* Þarf að læra að eyða og update-a í csv skránum
* a+ þýðir að append og plúsinn þýðir að skráin verður búin til ef hún er ekki til

Klasinn Video:

* Boogie stillir þessu þannig upp að video klasinn á get\_title fall og get\_length fall o.s.frv. svo er náð í þessar breytur og þeim hent inn í skránna

ATH hvernig boogie notar main.py til að keyra forritið í gang

(24:43 í fyrirlestri 2) cópíera það bara, láta ui klasann sjá um loop-una sem birtir main menu-ið

Hver klasi á sína eigin skrá! Sama hversu lítill hann er

Búa til möppu fyrir hvert layer

* Data (CSV skjöl)
* Models (POCO klasarnir)
* Repositories (Repo klasarnir sem tala við data layer-ið)
* Services (Sem vinna úr draslinu til að birta og validate-a draslið til að skrifa í grunninn
* Ui (Interface dótið)
* Svo er main.py utan folders

Punktar frá mér:

* Næst á dagskrá, að taka einn bíl frá UI layer í gegnum business layer niður í repo layer og skrifa hann í grunninn
* Hann virðist alltaf nota from X import X (þyrfti kannski að nota þann stíl (sjá 31:24 í fyrirlestri 2)
* Error handling og validation eitthvað sem ég þarf að huga að í gegnum allt

Spurningar: