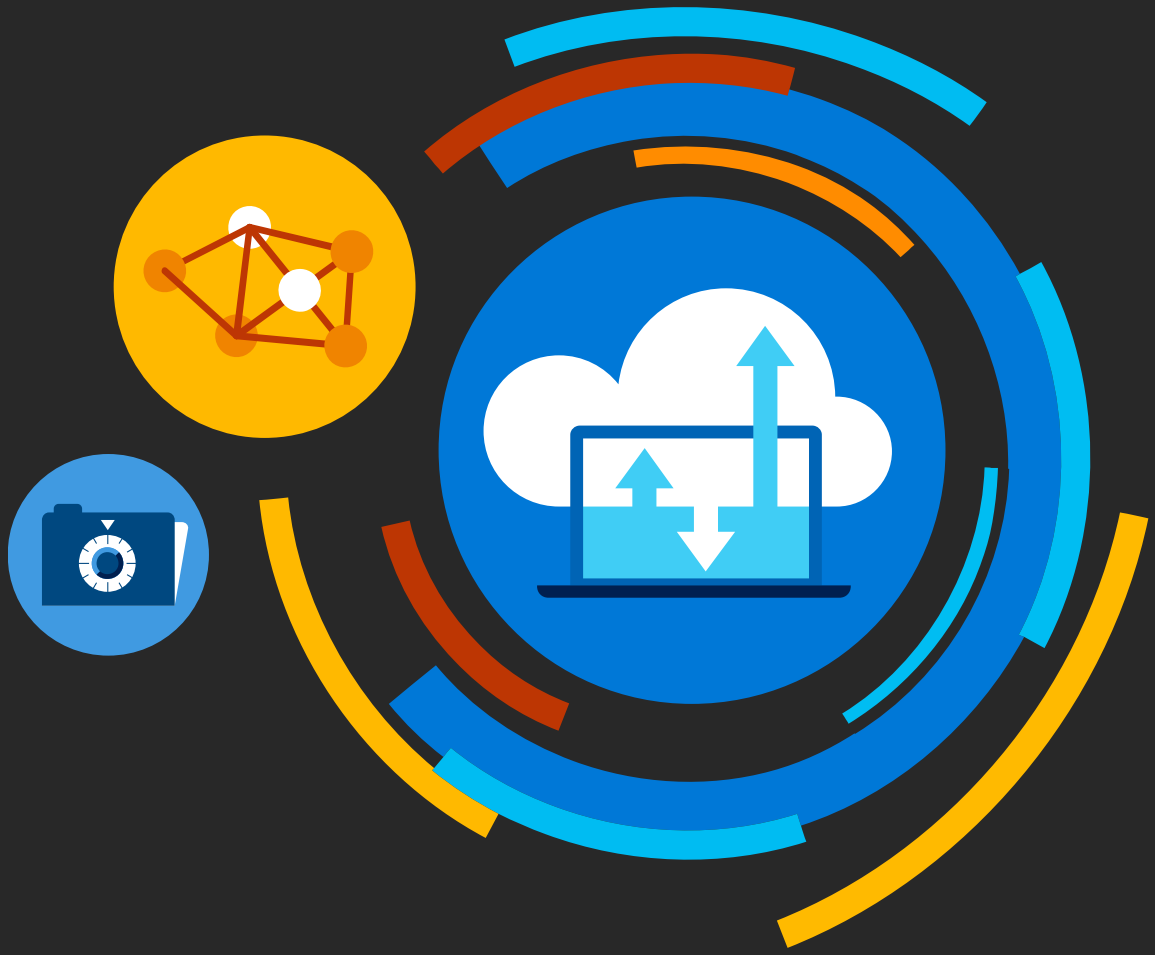
Microsoft

#techsummitCH

From Zero to Azure with Angular

@FabianGosebrink




from Zero
to Azure
with Angular


@FabianGosebrink


@FabianGosebrink
@EverythingGoats

Swiss Angular

myCSharp.de
gemeinsam mehr erreichen

Google Developers Experts

Microsoft® MVP Most Valuable Professional

@FabianGosebrink

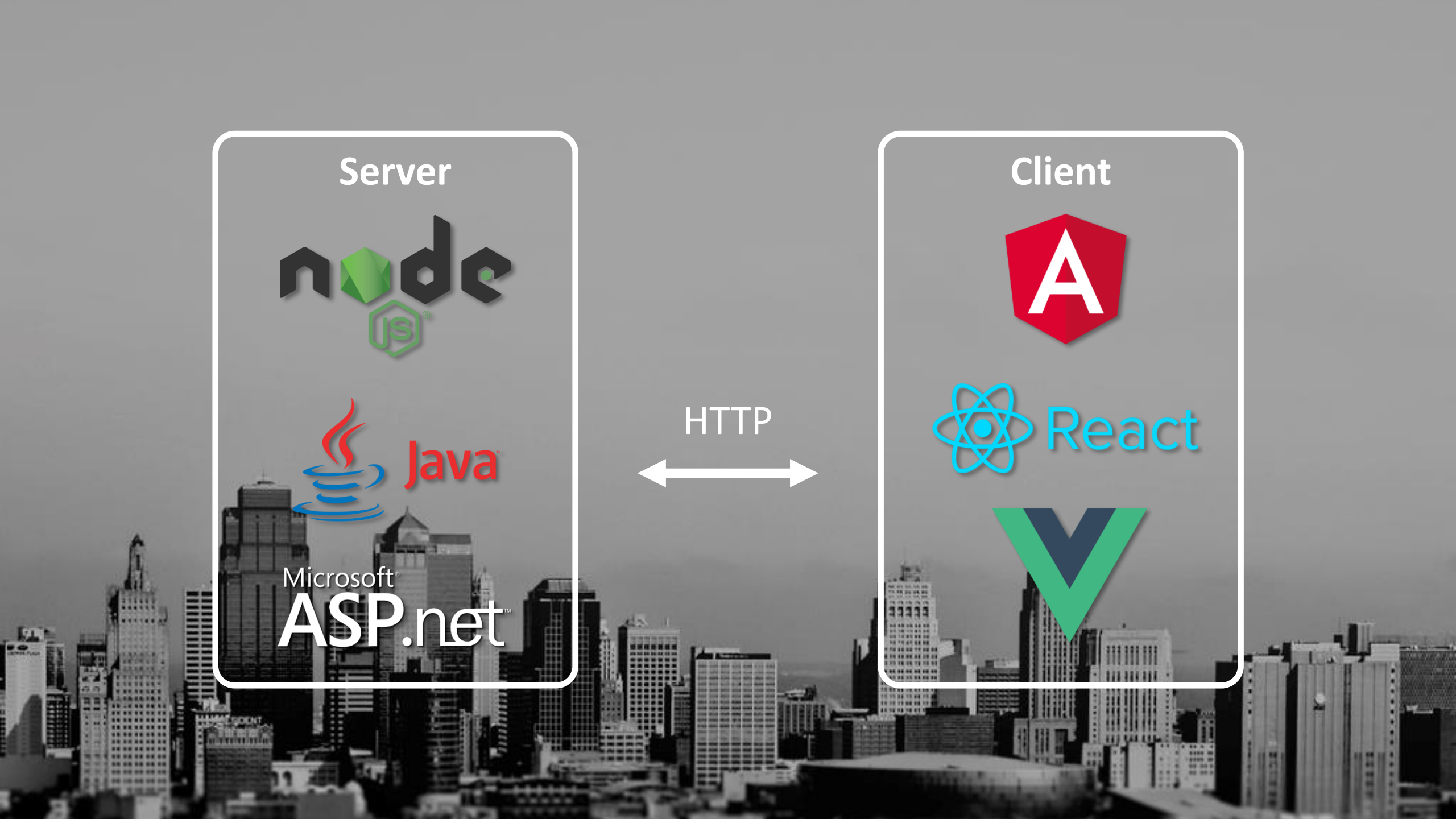
ZÜRICH DEV .NET
ZÜRICH .NET DEVELOPERS UG


OFFERING
SOLUTIONS
SOFTWARE

A<A>EMY


from Zero
to Azure
with Angular

Frontend


Server
node
JS
Java
Microsoft
ASP.net
HTTP
Client
A
React


Keep it
separated!

