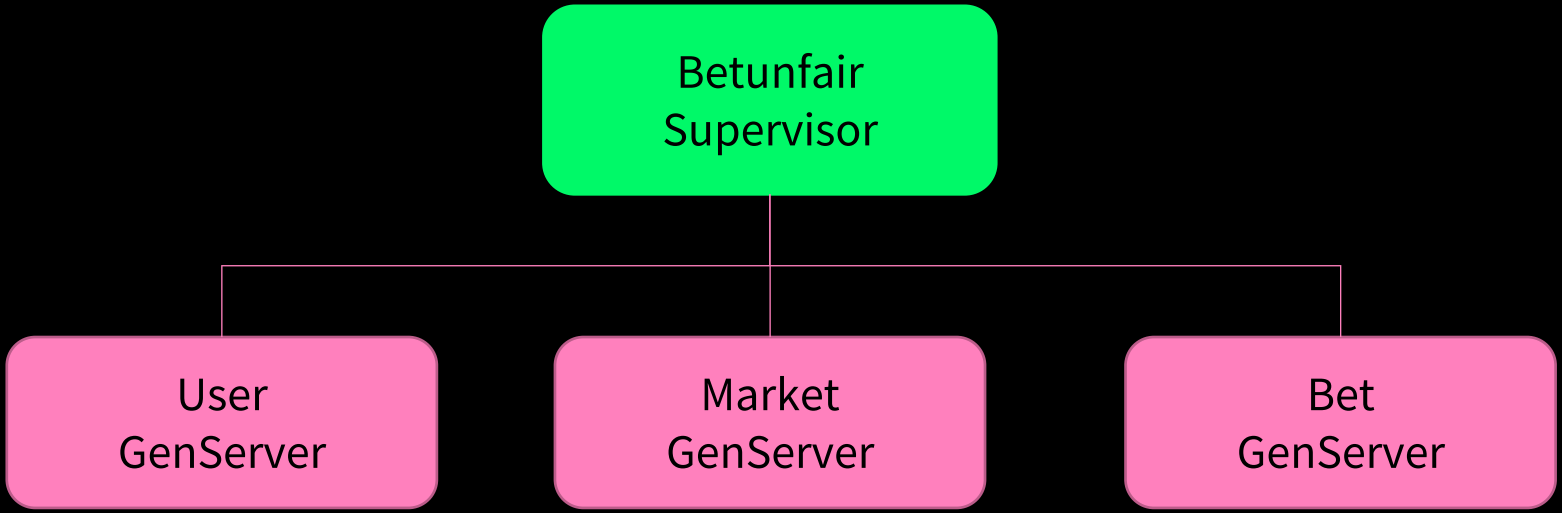
BETUNFAIR

MANUEL LOZANO RAMOS
ALONSO GARCÍA VELASCO
TRISTÁN VAQUERO POVEDANO

Design: modules

The project is break down into 4 modules:



Design: database

CubDB

- Key-value database
- ACID transactions
- Concurrency control

Design: database

• UsersDB	• Example
• Id	• “u1”
• User identifier	• “09317384J”
• User name	• “Federica García Lorca”
• User balance	• 3000

Design: database

- MarketsDB
  - Id
  - Name
  - Description
  - Status
  - Bets:
    - Back
    - Lay

- Example
  - “m1”
  - “El Clásico”
  - “Madrid vs Barcelona 7-0”
  - :active
  - Bets:
    - [%{“bb1”, :back, “m1”, “u1”, 2, 1000, 1000, [], :active}]
    - [%{“bb2”, :back, “m1”, “u2”, 2, 1000, 1000, [], :active}]

Design: database

BetsDB	Example
Bet_id	“bb1”
Bet_type	:back
Market_id	“m1”
User_id	“u1”
Odds	2
Original_stake	1000
Remaining_stake	1000
Matched_bets	[]
Status	:active

