Bre zh C@mp
Mix de technologies

CapitalMatch
Singapore

Haskell: du sol au plafond

Building a startup using Haskell

Arnaud Bailly
arnaud@capital-match.com
http://www.capital-match.com
@abailly

Agenda

CapitalMatch
Singapore

- Who/Why/What
- How we use Haskell
- The Good, the Bad and the Ugly
- Future work

This is not a Monad tutorial….

CapitalMatch
Singapore

Who? What? Why?

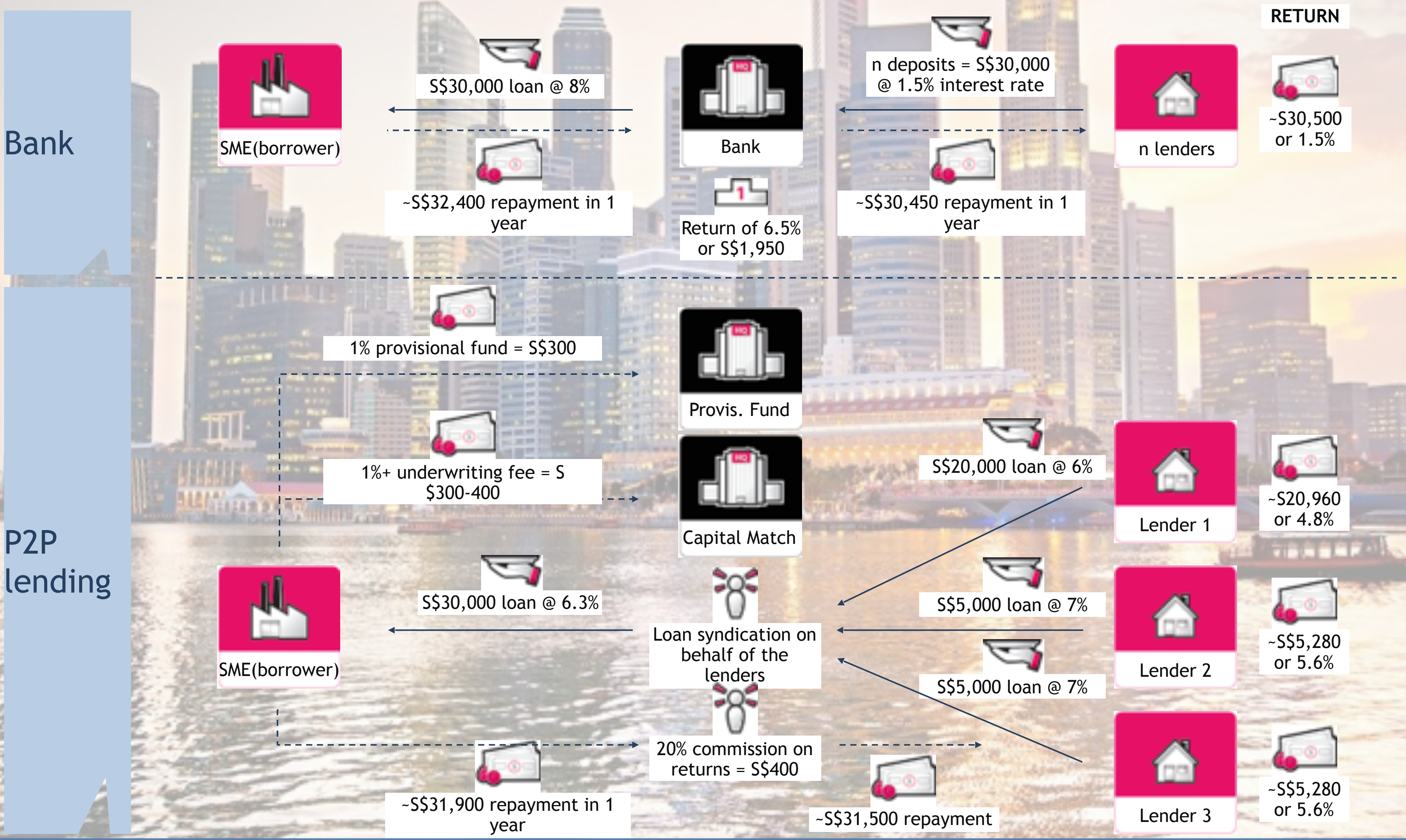
Who?

CapitalMatch
Singapore

- Pawel Kuznicki
  - CEO, ex-Rocket, ex-McKinsey, ex-Zalora
- Kevin Lim
  - CFO, ex-JPMorgan, ex-SCB
- **Arnaud Bailly**
  - 20+ years experience, mostly Java for food, discovered Haskell in 2001
- Willem van den Ende
  - 20+ years experience, XP since the beginning, Smalltalk/Clojure/.Net…

What? Marketplace Lending!

CapitalMatch Singapore


RETURN
Bank
SME(borrower)
S$30,000 loan @ 8%
~S$32,400 repayment in 1 year
Bank
Return of 6.5% or S$1,950
n deposits = S$30,000 @ 1.5% interest rate
~S$30,450 repayment in 1 year
n lenders
~S30,500 or 1.5%
P2P lending
1% provisional fund = S$300
Provis. Fund
1%+ underwriting fee = S $300-400
Capital Match
SME(borrower)
S$30,000 loan @ 6.3%
Loan syndication on behalf of the lenders
20% commission on returns = S$400
~S$31,900 repayment in 1 year
~S$31,500 repayment
S$20,000 loan @ 6%
Lender 1
~S20,960 or 4.8%
S$5,000 loan @ 7%
Lender 2
~S$5,280 or 5.6%
S$5,000 loan @ 7%
Lender 3
~S$5,280 or 5.6%

Why Haskell?