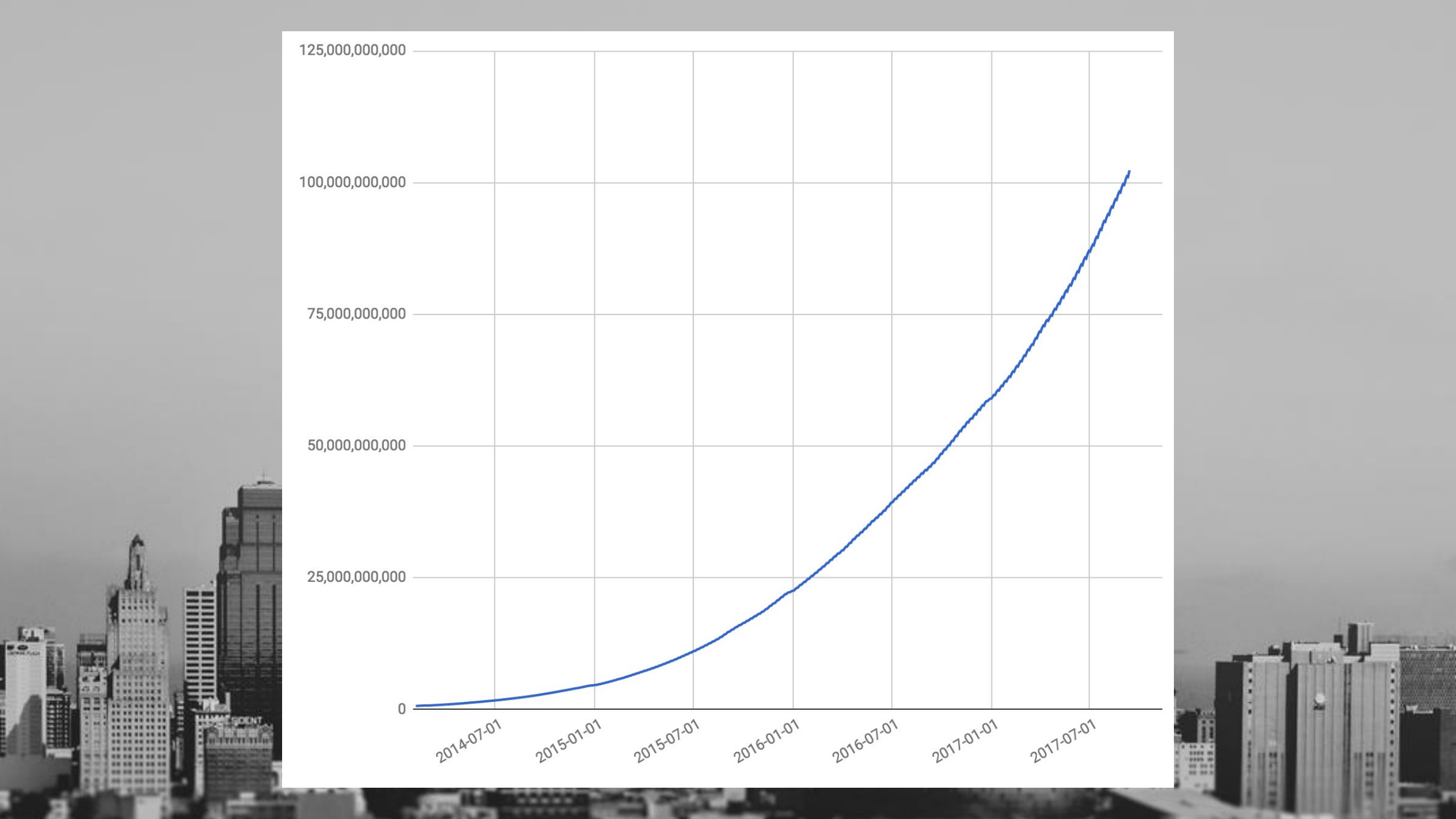

focussing on two
things
f*cking sucks

Client

Client

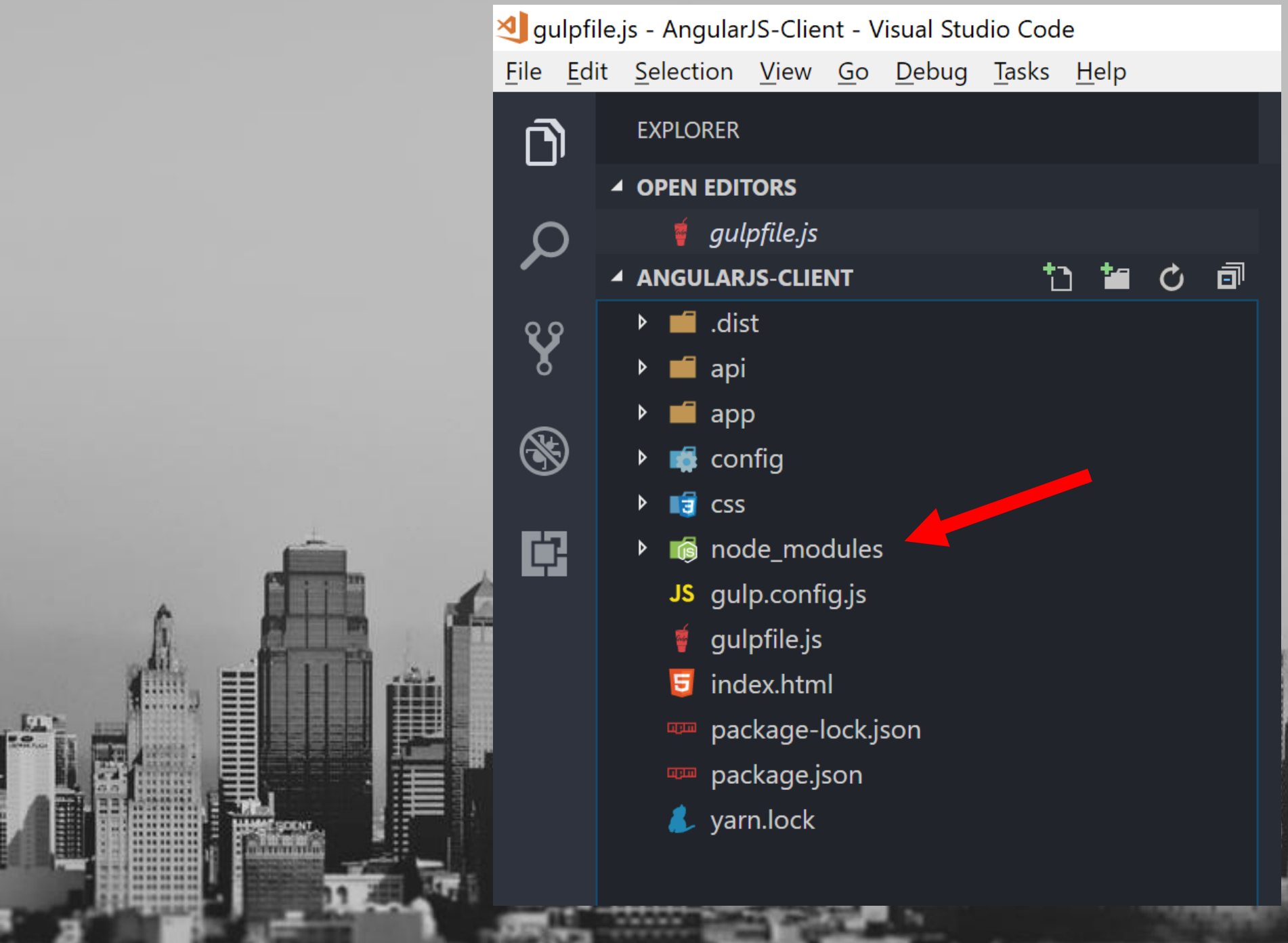

tools

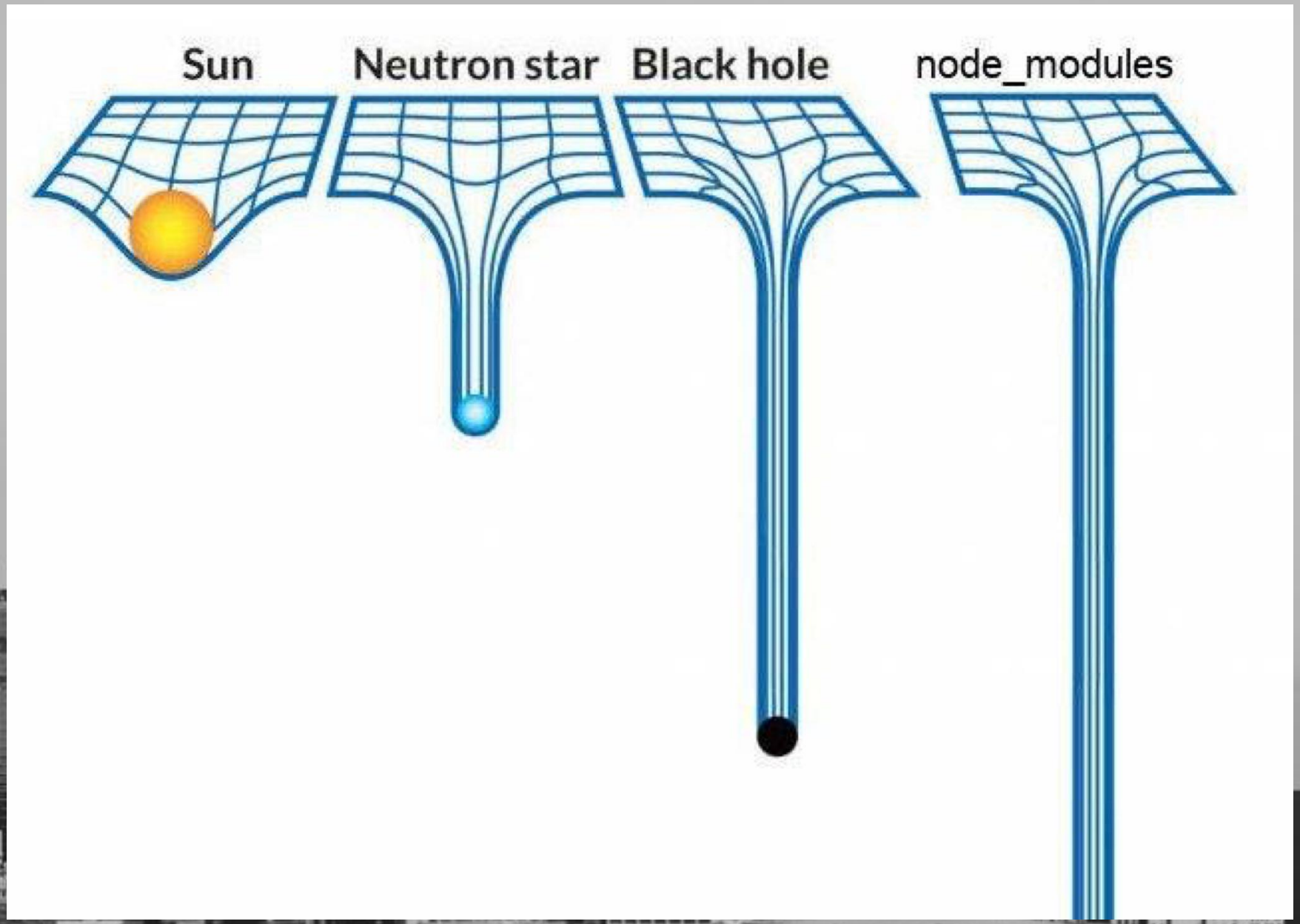

npm


125,000,000,000
100,000,000,000
75,000,000,000
50,000,000,000
25,000,000,000
0
2014-07-01
2015-01-01
2015-07-01
2016-01-01
2016-07-01
2017-01-01
2017-07-01

```
> npm install <myPackage>
```

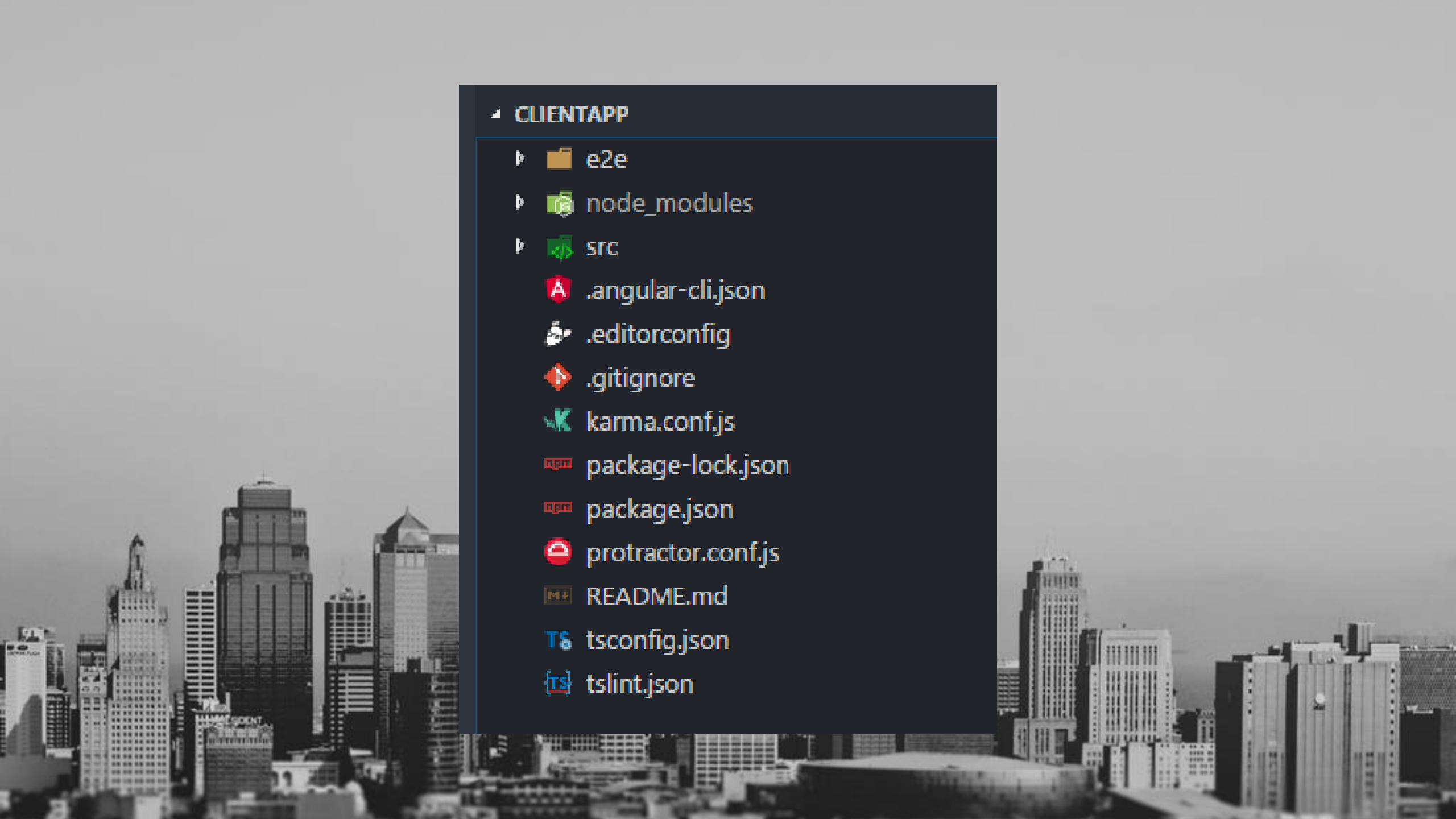
```
> npm install <myPackage> -g
```


gulpfile.js - AngularJS-Client - Visual Studio Code
File Edit Selection View Go Debug Tasks Help
EXPLORER
OPEN EDITORS
gulpfile.js
ANGULARJS-CLIENT
.dist
api
app
config
css
node_modules
gulp.config.js
gulpfile.js
index.html
package-lock.json
package.json
yarn.lock

