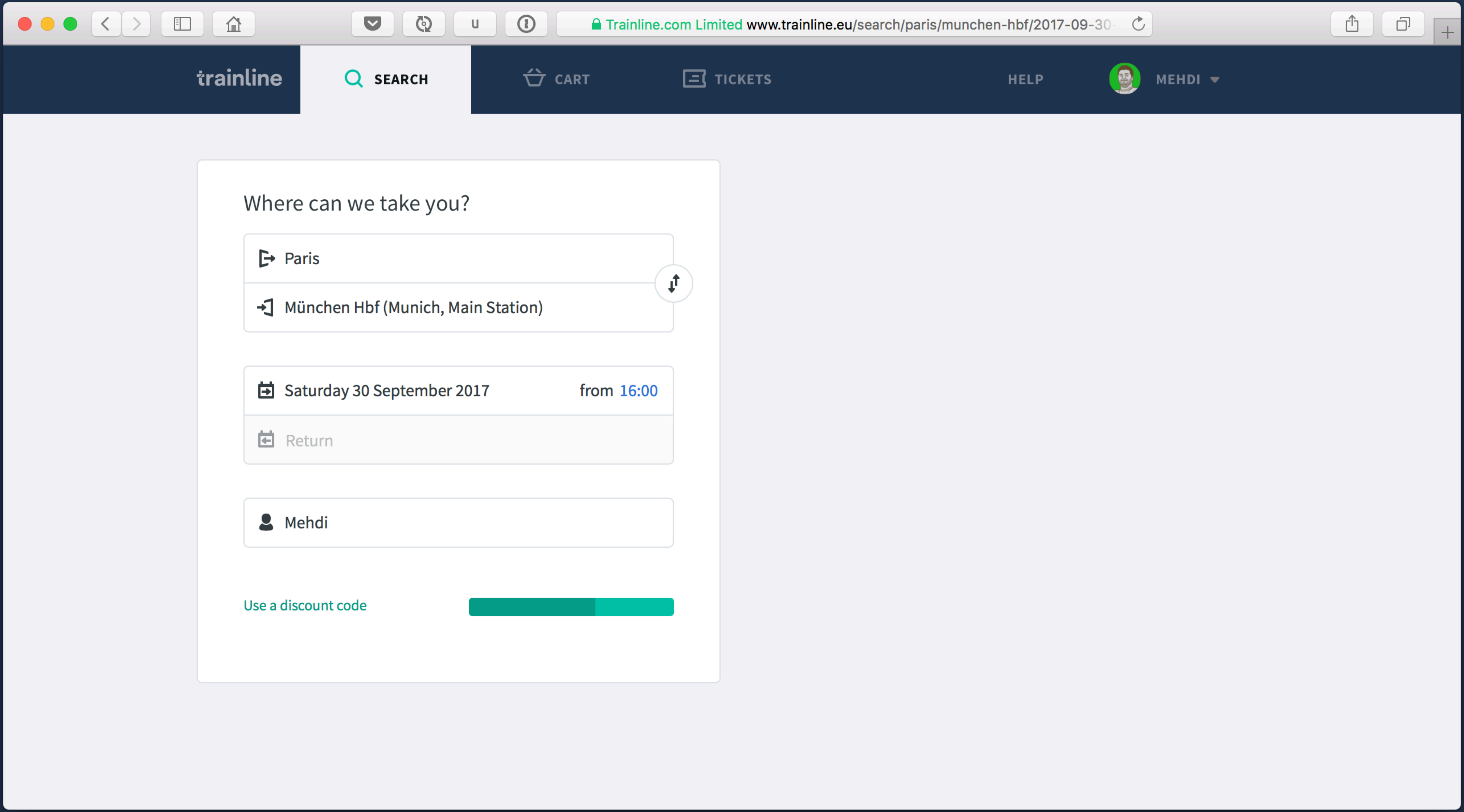
Middleware all the things

At Trainline,
we sell train tickets,
across EU and UK.