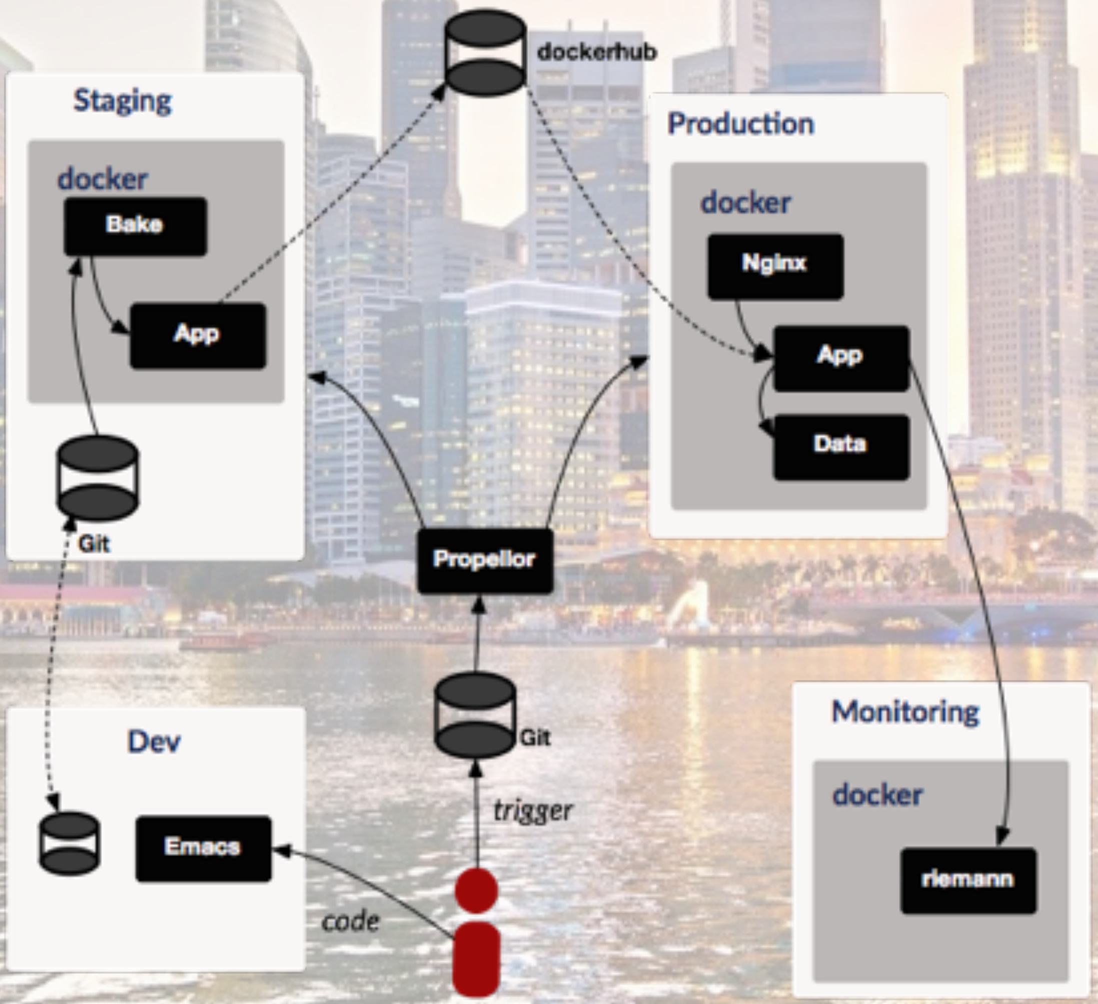
CapitalMatch Singapore

- Pawel had very good experience working with Haskell developers at previous job
- He posted job offer on http://functionaljobs.com
- I wanted to do some *real stuff* in Haskell
- I had good experience working for people in Singapore
- It seemed fun!

Let's do it!