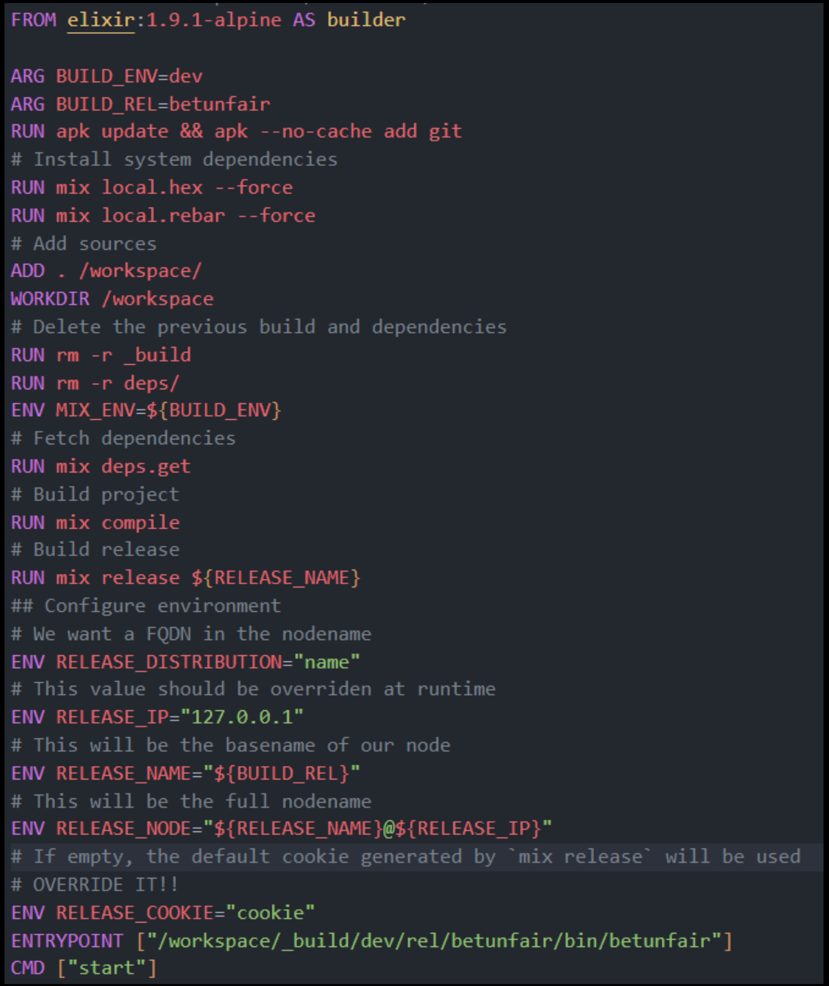
Scalability

Docker

Kubernetes

Docker Image

- Elixir as base image
- The image:
  - Copies and compiles the application
  - Installs dependencies
  - Set environment variables
  - Runs the application



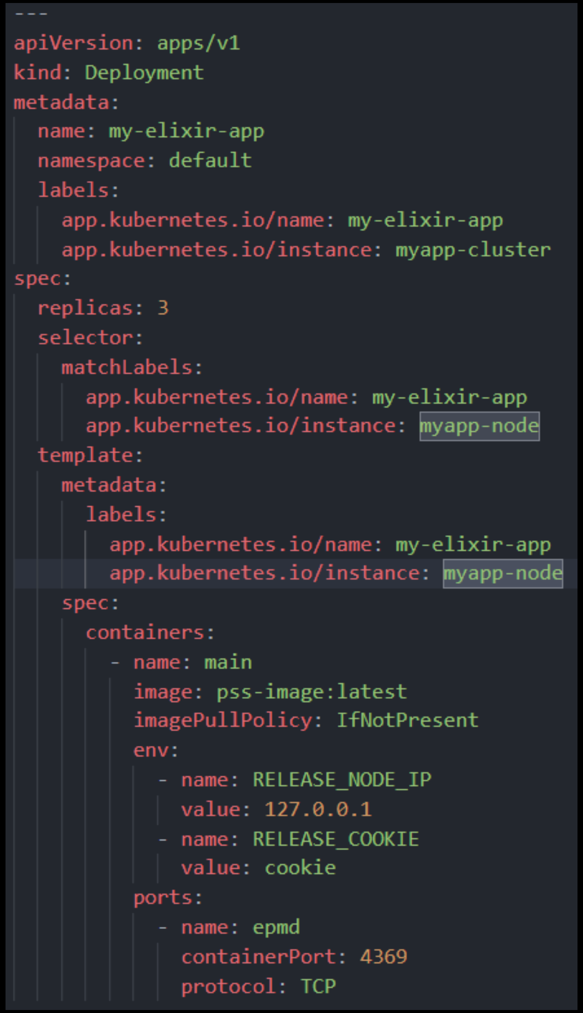
```
FROM elixir:1.9.1-alpine AS builder

ARG BUILD_ENV=dev
ARG BUILD_REL=betunfair
RUN apk update && apk --no-cache add git
# Install system dependencies
RUN mix local.hex --force
RUN mix local.rebar --force
# Add sources
ADD . /workspace/
WORKDIR /workspace
# Delete the previous build and dependencies
RUN rm -r _build
RUN rm -r deps/
ENV MIX_ENV=${BUILD_ENV}
# Fetch dependencies
RUN mix deps.get
# Build project
RUN mix compile
# Build release
RUN mix release ${RELEASE_NAME}
## Configure environment
# We want a FQDN in the nodename
ENV RELEASE_DISTRIBUTION="name"
# This value should be overriden at runtime
ENV RELEASE_IP="127.0.0.1"
# This will be the basename of our node
ENV RELEASE_NAME="${BUILD_REL}"
# This will be the full nodename
ENV RELEASE_NODE="${RELEASE_NAME}@${RELEASE_IP}"
# If empty, the default cookie generated by `mix release` will be used
# OVERRIDE IT!!
ENV RELEASE_COOKIE="cookie"
ENTRYPOINT ["/workspace/_build/dev/rel/betunfair/bin/betunfair"]
CMD ["start"]
```