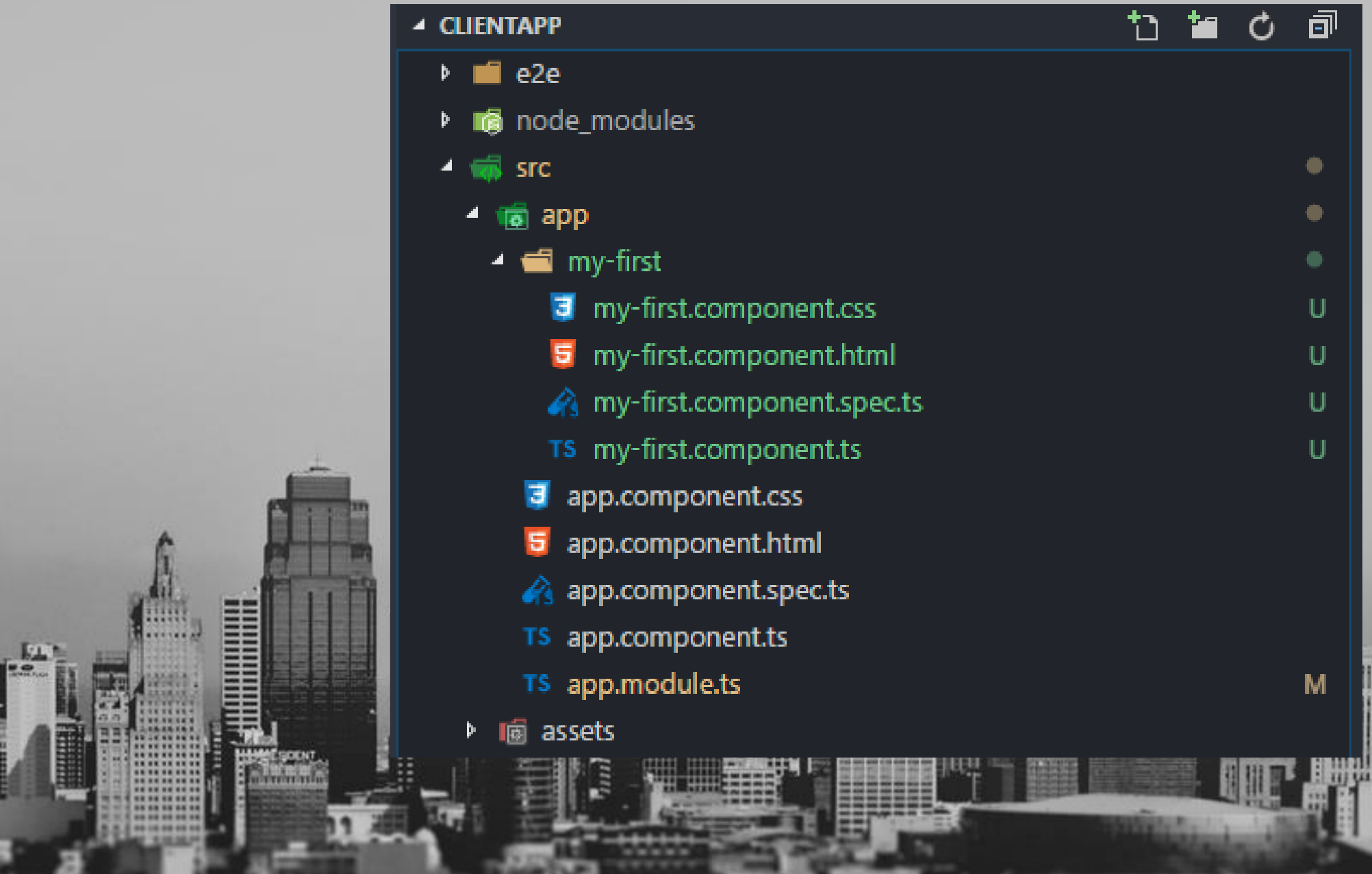

Sun
Neutron star
Black hole
node_modules

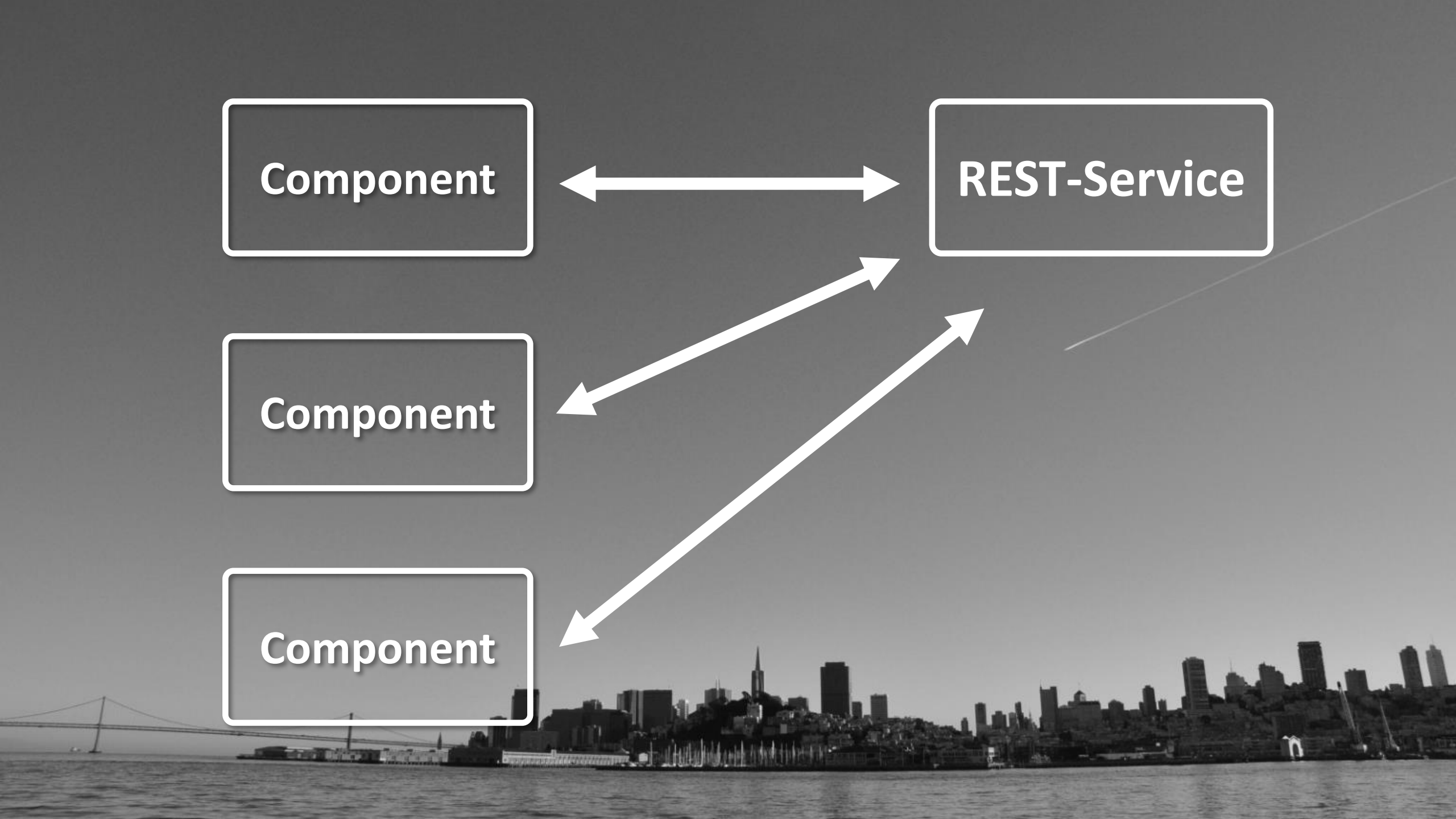
Angular CLI

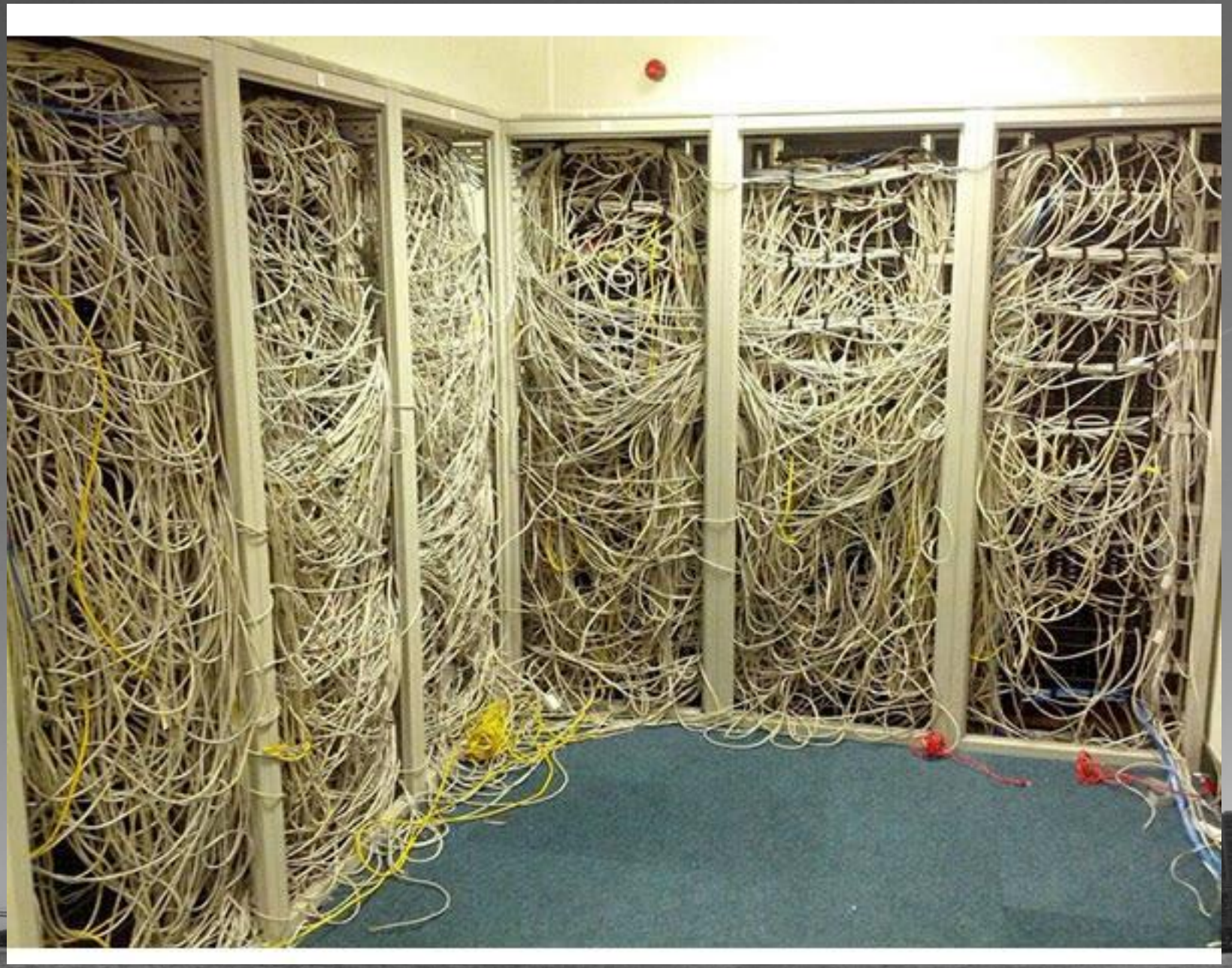

focus

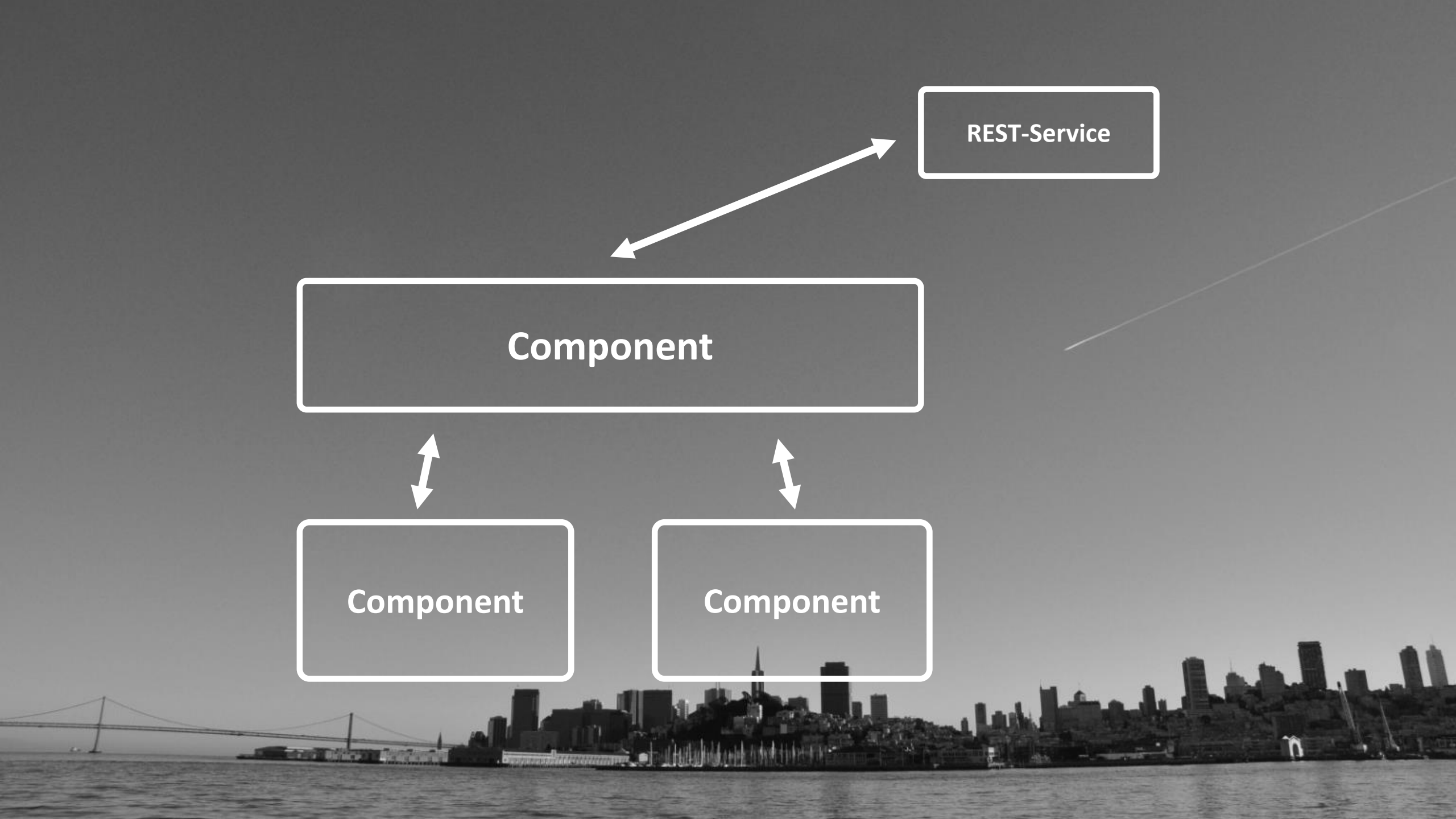

CLIENTAPP
e2e
node_modules
src
.angular-cli.json
.editorconfig
.gitignore
karma.conf.js
package-lock.json
package.json
protractor.conf.js
README.md
tsconfig.json
tslint.json