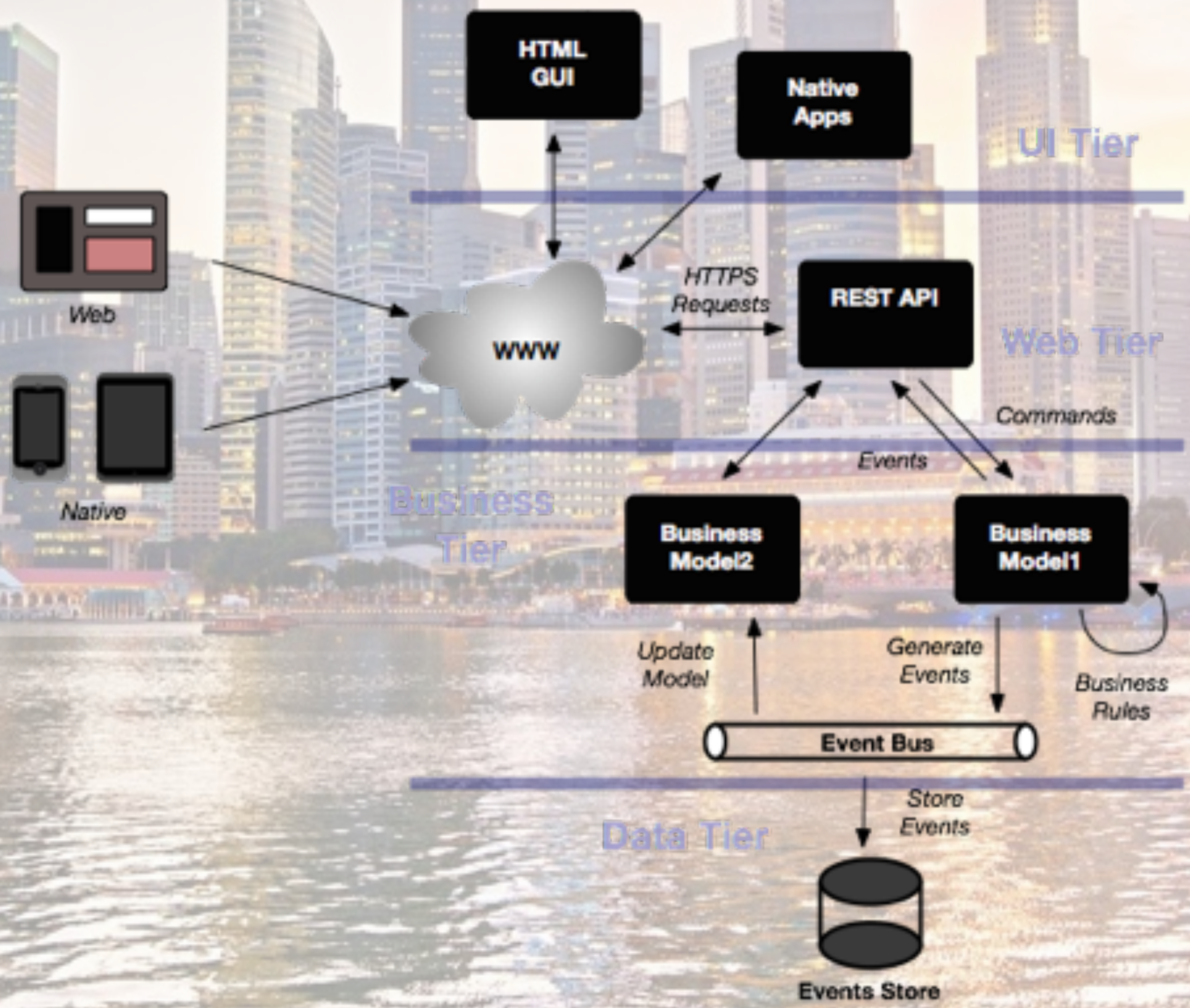
System Overview

CapitalMatch
Singapore



Application Architecture

CapitalMatch
Singapore



CapitalMatch
Singapore

How we use Haskell?

For (Nearly) Everything!

CapitalMatch Singapore

- Dev. Env ⟶ **ghc-mod, stylish-haskell**
- Web Backend ⟶ **Scotty, Blaze**
- Database ⟶ Custom Event Sourcing
- Unit/Integration Testing ⟶ **HSpec, QuickCheck**
- End-to-End Testing ⟶ **hs-webdriver**
- Build ⟶ **Cabal, Shake**
- CI Server ⟶ **Bake**
- Configuration Management ⟶ **Propellor**

For the rest…

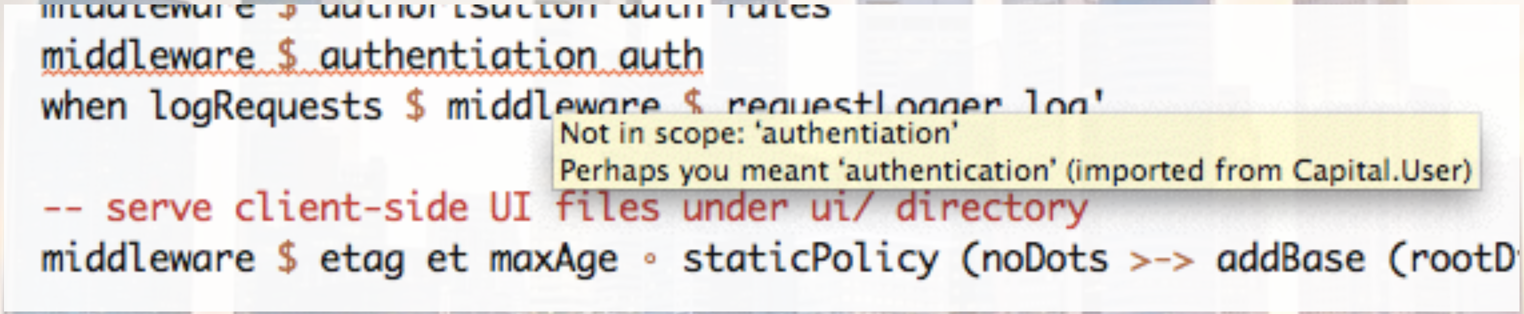
CapitalMatch
Singapore

- Web Front-end ⟶ **Om/Clojurescript**
- Version Control ⟶ **Git** (what else?)
- Packaging & Deployment ⟶ **Docker** (because we can)
- Infrastructure ⟶ DigitalOcean / S3
- Monitoring ⟶ **Riemann, collectd** (WIP)

Emacs Dev. Envt.

CapitalMatch
Singapore

Compilation FlyCheck

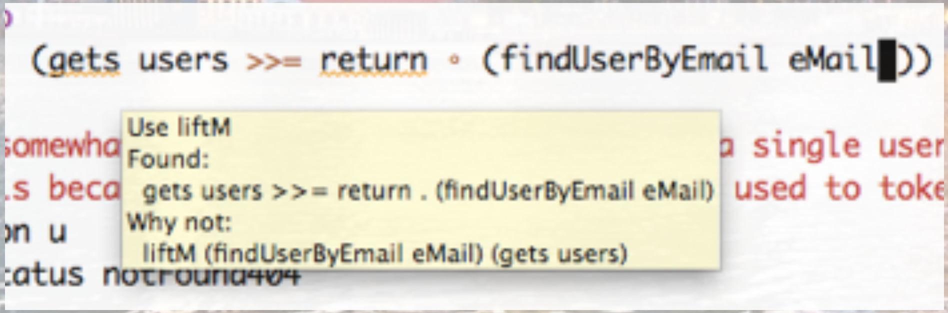


```
middleware $ authentiation auth
when logRequests $ middleware $ requestLogger log'
```

Not in scope: 'authentiation'
Perhaps you meant 'authentication' (imported from Capital.User)

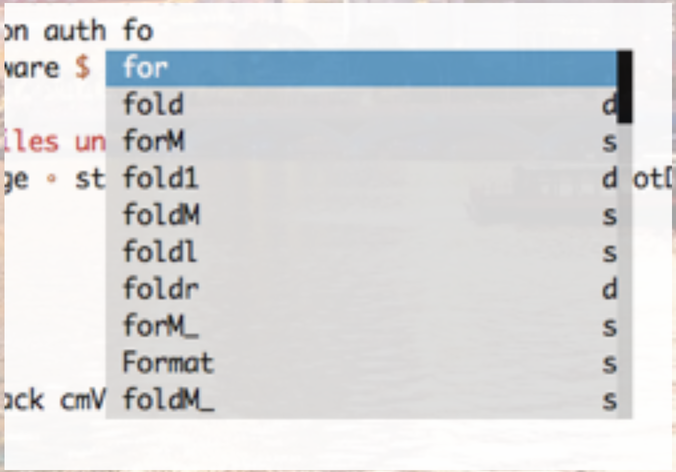
```
-- serve client-side UI files under ui/ directory
middleware $ etag et maxAge ∘ staticPolicy (noDots >-> addBase (rootD
```