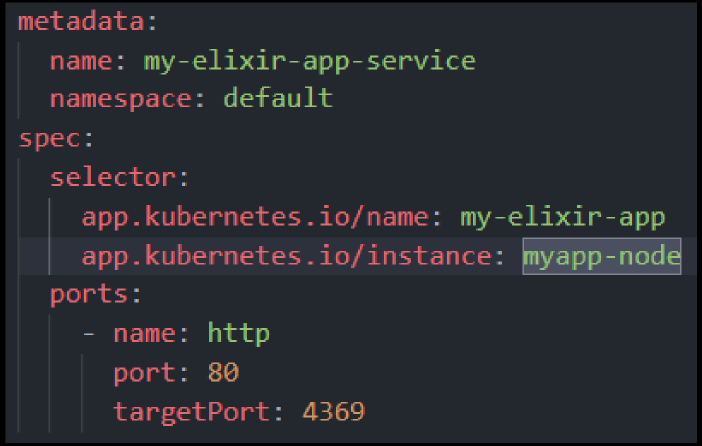
Kubernetes Deployment & Service

- Pod configuration:
  - Docker image
  - Environment variables
  - Exposed ports

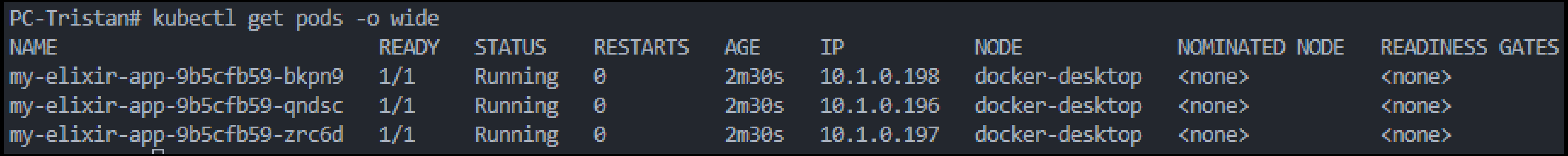
- Service:
  - Exposed port
  - Target port