Trainline.com Limited www.trainline.eu/search/paris/munchen-hbf/2017-09-30

Where can we take you?

Paris

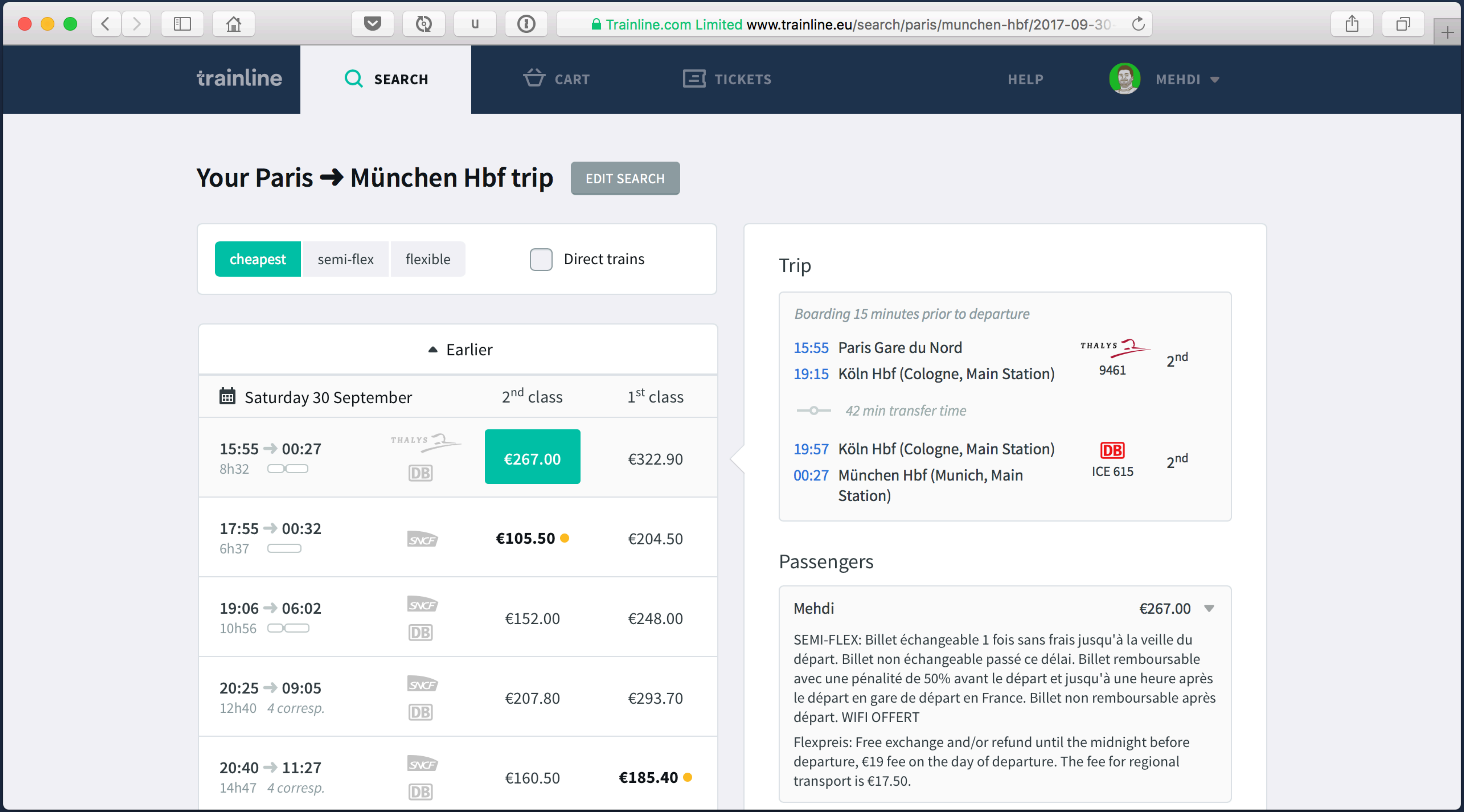
München Hbf (Munich, Main Station)