HLint



```
(gets users >>= return ∘ (findUserByEmail eMail))
```

Use liftM
Found:
  gets users >>= return . (findUserByEmail eMail)
Why not:
  liftM (findUserByEmail eMail) (gets users)



```
on auth fo
ware $  for
        fold     d
        forM     s
        fold1    d
        foldM    s
        foldl    s
        foldr    d
        forM_    s
        Format   s
        foldM_   s
```

Autocompletion

Scotty: REST Endpoints

CapitalMatch
Singapore

```
post "/api/users" $ do
  (u :: RegisteringUser) ← jsonData
  e ← inWeb $ registerUser u
  case e of
   RegisteredUser _              → status statusCreated
   InvalidUserRegistration t     → do
     status statusBadRequest
     text t
   UserAlreadyRegistered (EMail m)  → do
     status statusBadRequest
     text $ "you cannot register with email " <> m
   _ → do
     status statusBadRequest
     text $ T.pack $ show e
```

Scotty: Middleware

CapitalMatch
Singapore

```haskell
authorisation :: (AuthState s)
                 ⇒ TVar s
                 → Rule (s → Request → Authorisation)
                 → Middleware
authorisation ref rule app req sendResponse = do
    st ← atomically $ readTVar ref
    case authorise rule st req of
     Deny   → sendResponse $ responseFile status404 [] "404.html" Nothing
     _      → app req sendResponse
```

```haskell
middleware $ requestId
middleware $ authorisation auth rules
middleware $ authentication auth
when logRequests $ middleware $ requestLogger log'

-- serve client-side UI files under ui/ directory
middleware $ etag et maxAge ∘ staticPolicy (noDots >-> addBase (rootDir </> "ui"))
```

Blaze: Template HTML

CapitalMatch
Singapore

```haskell
borrowerProfile :: T.Text → H.Html
borrowerProfile displayName = do
  menu displayName
  H.div ! A.id "borrower-content" ! class_ "pure-g" $ do
    H.div ! A.id "register" ! class_ "pure-u-7-8 pure-form pure-form-aligned" $ do
      H.div ! A.id "notification-view" $ mempty
      H.div ! A.id "register-view" $ mempty
  cmFooter
  feedbackForm
```

HSpec: Integration Tests

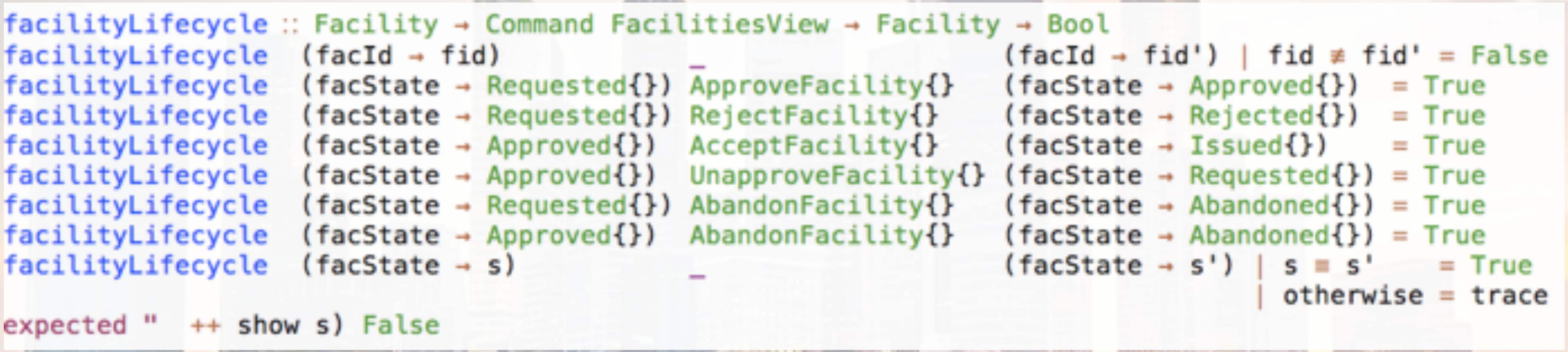
CapitalMatch
Singapore

```
it_ "on GET with id retrieves registered investor" $ do
  registerInvestor_ investor

  r ← getJSON "/api/investors/foo@somewhere.com" :: ClientT IO (Response Investor)

  r ^. responseBody `shouldBe` investor
```