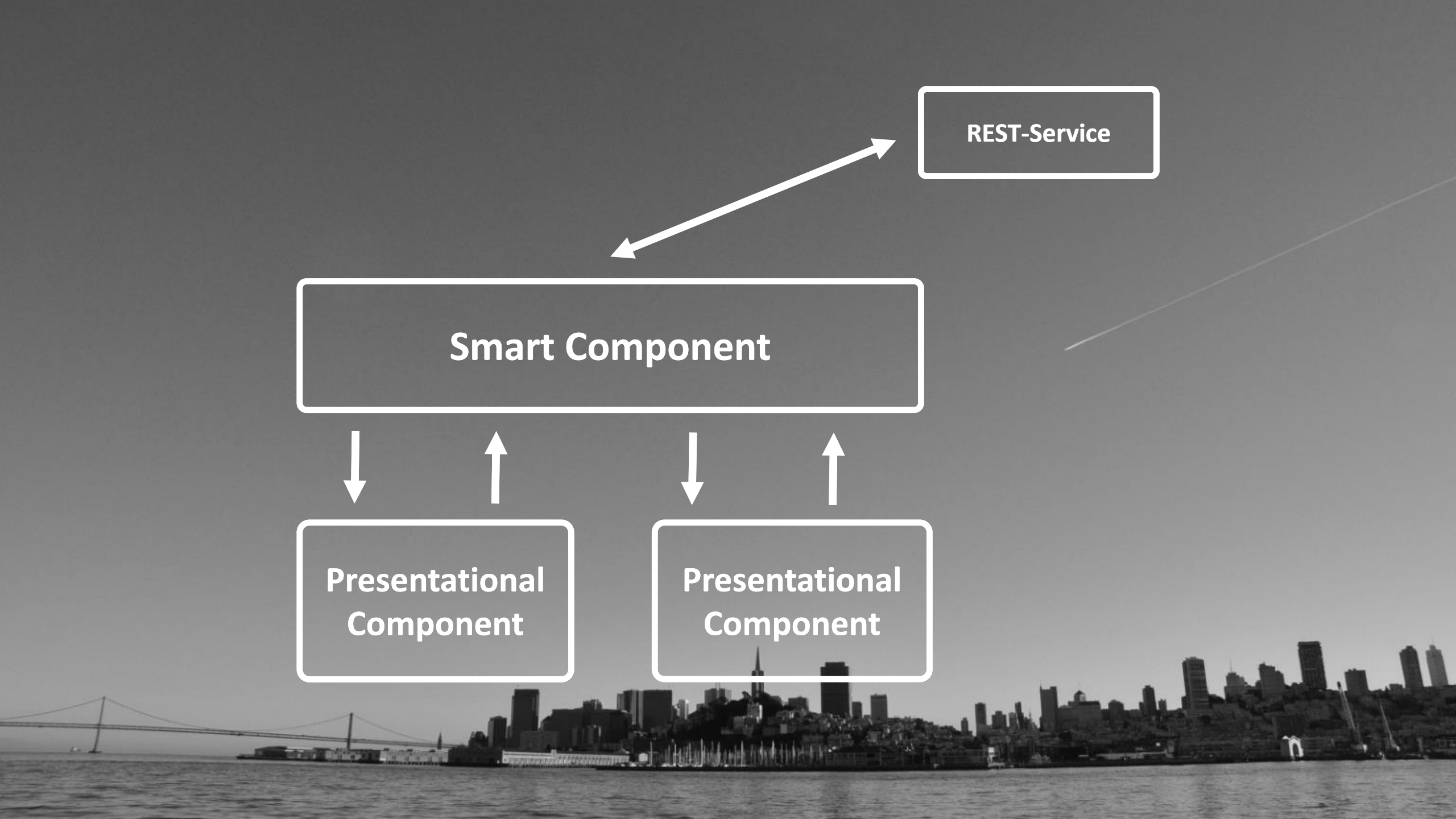
```
> ng g c myFirst
```

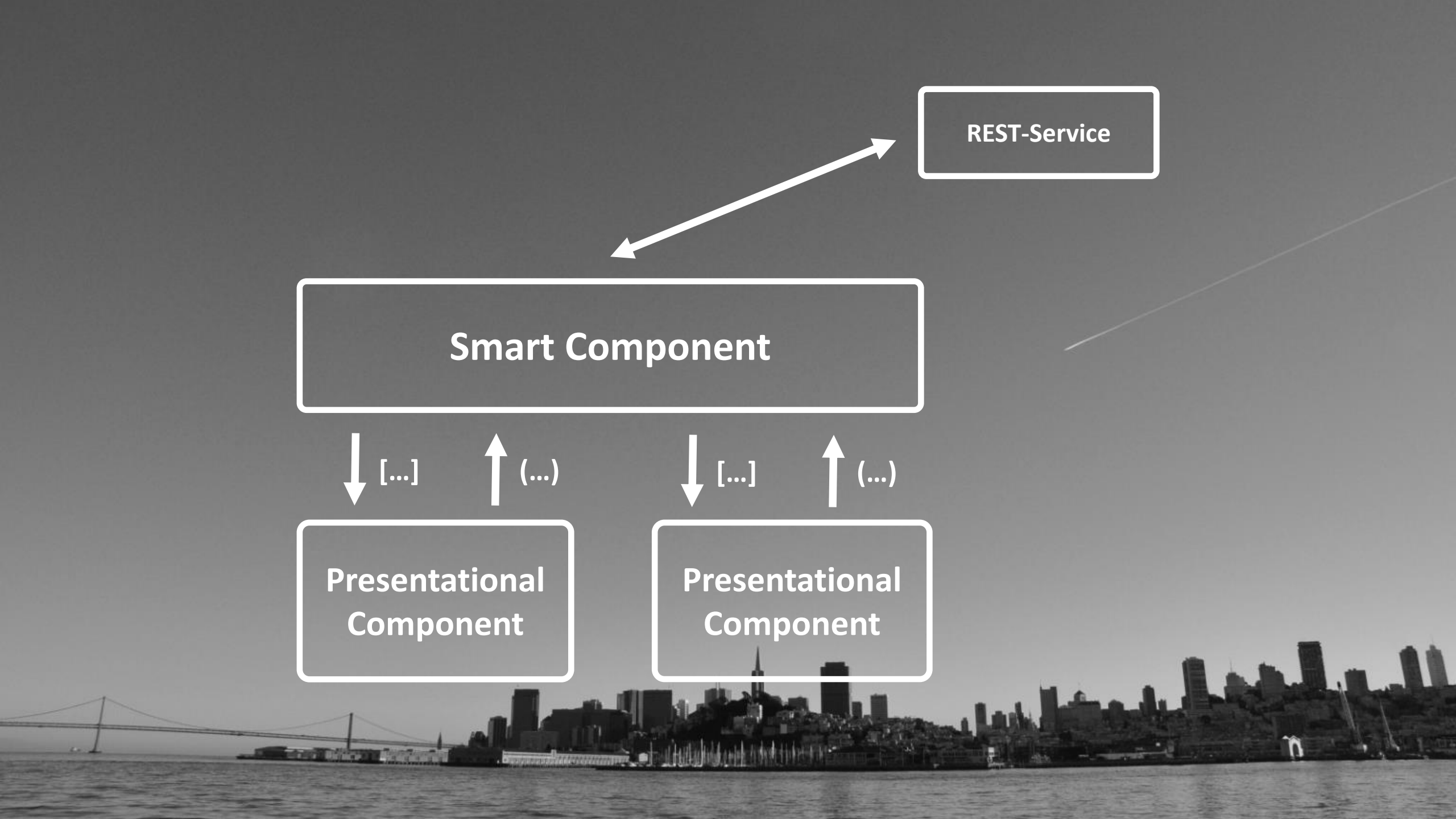

CLIENTAPP
e2e
node_modules
src
app
my-first
my-first.component.css
U
my-first.component.html
U
my-first.component.spec.ts
U
my-first.component.ts
U
app.component.css
app.component.html
app.component.spec.ts
app.component.ts
app.module.ts
M
assets

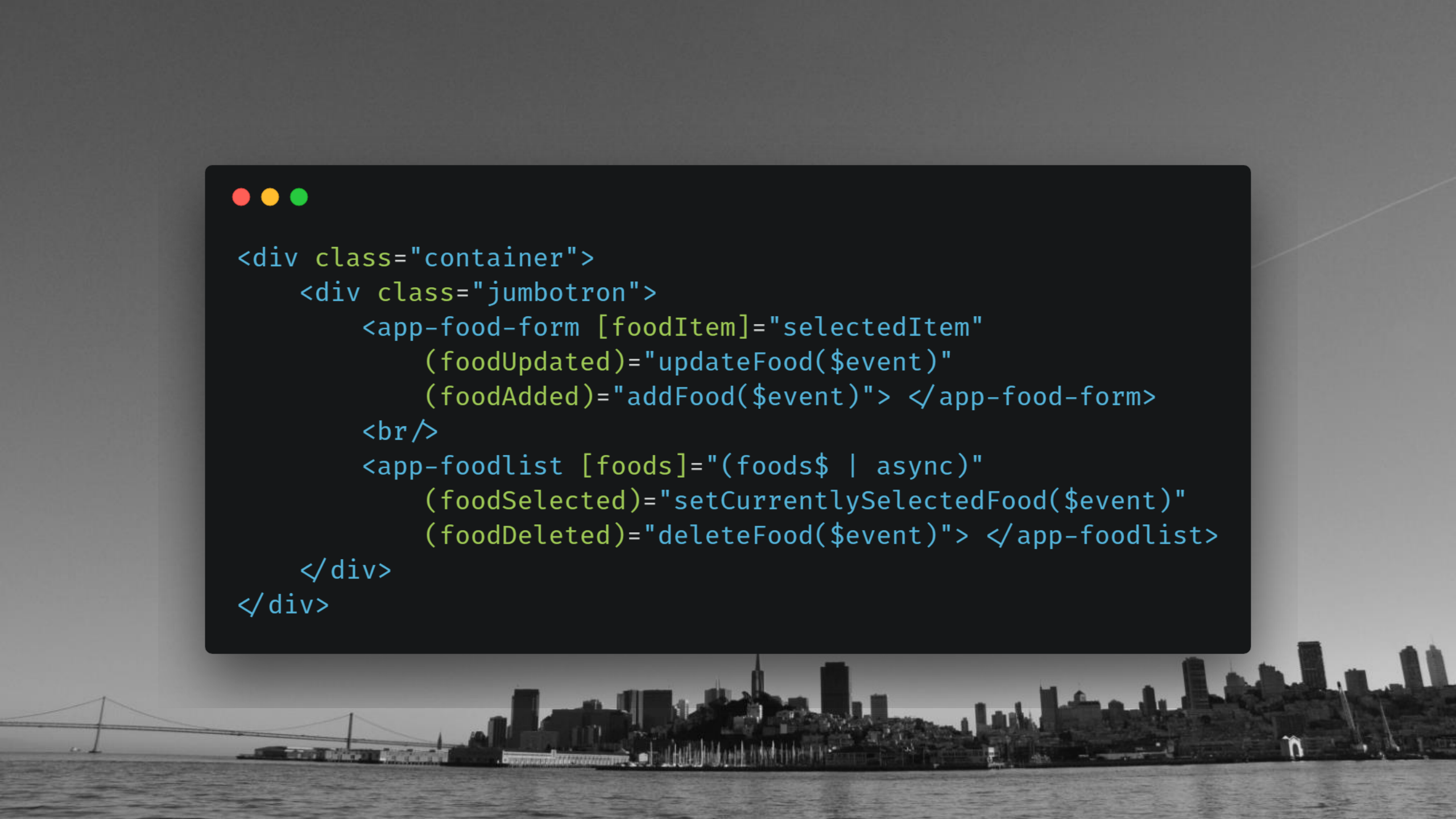

Component
Component
Component
REST-Service




REST-Service
Component
Component
Component


REST-Service
Smart Component
Presentational Component
Presentational Component


REST-Service
Smart Component
[…]
(…)
[…]
(…)
Presentational Component
Presentational Component