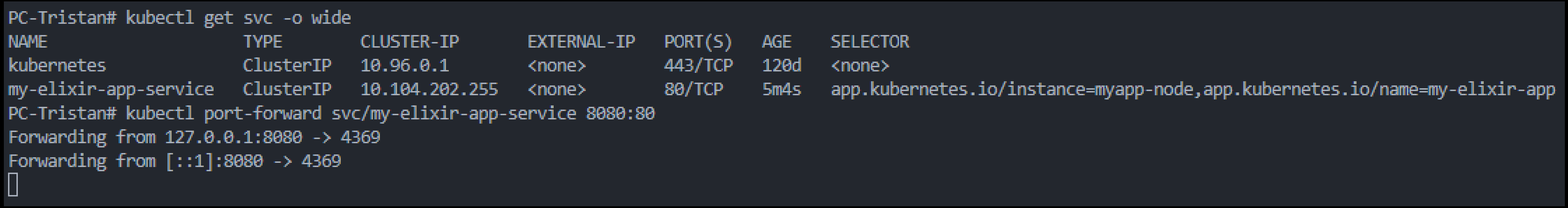
```yaml
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-elixir-app
  namespace: default
  labels:
    app.kubernetes.io/name: my-elixir-app
    app.kubernetes.io/instance: myapp-cluster
spec:
  replicas: 3
  selector:
    matchLabels:
      app.kubernetes.io/name: my-elixir-app
      app.kubernetes.io/instance: myapp-node
  template:
    metadata:
      labels:
        app.kubernetes.io/name: my-elixir-app
        app.kubernetes.io/instance: myapp-node
    spec:
      containers:
        - name: main
          image: pss-image:latest
          imagePullPolicy: IfNotPresent
          env:
            - name: RELEASE_NODE_IP
              value: 127.0.0.1
            - name: RELEASE_COOKIE
              value: cookie
          ports:
            - name: epmd
              containerPort: 4369
              protocol: TCP
```



```yaml
metadata:
  name: my-elixir-app-service
  namespace: default
spec:
  selector:
    app.kubernetes.io/name: my-elixir-app
    app.kubernetes.io/instance: myapp-node
  ports:
    - name: http
      port: 80
      targetPort: 4369
```



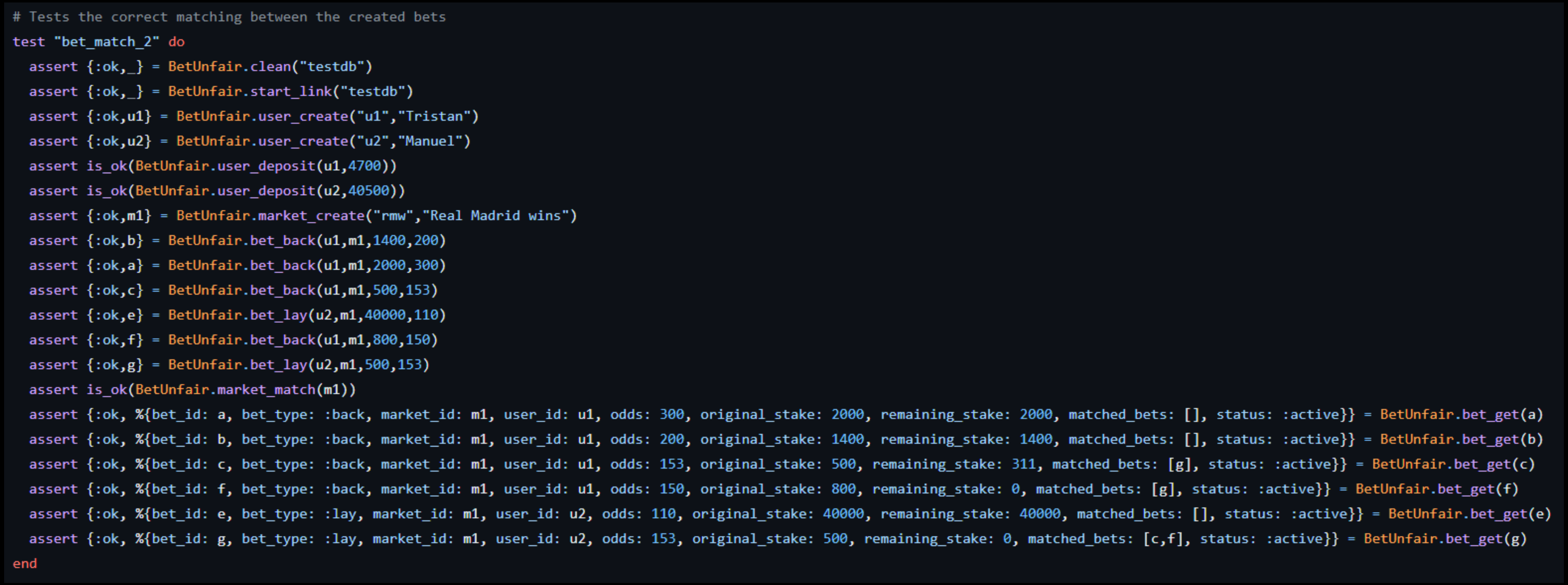
```
PC-Tristan# kubectl get pods -o wide
NAME                            READY   STATUS    RESTARTS   AGE     IP           NODE             NOMINATED NODE   READINESS GATES
my-elixir-app-9b5cfb59-bkpn9    1/1     Running   0          2m30s   10.1.0.198   docker-desktop   <none>           <none>
my-elixir-app-9b5cfb59-qndsc    1/1     Running   0          2m30s   10.1.0.196   docker-desktop   <none>           <none>
my-elixir-app-9b5cfb59-zrc6d    1/1     Running   0          2m30s   10.1.0.197   docker-desktop   <none>           <none>
```



```
PC-Tristan# kubectl get svc -o wide
NAME                    TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)   AGE    SELECTOR
kubernetes              ClusterIP   10.96.0.1        <none>        443/TCP   120d   <none>
my-elixir-app-service   ClusterIP   10.104.202.255   <none>        80/TCP    5m4s   app.kubernetes.io/instance=myapp-node,app.kubernetes.io/name=my-elixir-app
PC-Tristan# kubectl port-forward svc/my-elixir-app-service 8080:80
Forwarding from 127.0.0.1:8080 -> 4369
Forwarding from [::1]:8080 -> 4369
```