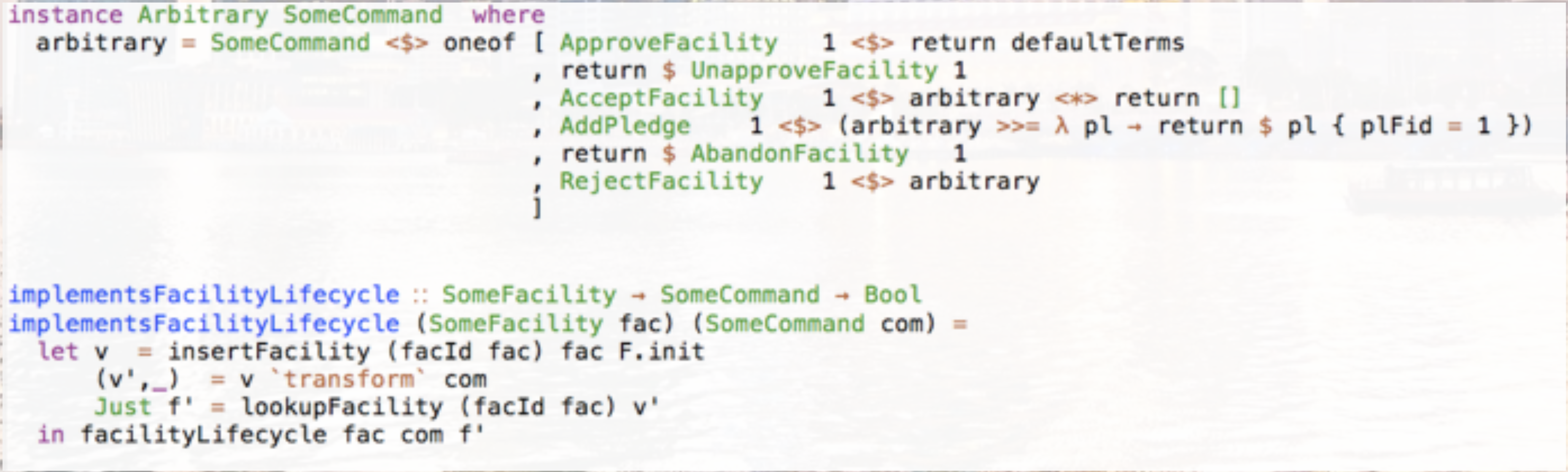
```
--         path                                        borrower foo  investor bar  admin      visitor
accessTo (Get "/admin/investors")                     isNotAllowed  isNotAllowed  isAllowed  isNotAllowed
accessTo (Get "/admin/borrowers")                     isNotAllowed  isNotAllowed  isAllowed  isNotAllowed
accessTo (Get "/admin/facilities")                    isNotAllowed  isNotAllowed  isAllowed  isNotAllowed
```

QuickCheck

CapitalMatch
Singapore



```
facilityLifecycle :: Facility → Command FacilitiesView → Facility → Bool
facilityLifecycle  (facId → fid)             _                    (facId → fid') | fid ≠ fid' = False
facilityLifecycle  (facState → Requested{}) ApproveFacility{}    (facState → Approved{})   = True
facilityLifecycle  (facState → Requested{}) RejectFacility{}     (facState → Rejected{})   = True
facilityLifecycle  (facState → Approved{})  AcceptFacility{}     (facState → Issued{})     = True
facilityLifecycle  (facState → Approved{})  UnapproveFacility{}  (facState → Requested{})  = True
facilityLifecycle  (facState → Requested{}) AbandonFacility{}    (facState → Abandoned{})  = True
facilityLifecycle  (facState → Approved{})  AbandonFacility{}    (facState → Abandoned{})  = True
facilityLifecycle  (facState → s)            _                    (facState → s') | s ≡ s'     = True
                                                                                  | otherwise = trace
expected "  ++ show s) False
```



```
instance Arbitrary SomeCommand  where
  arbitrary = SomeCommand <$> oneof [ ApproveFacility   1 <$> return defaultTerms
                                    , return $ UnapproveFacility 1
                                    , AcceptFacility    1 <$> arbitrary <*> return []
                                    , AddPledge     1 <$> (arbitrary >>= λ pl → return $ pl { plFid = 1 })
                                    , return $ AbandonFacility    1
                                    , RejectFacility    1 <$> arbitrary
                                    ]


implementsFacilityLifecycle :: SomeFacility → SomeCommand → Bool
implementsFacilityLifecycle (SomeFacility fac) (SomeCommand com) =
  let v   = insertFacility (facId fac) fac F.init
      (v',_)  = v `transform` com
      Just f' = lookupFacility (facId fac) v'
  in facilityLifecycle fac com f'
```

hs-webdriver: ETE Testing

CapitalMatch
Singapore

```
it "Borrower applies for loan which is funded by investor" $ runWD $ do
  createAccountSpecFor appServer willem userPassword "Borrower"
  createAccountSpecFor appServer arnaud userPassword "Investor"
  createAccountSpecFor appServer pawel  userPassword "Investor"

  userLogsIn appServer "admin@capital-match.com" "secret"
  adminApprovesBorrower appServer willem
  adminApprovesInvestors appServer [ arnaud ]
  adminIncreaseCashBalance appServer arnaud "90000" "90,000"
  userLogsOut

  borrowerAppliesForALoan appServer willem userPassword "Loan" "100000" "9" "Interest and Principal"

  userLogsIn appServer "admin@capital-match.com" "secret"
  adminApprovesLoan appServer "1" "2.5"
  userLogsOut

  investorFundsLoan appServer arnaud userPassword "1" "90000"

  investorFundsLoanOverAvailable appServer arnaud userPassword "1" "9000"

  borrowerAcceptsFullyFundedLoan appServer willem userPassword "1"

  investorSeesOutstandingLoan appServer arnaud userPassword "1"
  investorSeesAcceptedLoan appServer pawel userPassword "1"
```

Shake: Better make

CapitalMatch
Singapore

```
"images/app.uuid" *> λuuidFile → do
    recursiveNeed "ui" ["//*.cljs","//*.clj","//*.html"]
    recursiveNeed "src" ["//*.hs"]
    recursiveNeed "test" ["//*.hs"]
    recursiveNeed "end-to-end-test" ["//*.hs"]
    recursiveNeed "main" ["//*.hs"]
    need ["images/deps.uuid", "Dockerfile", "ui/project.clj", "404.html"]
    buildAppImage uuidFile organisation
```