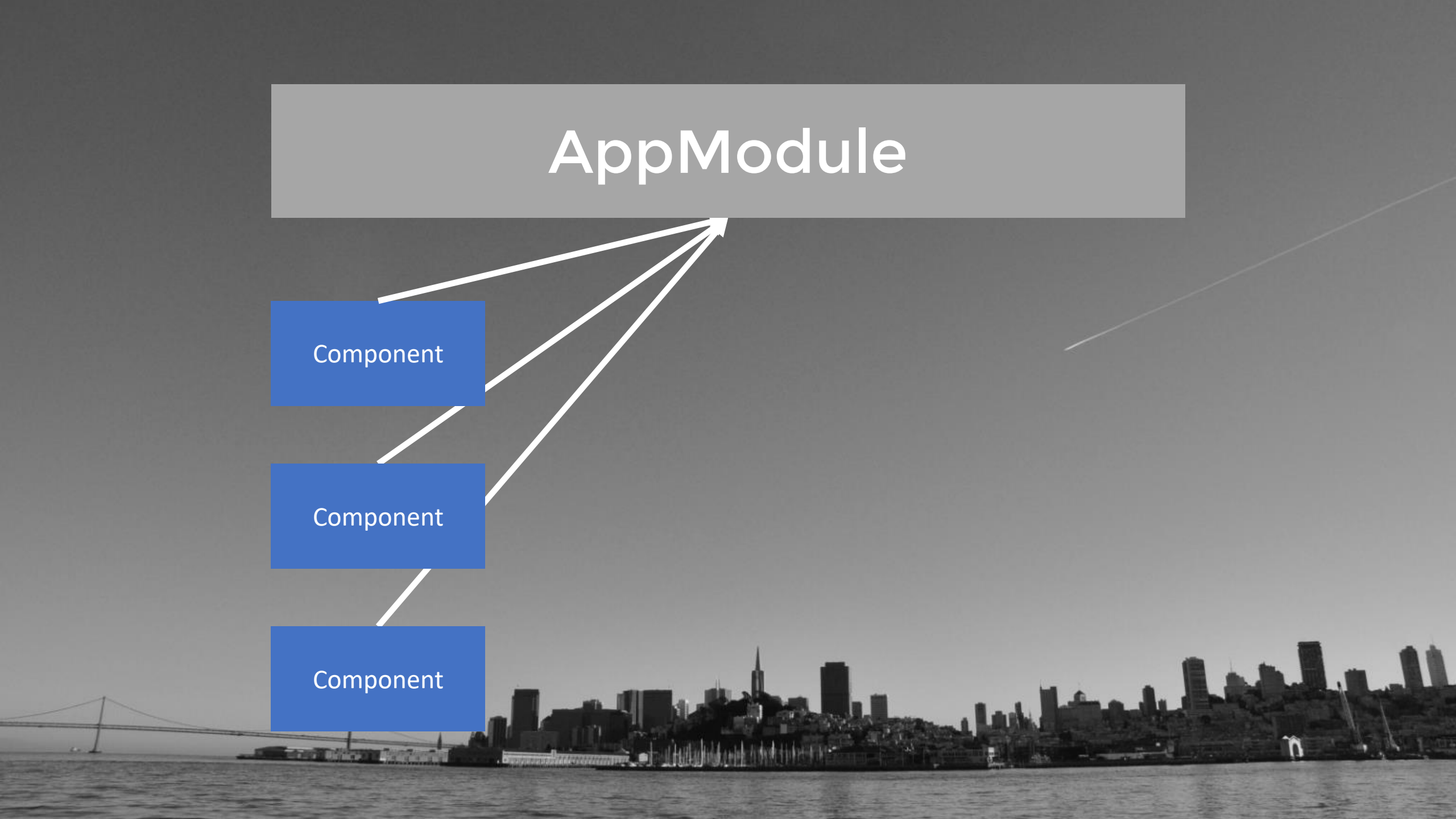


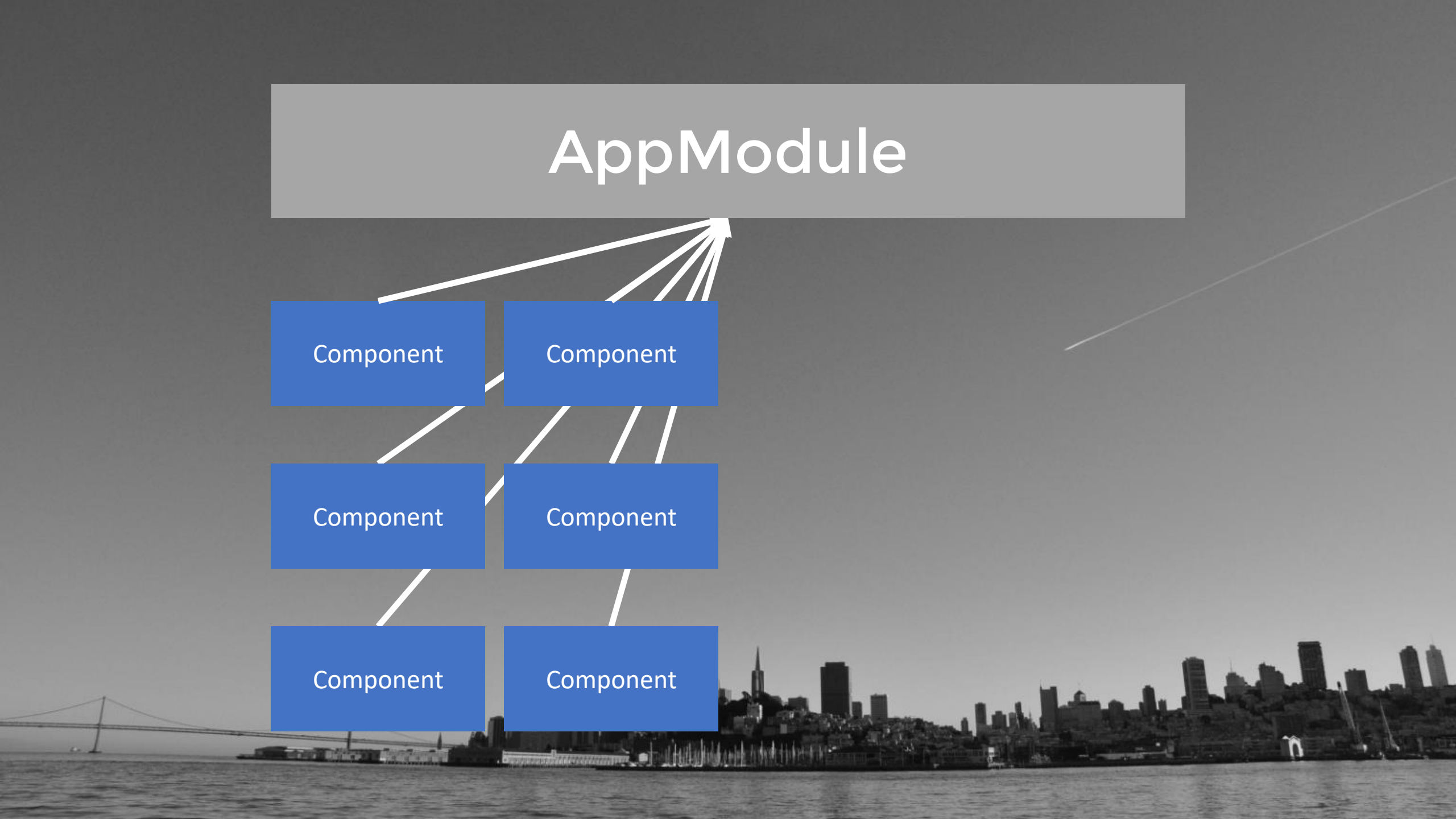
<app-food-form [foodItem]="selectedItem"
(foodUpdated)="updateFood($event)"
(foodAdded)="addFood($event)"> </app-food-form>

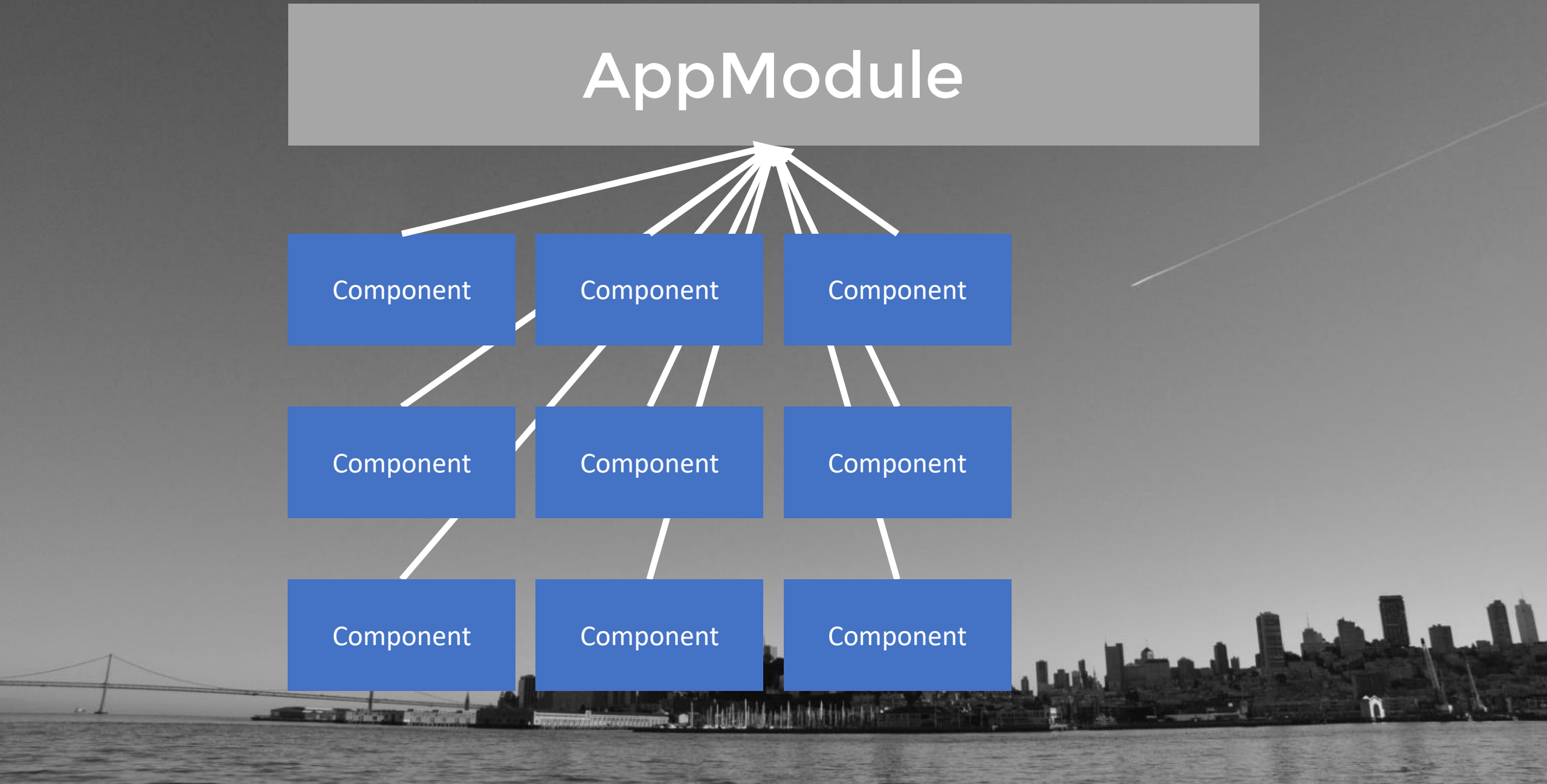
<app-foodlist [foods]="(foods$ | async)"
(foodSelected)="setCurrentlySelectedFood($event)"
(foodDeleted)="deleteFood($event)"> </app-foodlist>

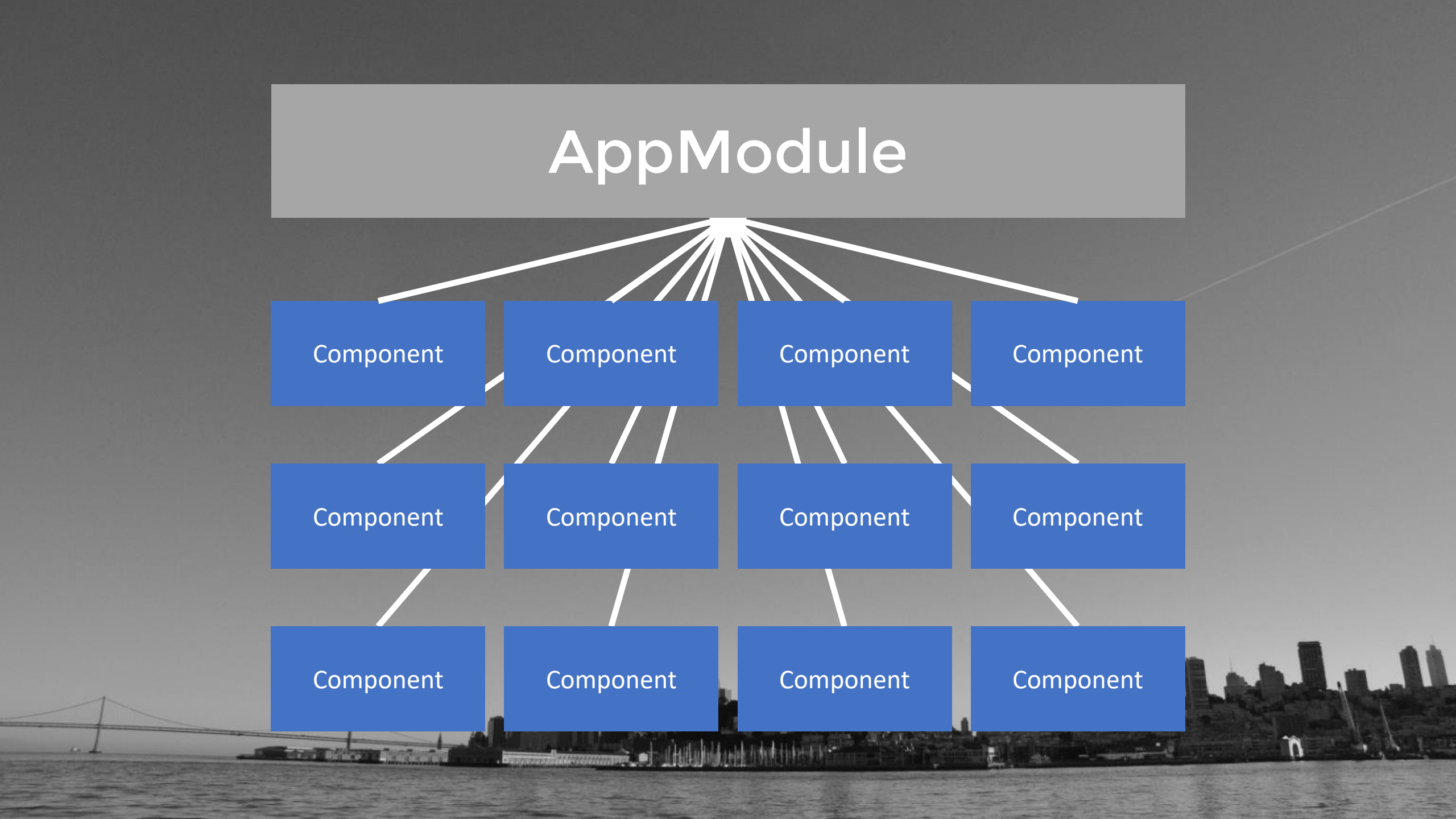

@Component({ /* ... */})
export class FoodListComponent {
@Input() foods: FoodItem[];
@Output() foodSelected = new EventEmitter<FoodItem>();
@Output() foodDeleted = new EventEmitter<FoodItem>();
// ... Logic!
}

AppModule


AppModule
Component
Component
Component


AppModule
Component
Component
Component
Component
Component
Component


AppModule
Component
Component
Component
Component
Component
Component
Component
Component
Component


AppModule
Component
Component
Component
Component
Component
Component
Component
Component
Component
Component
Component
Component

Separate into modules

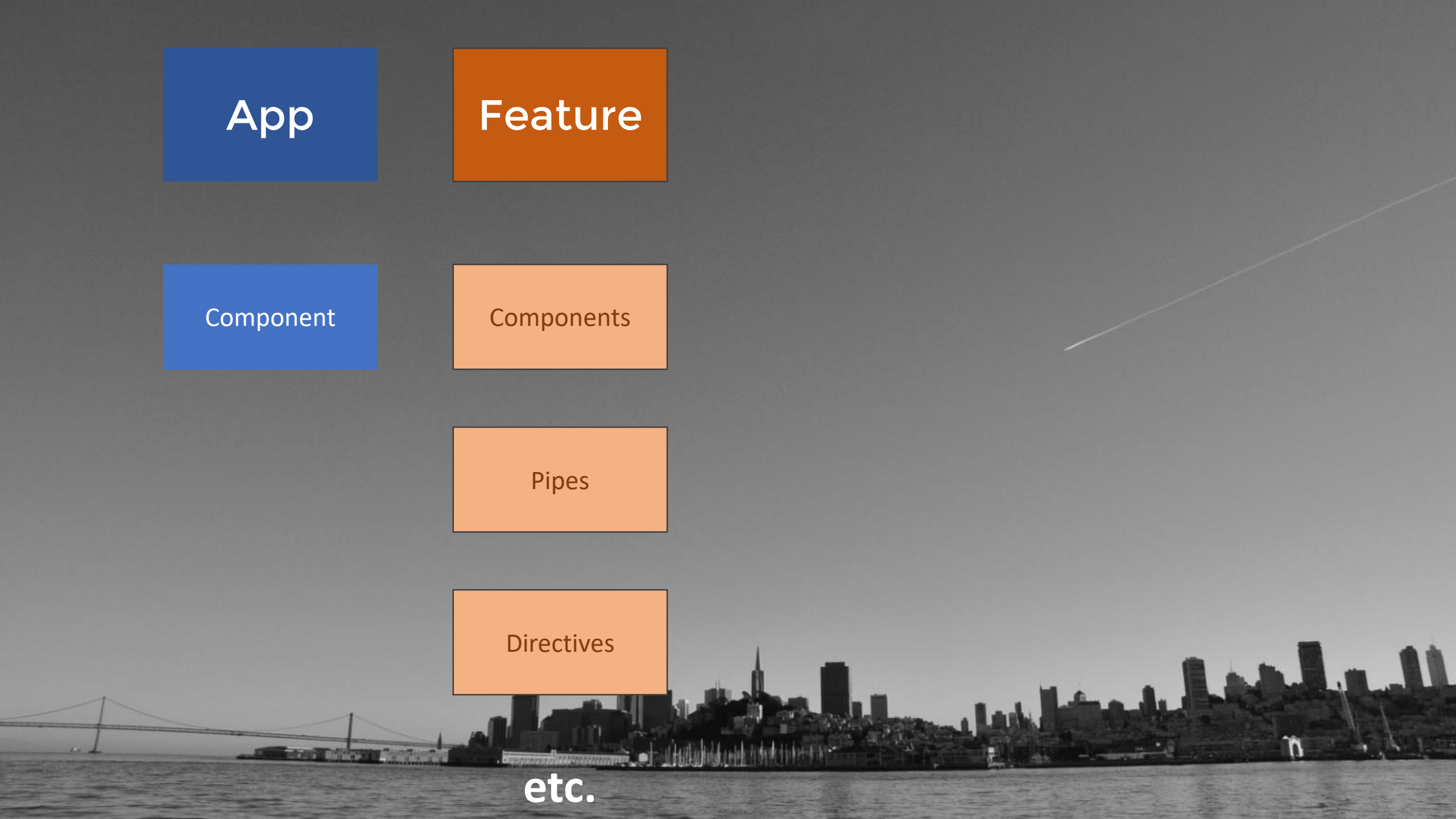

@NgModule({
imports: [
// Modules
],
declarations: [
// Components & directives
],
providers: [
// Services
],
exports: [/* ... */]
})