Saturday 30 September 2017 from 16:00

Return

Mehdi

Use a discount code



Your Paris ➔ München Hbf trip

Saturday 30 September		2^{nd} class	1^{st} class
15:55 → 00:27 8h32	THALYS DB	€267.00	€322.90
17:55 → 00:32 6h37	SNCF	**€105.50** ●	€204.50
19:06 → 06:02 10h56	SNCF DB	€152.00	€248.00
20:25 → 09:05 12h40 *4 corresp.*	SNCF DB	€207.80	€293.70
20:40 → 11:27 14h47 *4 corresp.*	SNCF DB	€160.50	**€185.40** ●

Trip

Boarding 15 minutes prior to departure

15:55	Paris Gare du Nord	THALYS 9461	2^{nd}
19:15	Köln Hbf (Cologne, Main Station)		

42 min transfer time

19:57	Köln Hbf (Cologne, Main Station)	DB ICE 615	2^{nd}
00:27	München Hbf (Munich, Main Station)		

Passengers

Mehdi €267.00

SEMI-FLEX: Billet échangeable 1 fois sans frais jusqu'à la veille du départ. Billet non échangeable passé ce délai. Billet remboursable avec une pénalité de 50% avant le départ et jusqu'à une heure après le départ en gare de départ en France. Billet non remboursable après départ. WIFI OFFERT

Flexpreis: Free exchange and/or refund until the midnight before departure, €19 fee on the day of departure. The fee for regional transport is €17.50.

How does it work?