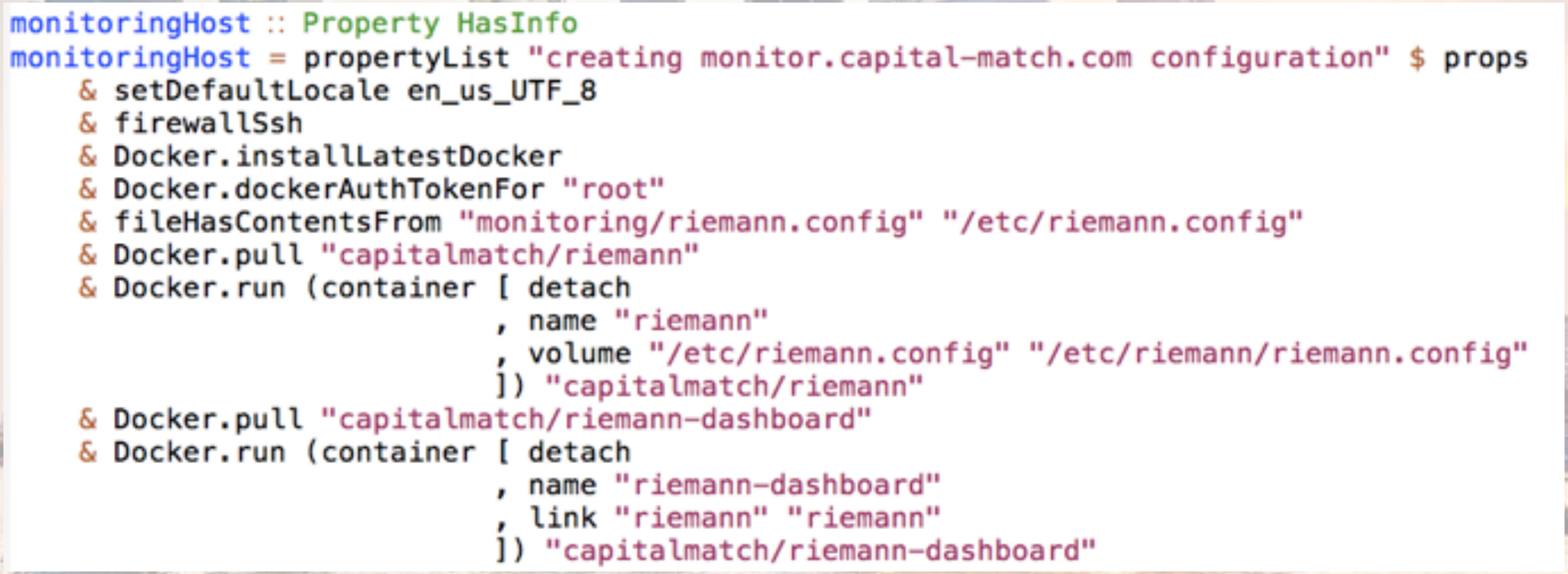
Bake: CI

CapitalMatch
Singapore

```
allTests :: [Action]
allTests = [Compile, Deploy]

execute :: Action → TestInfo Action
execute Compile = run $ do
  opt ← addPath ["."] []
  ∅ ← cmd opt "./build.sh"
  sleep 1
  incrementalDone
execute Deploy  = require [Compile] $ run $ do
  patch ← readFile patchFile
  -- we make sure that tag is not set to another image
  ∅ ← cmd $ "docker tag -f " ++ latest imageName ++ " " ++ tagged imageName patch
  ∅ ← cmd $ "docker push " ++ tagged imageName patch
  incrementalDone
    where
      tagged name patch = name ++ ":" ++ patch
      latest name       = tagged name "latest"
```

Propellor: Config Mgt.

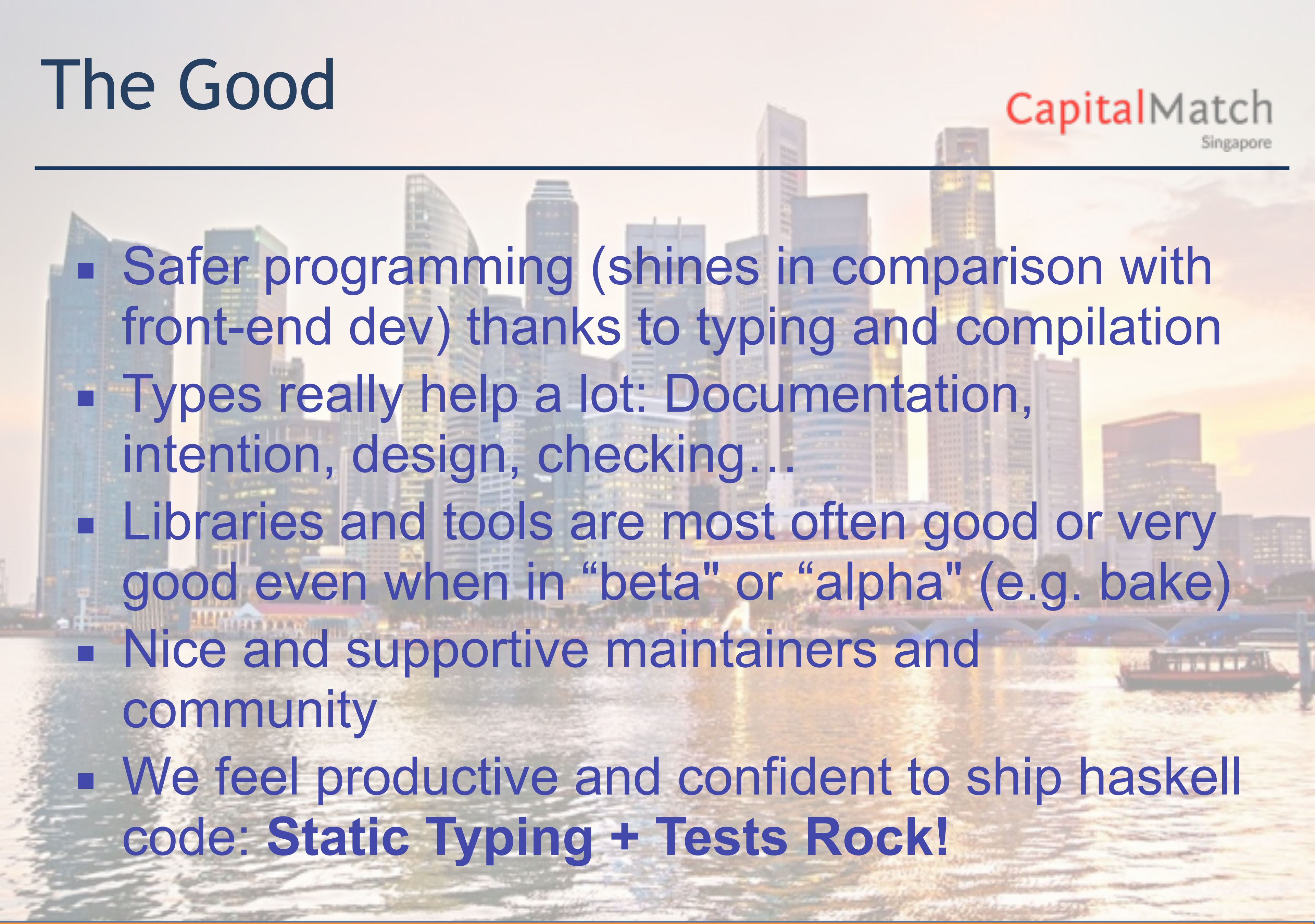
CapitalMatch
Singapore



```haskell
monitoringHost :: Property HasInfo
monitoringHost = propertyList "creating monitor.capital-match.com configuration" $ props
    & setDefaultLocale en_us_UTF_8
    & firewallSsh
    & Docker.installLatestDocker
    & Docker.dockerAuthTokenFor "root"
    & fileHasContentsFrom "monitoring/riemann.config" "/etc/riemann.config"
    & Docker.pull "capitalmatch/riemann"
    & Docker.run (container [ detach
                            , name "riemann"
                            , volume "/etc/riemann.config" "/etc/riemann/riemann.config"
                            ]) "capitalmatch/riemann"
    & Docker.pull "capitalmatch/riemann-dashboard"
    & Docker.run (container [ detach
                            , name "riemann-dashboard"
                            , link "riemann" "riemann"
                            ]) "capitalmatch/riemann-dashboard"
```