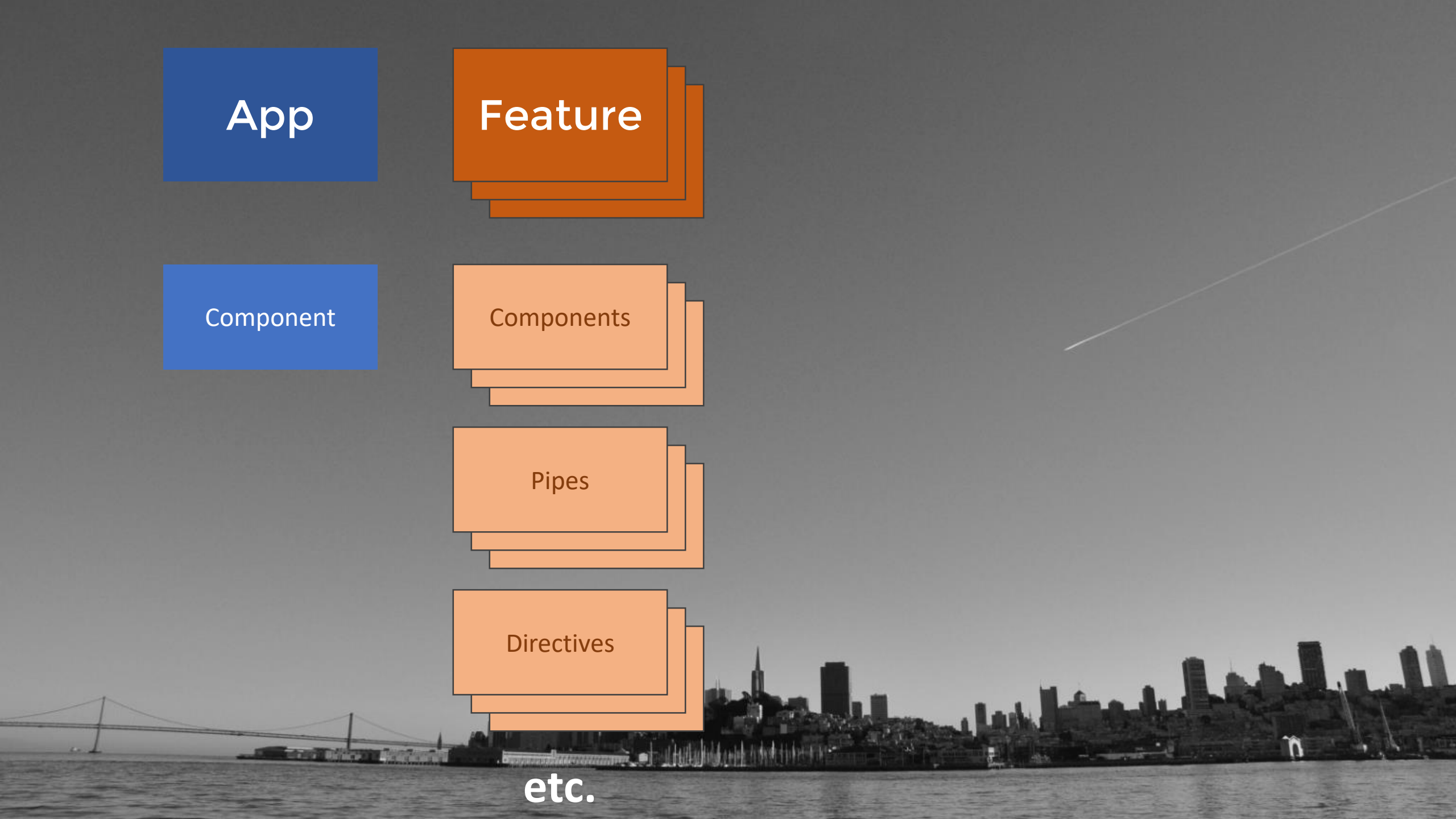
When to create a module?

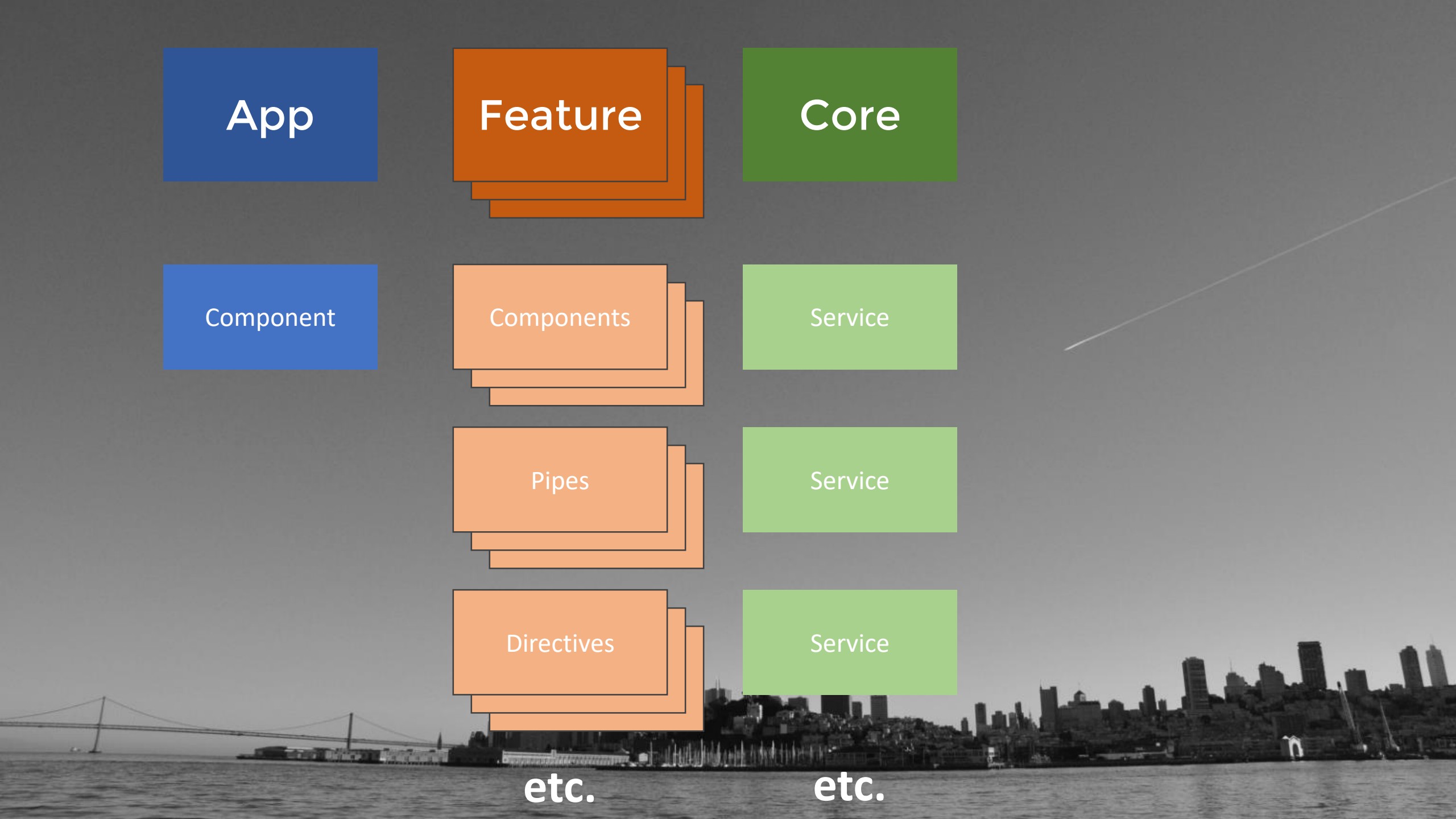
“When to create a module?”

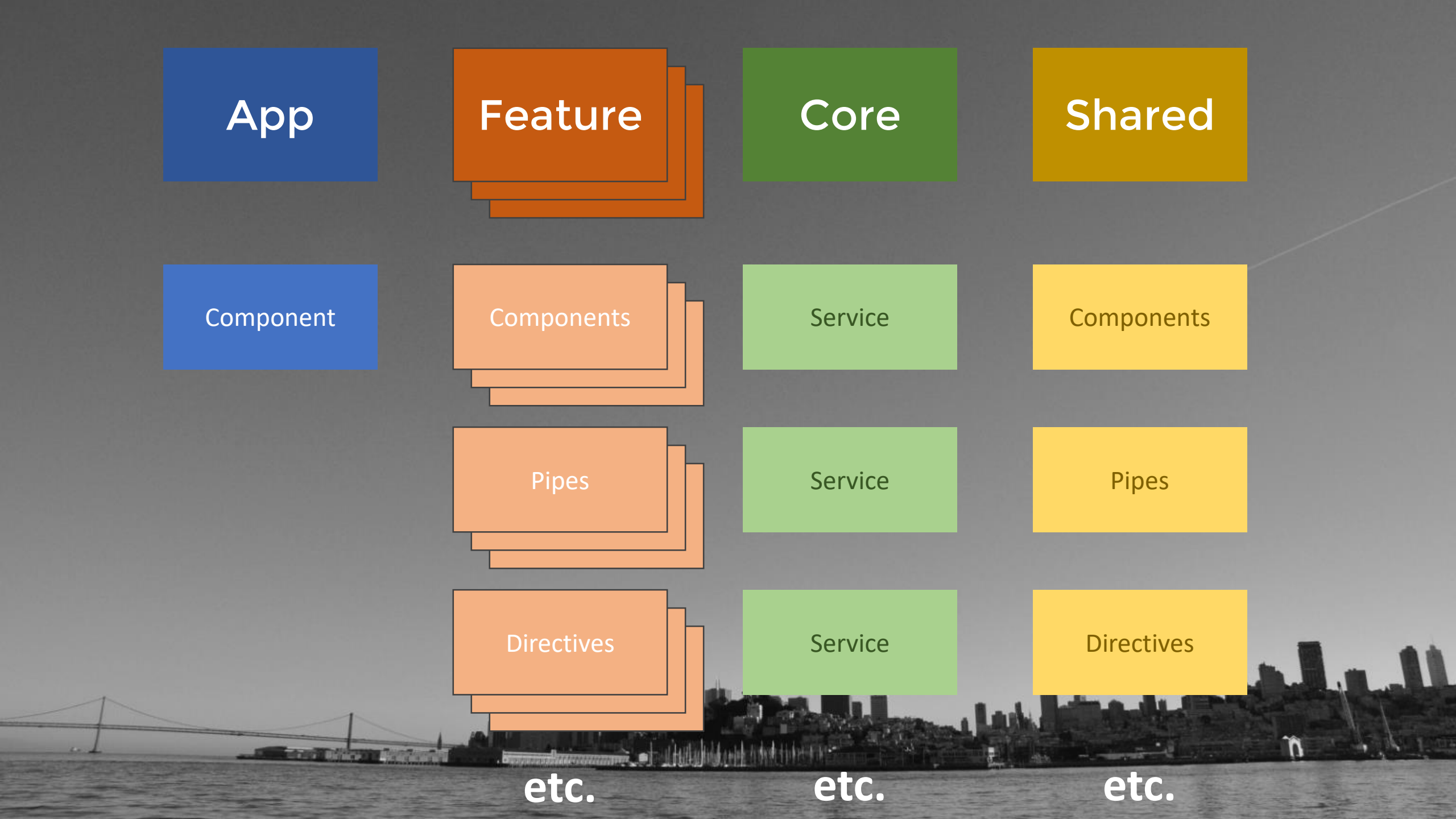
App

Component


App
Feature
Component
Components
Pipes
Directives
etc.


App
Feature
Component
Components
Pipes
Directives
etc.