Frontends

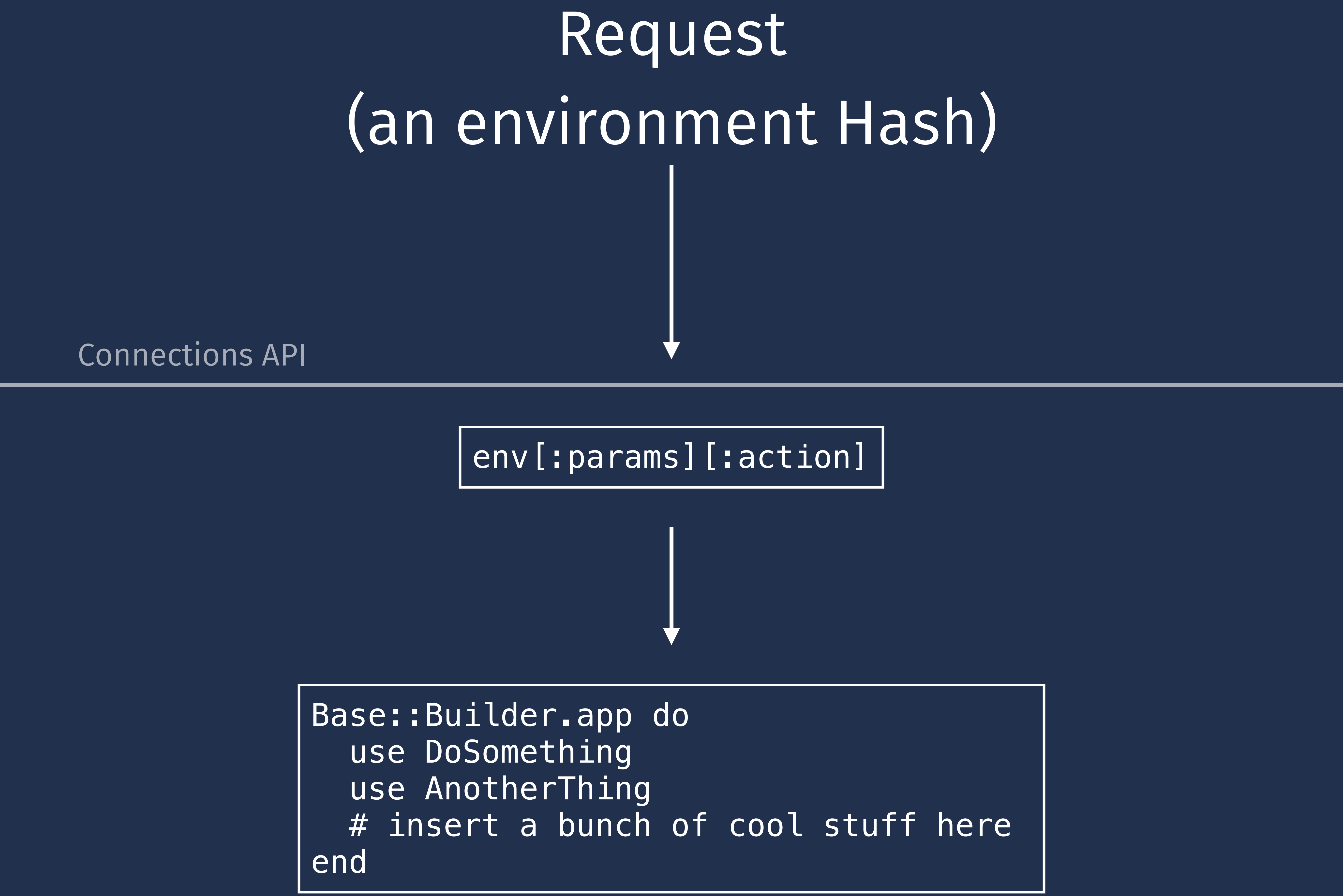
Rails API

Connections API

aka connection with carriers APIs

A bunch of middlewares

Requests are processed by generated apps on the fly, which are stacks of middlewares.



Request
(an environment Hash)

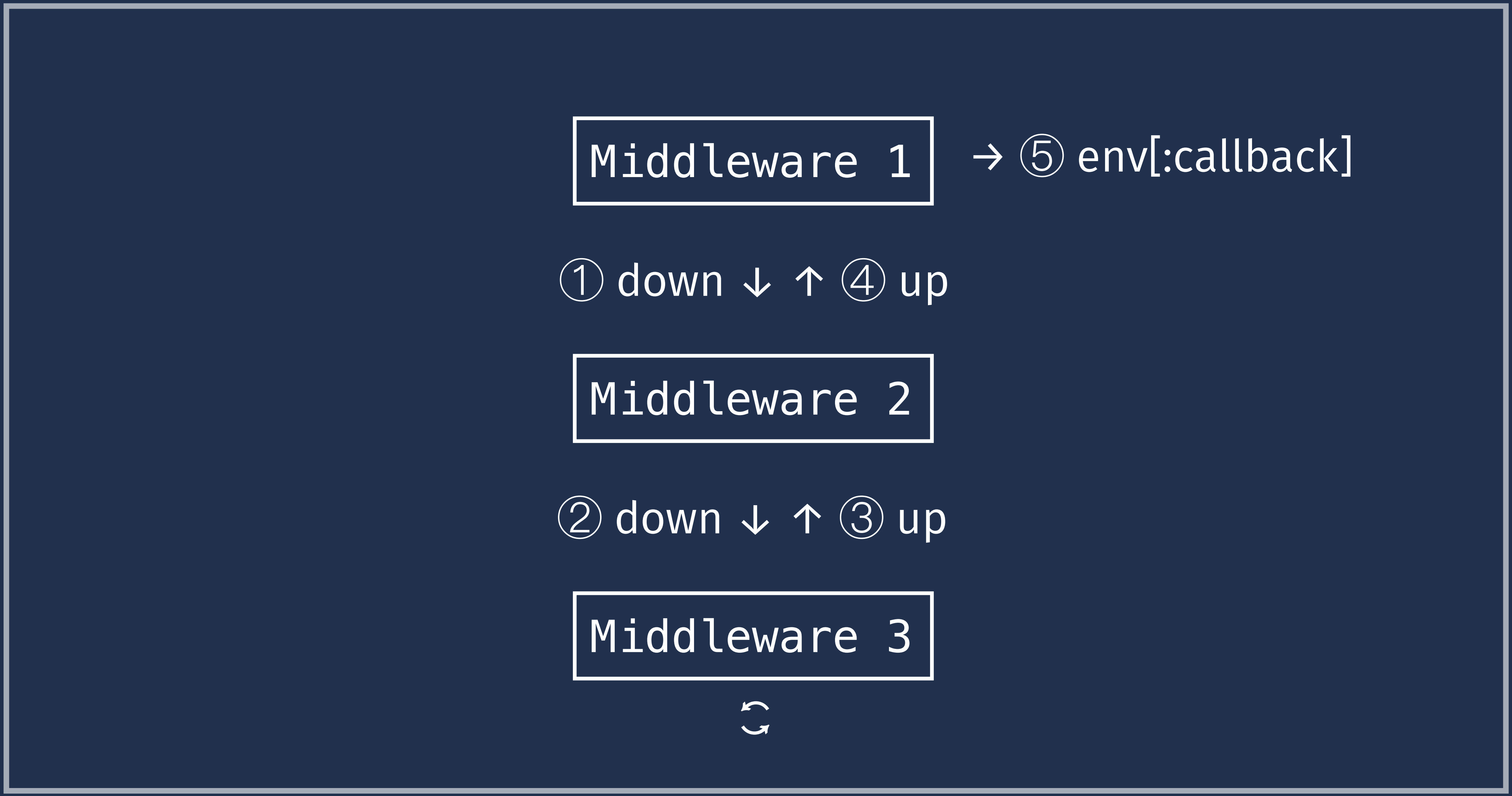
Connections API

```
env[:params][:action]
```

```
Base::Builder.app do
  use DoSomething
  use AnotherThing
  # insert a bunch of cool stuff here
end
```