CapitalMatch
Singapore

The Good, the Bad and the Ugly



The Good

CapitalMatch
Singapore

- Safer programming (shines in comparison with front-end dev) thanks to typing and compilation
- Types really help a lot: Documentation, intention, design, checking…
- Libraries and tools are most often good or very good even when in “beta" or “alpha" (e.g. bake)
- Nice and supportive maintainers and community
- We feel productive and confident to ship haskell code: **Static Typing + Tests Rock!**

The Good (contd.)

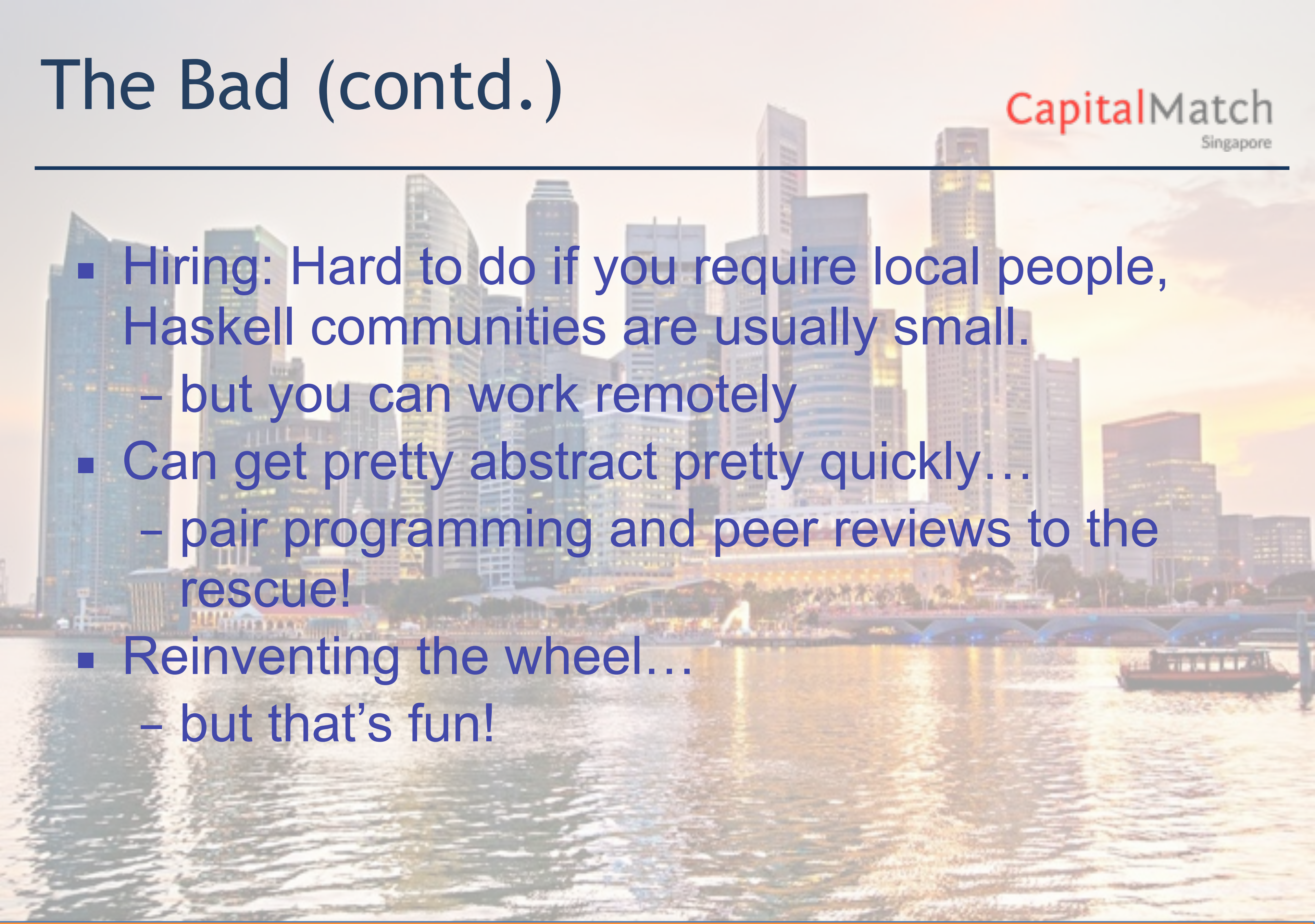
CapitalMatch
Singapore

- Refactoring is easier: Change a type and fix compiler’s errors
- Good for hiring: Haskell attracts “interesting" people
- Very easy to replace clunky scripts with typesafe and compiled DSL

The Bad

CapitalMatch Singapore

- Cabal
  - but does its job, no binary packages possible and there is Shake for funky stuff
  - it is improving (e.g. Stackage)
- Dev. Env. is still not on par with Eclipse/IntelliJ/VS
  - but FPComplete and others are making progress fast and tooling improves
- Compilation typing errors
  - but you get accustomed to it once your code base is stable



The Bad (contd.)