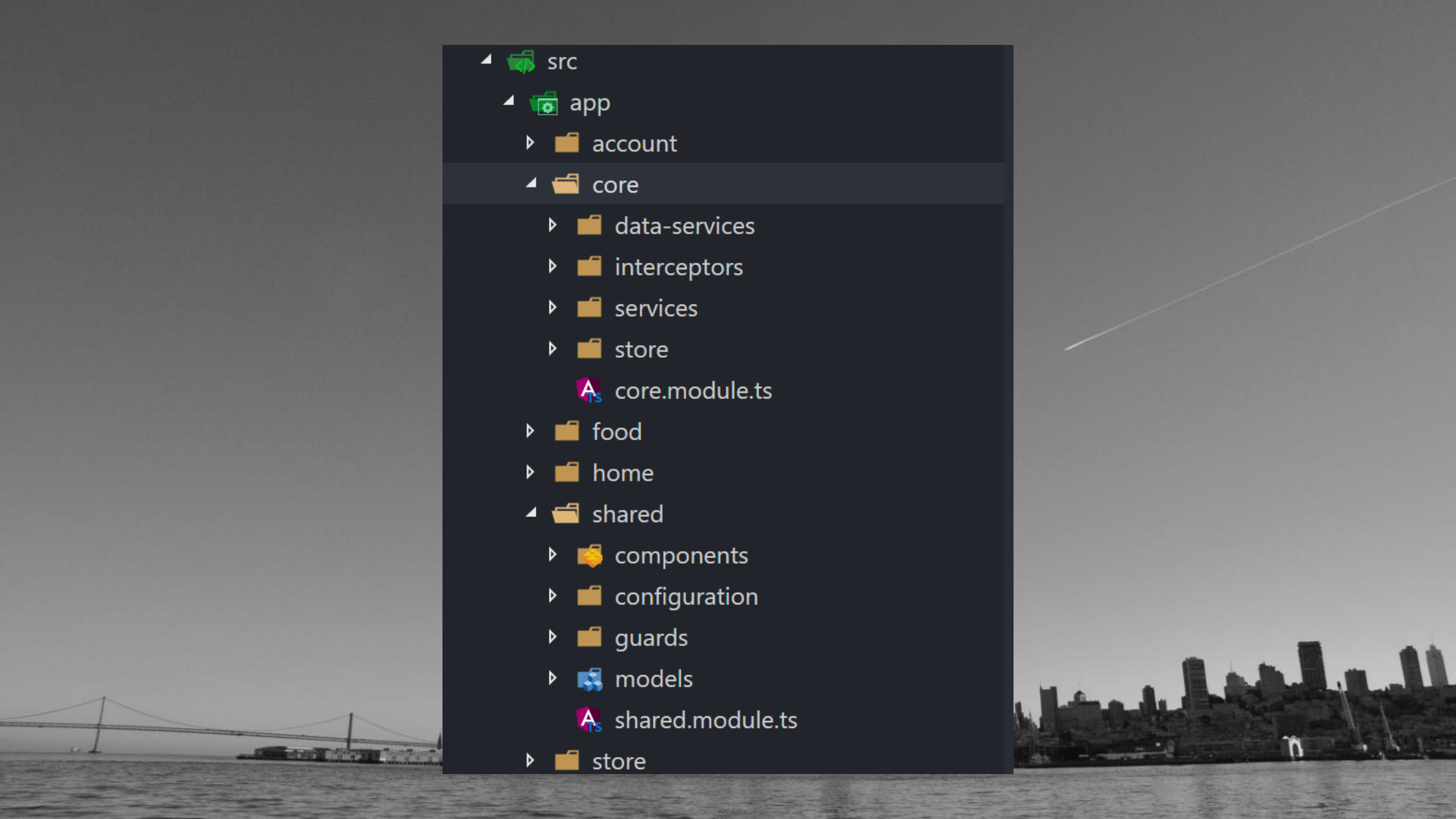

App
Feature
Core
Component
Components
Service
Pipes
Service
Directives
Service
etc.
etc.


App
Feature
Core
Shared
Component
Components
Service
Components
Pipes
Service
Pipes
Directives
Service
Directives
etc.
etc.
etc.

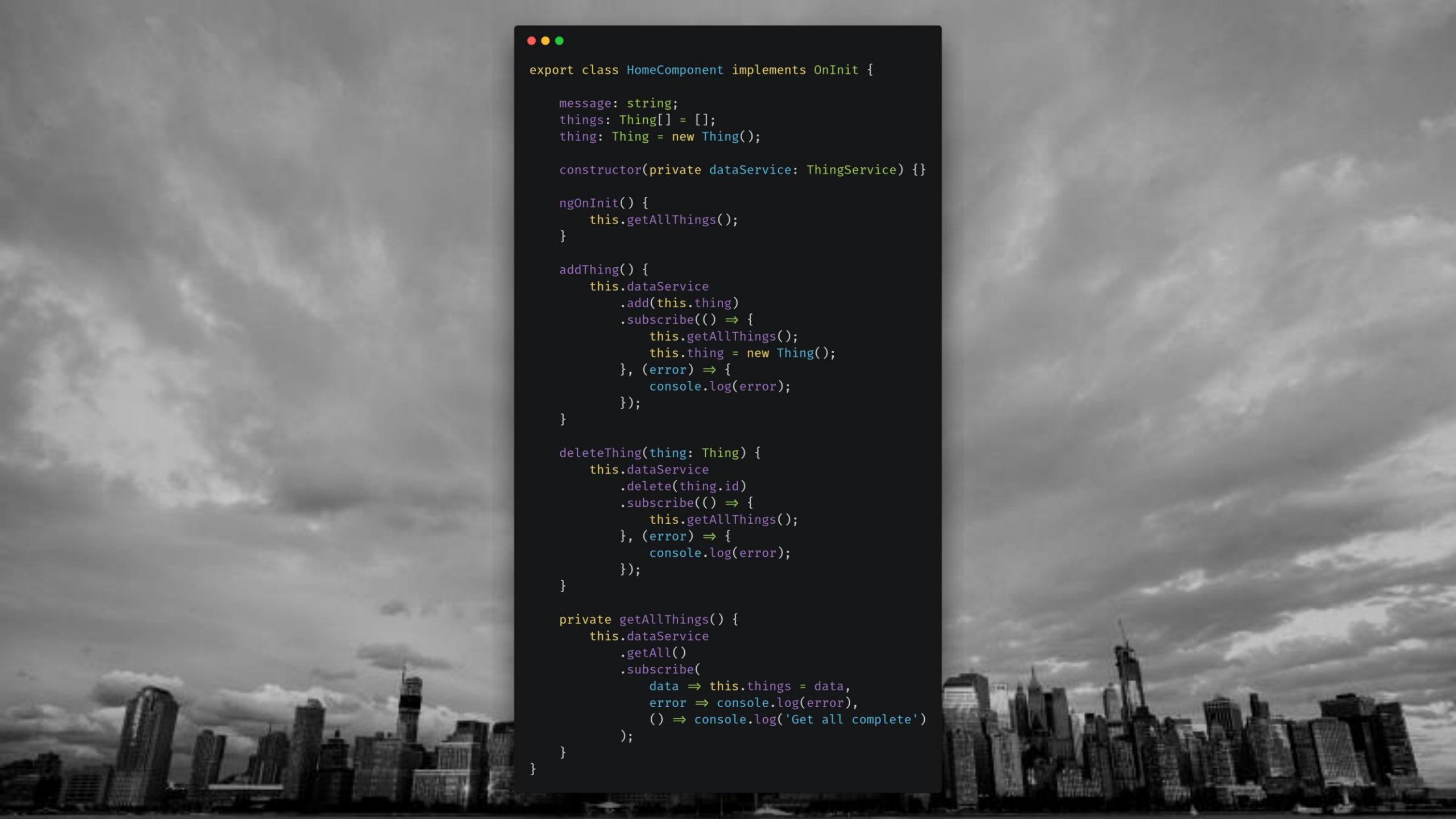
https://stackblitz.com/edit/angular-q3ruah


Core Module

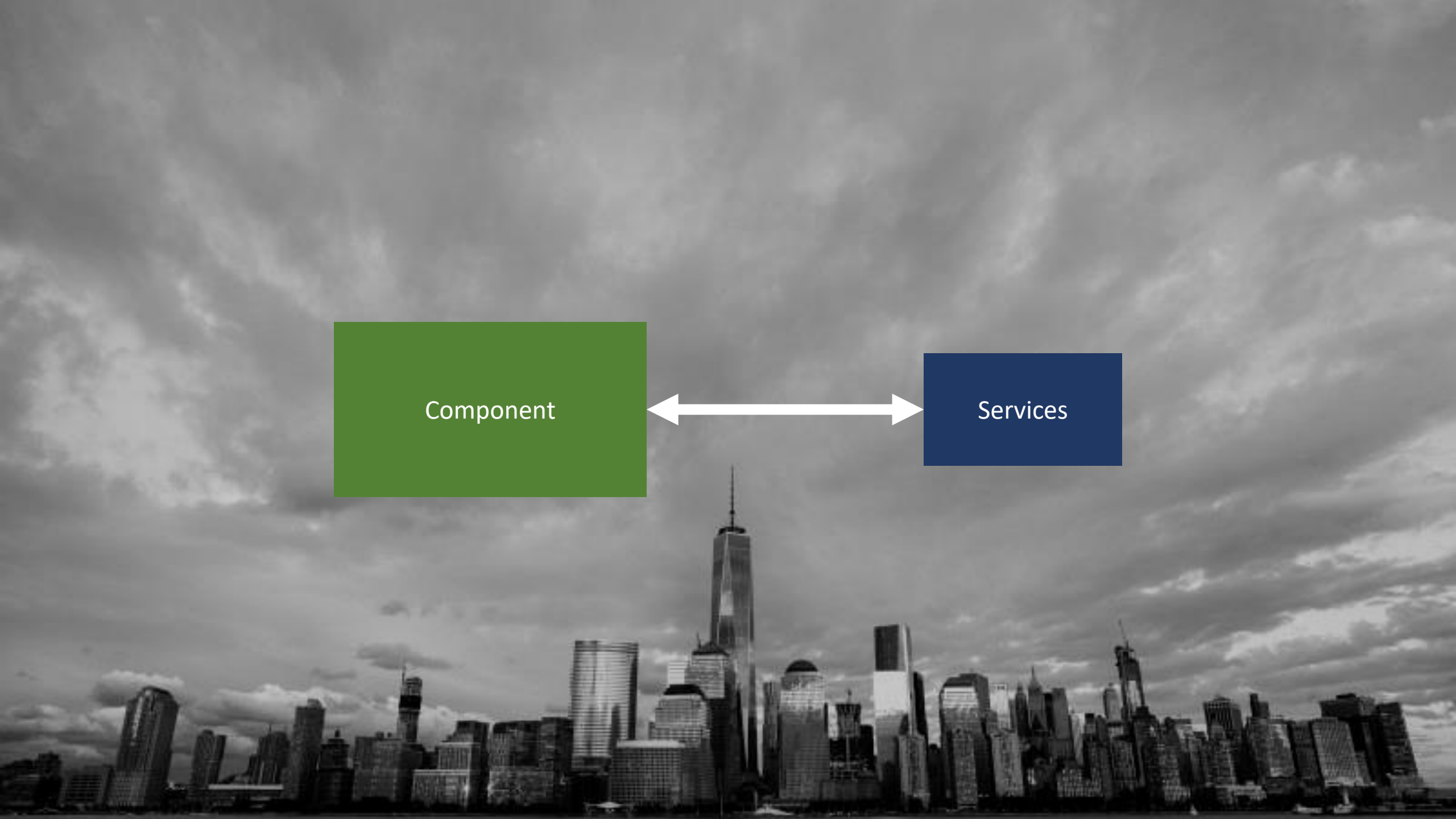
Shared Module


src
app
account
core
data-services
interceptors
services
store
core.module.ts
food
home
shared
components
configuration
guards
models
shared.module.ts
store

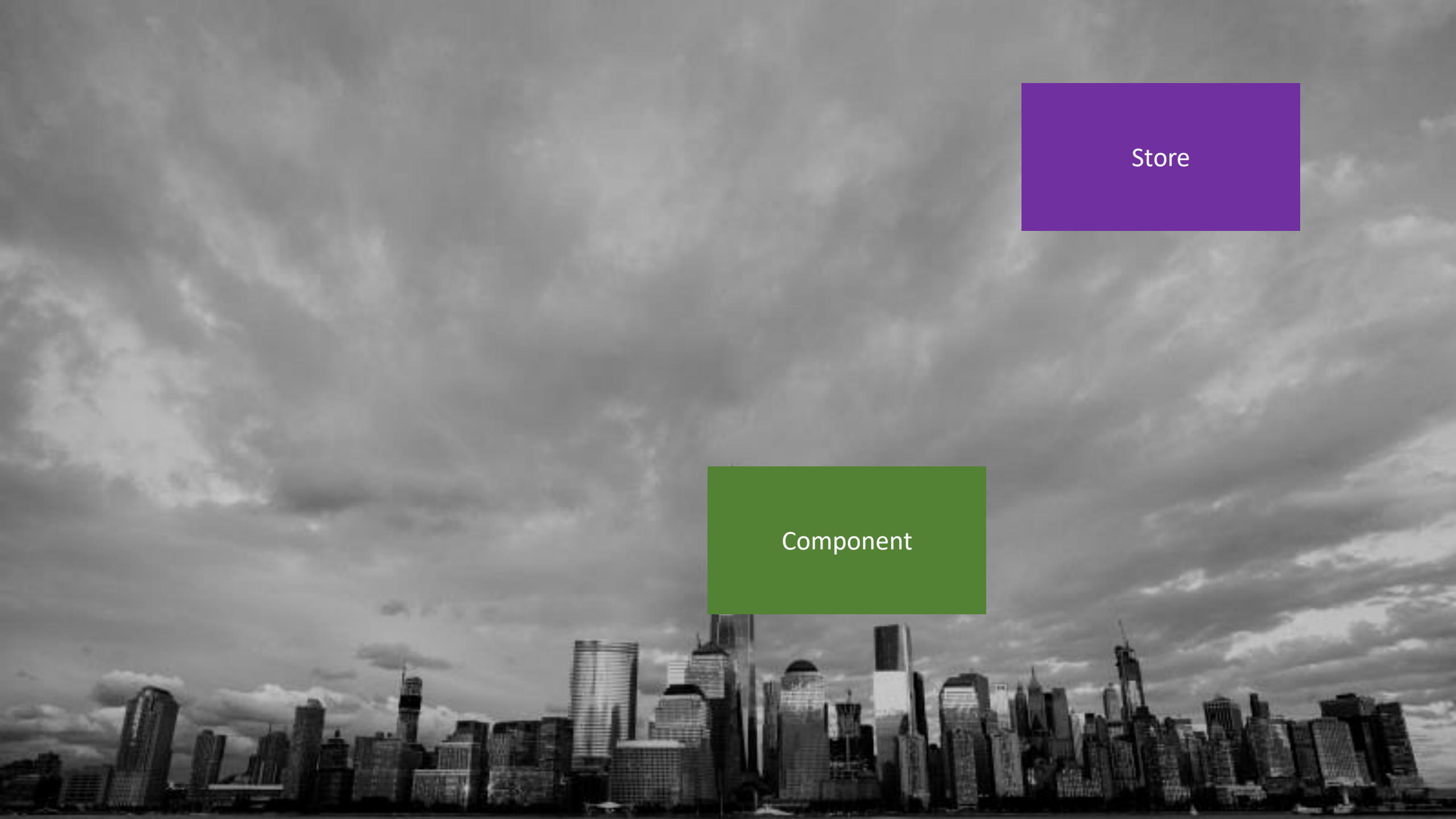

allThings: Thing[] = [];
getAllThings(): void {
this.dataService.getAll().subscribe(
data => {
this.allThings = data;
},
error => console.log(error)
);
}



```
export class HomeComponent implements OnInit {

    message: string;
    things: Thing[] = [];
    thing: Thing = new Thing();

    constructor(private dataService: ThingService) {}

    ngOnInit() {
        this.getAllThings();
    }

    addThing() {
        this.dataService
            .add(this.thing)
            .subscribe(() => {
                this.getAllThings();
                this.thing = new Thing();
            }, (error) => {
                console.log(error);
            });
    }

    deleteThing(thing: Thing) {
        this.dataService
            .delete(thing.id)
            .subscribe(() => {
                this.getAllThings();
            }, (error) => {
                console.log(error);
            });
    }

    private getAllThings() {
        this.dataService
            .getAll()
            .subscribe(
                data => this.things = data,
                error => console.log(error),
                () => console.log('Get all complete')
            );
    }
}
```