Connections API

An action (an app)



Middleware 1 → ⑤ env[:callback]

① down ↓ ↑ ④ up

Middleware 2

② down ↓ ↑ ③ up

Middleware 3

🔄

```
% tree lib/pacon/actions -L 1

lib/pacon/actions
├── book.rb
├── book_subscription.rb
├── cancel.rb
├── emit.rb
├── emit_subscription.rb
├── estimate_exchange.rb
├── estimate_refund.rb
├── exchange.rb
├── exchange_pay.rb
├── exchange_search.rb
├── fetch_pnr.rb
├── fetch_subscription.rb
├── fetch_travel_document.rb
├── immediate_cancel.rb
├── load_timetable.rb
├── pay.rb
├── refund.rb
└── search.rb
```

```
def self.search
  Base::Builder.app do
    use Base::LogAction
    use Base::VolatileStations
    use Base::InitResultCache
    use Base::LogSearchFruitfulness
    use Base::AvoidFruitlessSearch
    use Base::ReportAndBlockCombinations
    use Base::TryAgain
    use Base::SearchWindow
    use Base::FilterUnsellableFolderMix
    use Base::FilterTimetableResults
    use Base::TooManyPassengers
    use Base::WarnDifferentStation
    use Base::Parallel
  end
end
```

```
def self.search
  Base::Builder.app do
    use Base::LogAction
    use Base::VolatileStations
    use Base::InitResultCache
    use Base::LogSearchFruitfulness
    use Base::AvoidFruitlessSearch
    use Base::ReportAndBlockCombinations
    use Base::TryAgain
    use Base::SearchWindow
    use Base::FilterUnsellableFolderMix
    use Base::FilterTimetableResults
    use Base::TooManyPassengers
    use Base::WarnDifferentStation
    use Base::Parallel
  end
end
```

Will call `SNCF.search, DB.search`…

```
module SNCF
  class << self

    def search
      Base::Builder.app do
       use SNCF::Stations
        use SNCF::ExtractFoldersFromJourneys
        # tons of very secret sauce
        use SNCF::HttpRequest
      end
    end
  end
end
```

Event Machine FTW

The whole is based on Event Machine to coordinate all the parallel requests made to carriers APIs.