Testing

- Test-Driven-Development
- 19 tests proving different scenarios



```
# Tests the correct matching between the created bets
test "bet_match_2" do
  assert {:ok,_} = BetUnfair.clean("testdb")
  assert {:ok,_} = BetUnfair.start_link("testdb")
  assert {:ok,u1} = BetUnfair.user_create("u1","Tristan")
  assert {:ok,u2} = BetUnfair.user_create("u2","Manuel")
  assert is_ok(BetUnfair.user_deposit(u1,4700))
  assert is_ok(BetUnfair.user_deposit(u2,40500))
  assert {:ok,m1} = BetUnfair.market_create("rmw","Real Madrid wins")
  assert {:ok,b} = BetUnfair.bet_back(u1,m1,1400,200)
  assert {:ok,a} = BetUnfair.bet_back(u1,m1,2000,300)
  assert {:ok,c} = BetUnfair.bet_back(u1,m1,500,153)
  assert {:ok,e} = BetUnfair.bet_lay(u2,m1,40000,110)
  assert {:ok,f} = BetUnfair.bet_back(u1,m1,800,150)
  assert {:ok,g} = BetUnfair.bet_lay(u2,m1,500,153)
  assert is_ok(BetUnfair.market_match(m1))
  assert {:ok, %{bet_id: a, bet_type: :back, market_id: m1, user_id: u1, odds: 300, original_stake: 2000, remaining_stake: 2000, matched_bets: [], status: :active}} = BetUnfair.bet_get(a)
  assert {:ok, %{bet_id: b, bet_type: :back, market_id: m1, user_id: u1, odds: 200, original_stake: 1400, remaining_stake: 1400, matched_bets: [], status: :active}} = BetUnfair.bet_get(b)
  assert {:ok, %{bet_id: c, bet_type: :back, market_id: m1, user_id: u1, odds: 153, original_stake: 500, remaining_stake: 311, matched_bets: [g], status: :active}} = BetUnfair.bet_get(c)
  assert {:ok, %{bet_id: f, bet_type: :back, market_id: m1, user_id: u1, odds: 150, original_stake: 800, remaining_stake: 0, matched_bets: [g], status: :active}} = BetUnfair.bet_get(f)
  assert {:ok, %{bet_id: e, bet_type: :lay, market_id: m1, user_id: u2, odds: 110, original_stake: 40000, remaining_stake: 40000, matched_bets: [], status: :active}} = BetUnfair.bet_get(e)
  assert {:ok, %{bet_id: g, bet_type: :lay, market_id: m1, user_id: u2, odds: 153, original_stake: 500, remaining_stake: 0, matched_bets: [c,f], status: :active}} = BetUnfair.bet_get(g)
end
```

Future improvements: Phoenix for REST API

Create a REST API to connect via Kubernetes to the services created

Challenges faced

- Making the decision on when to return the money
- The matching algorithm
- Dealing with language for the first time

```
def question(question) do
    answer
end
```