CapitalMatch
Singapore

- Hiring: Hard to do if you require local people, Haskell communities are usually small.
  - but you can work remotely
- Can get pretty abstract pretty quickly…
  - pair programming and peer reviews to the rescue!
- Reinventing the wheel…
  - but that's fun!

The Ugly

CapitalMatch Singapore

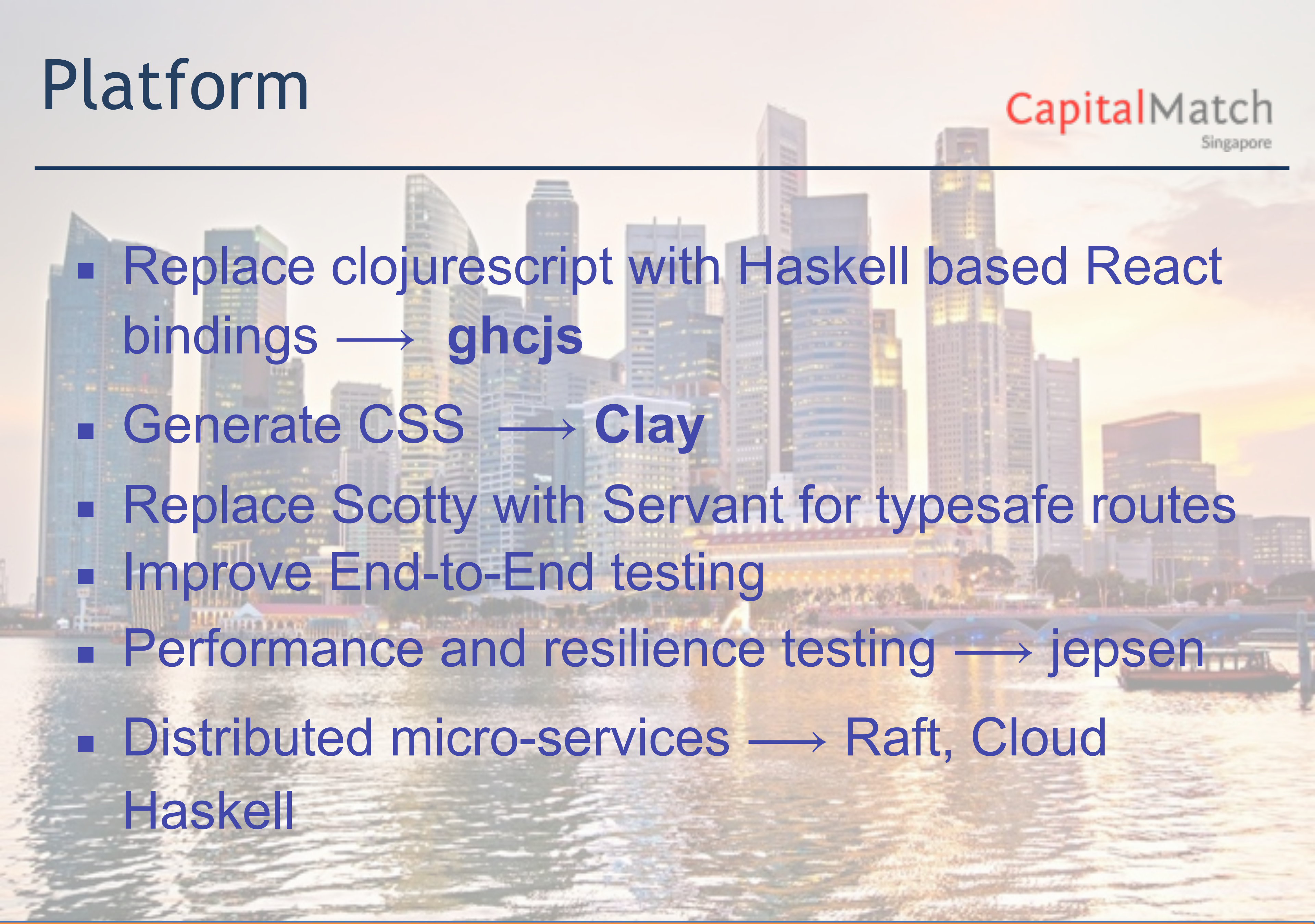
- String vs. Data.Text vs. Data.Text.Lazy vs.
  - $\longrightarrow$ Oh My! Haskell is Old!
- Runtime error reporting
  - $\longrightarrow$ No Stack Traces!
- Aeson deserialization errors
  - $\longrightarrow$ Cryptic No Parse
- Conflicting GHC versions/Libs requirements
  - $\longrightarrow$ Cabal Hell
- Laziness can bite you

CapitalMatch
Singapore

Future Work

Platform

CapitalMatch
Singapore



- Replace clojurescript with Haskell based React bindings $\longrightarrow$ **ghcjs**
- Generate CSS $\longrightarrow$ **Clay**
- Replace Scotty with Servant for typesafe routes
- Improve End-to-End testing
- Performance and resilience testing $\longrightarrow$ jepsen
- Distributed micro-services $\longrightarrow$ Raft, Cloud Haskell

Tooling

CapitalMatch Singapore

- Improve CI:
  - handle multiple branches, parallel builds, better reporting
- Improve configuration management
  - replace scripts w/ Haskell code for provisioning, better propellor configs
- Better containers builds and orchestration

Credit Risk Analysis

CapitalMatch
Singapore

- Build a database and inference engine to better automate Credit Risk Assessment process
  - NLP, ML, Big Data…

CapitalMatch Singapore

We are hiring!

\

http://functionaljobs.com/jobs/8802-full-stack-software-engineer-at-capital-match

CapitalMatch
Singapore

Questions?