Component
Services


Component


Store
Component


export interface FoodState {
entities: FoodItem[];
loaded: boolean;
loading: boolean;
}
export const initialState: FoodState = {
entities: [],
loaded: false,
loading: false
};


@NgModule({
imports: [
StoreModule.forRoot({})
],
providers: [],
declarations: [AppComponent],
bootstrap: [AppComponent]
})
export class AppModule {}

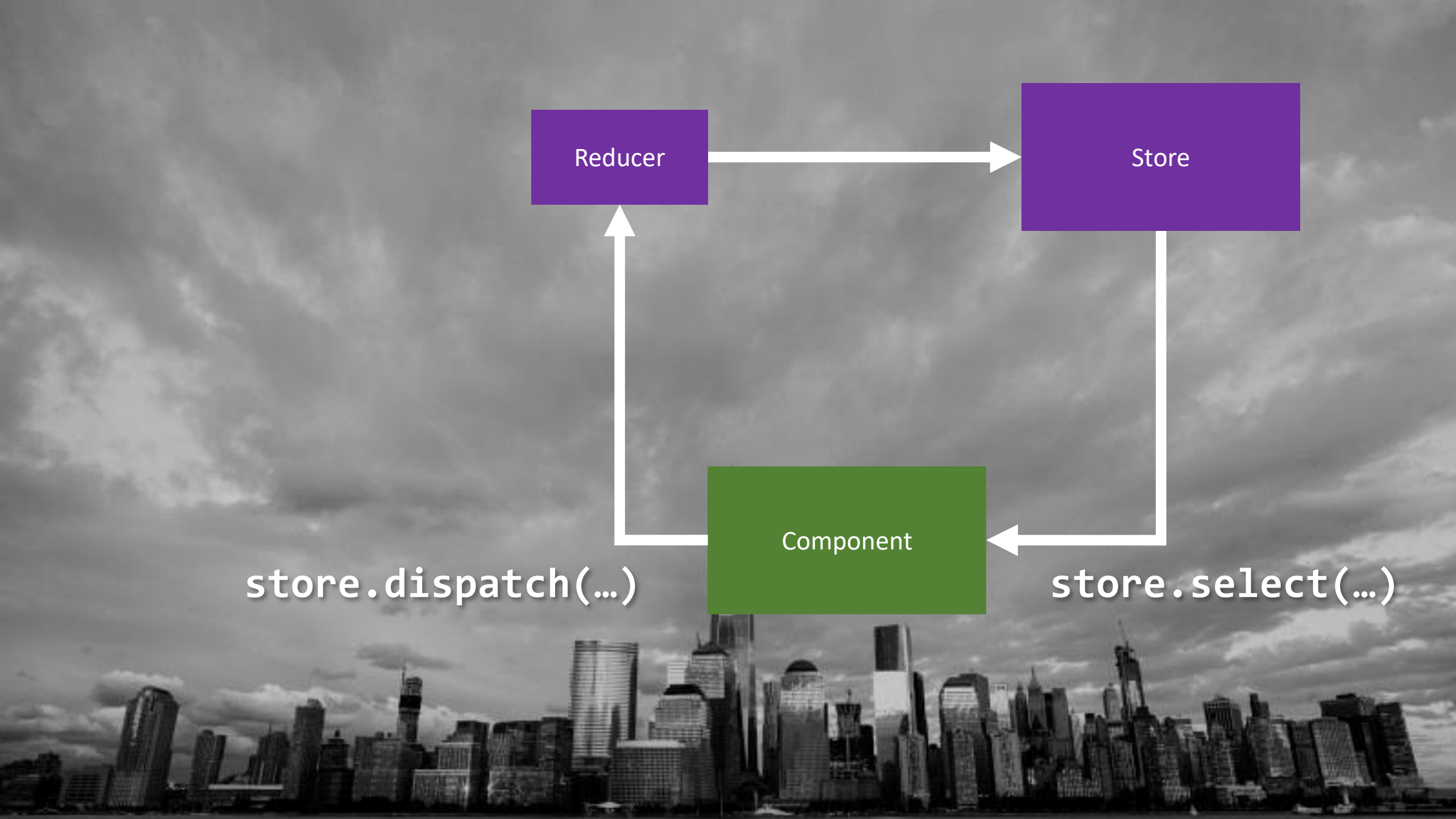

@NgModule({
imports: [
StoreModule.forFeature('food', reducers)
],
declarations: [
// ...
],
providers: [],
exports: []
})
export class FoodModule {}

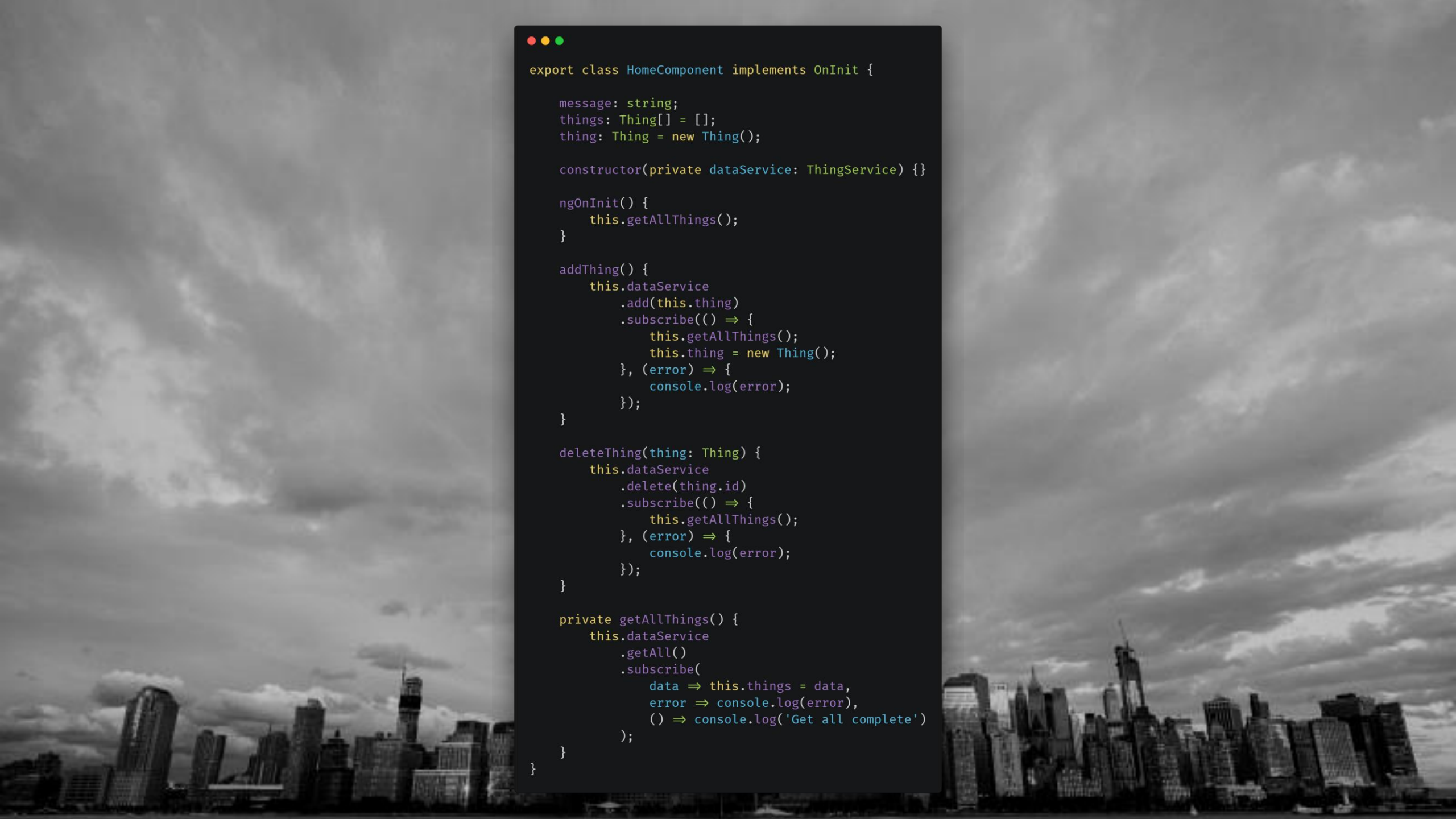

Reducer
Store
Component

```
export function foodItemsReducer(
  state = initialState,
  action: foodActions.FoodActions
): FoodState {
  switch (action.type) {
    case foodActions.ADD_FOOD_SUCCESS:
    case foodActions.UPDATE_FOOD_SUCCESS: {
      return {
        // a new state
      };
    }

    case foodActions.LOAD_FOOD_SUCCESS: {
      return {
        // a new state
      };
    }

    case foodActions.DELETE_FOOD_SUCCESS: {
      return {
        // a new state
      };
    }

    default:
      return state;
  }
}
```


Reducer
Store
Component


Reducer
Store
Component
store.select(…)

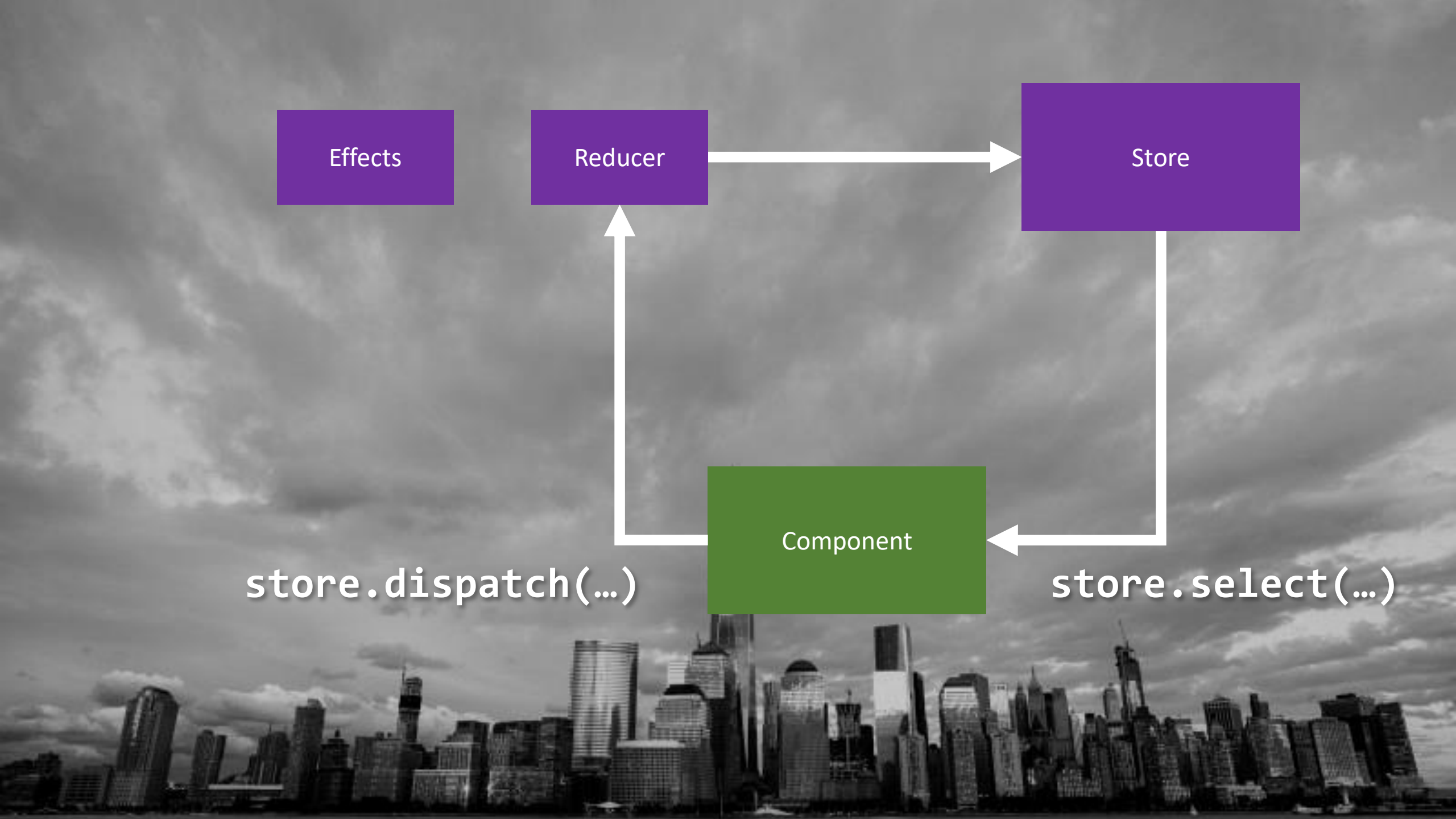

Reducer
Store
Component
store.dispatch(…)
store.select(…)



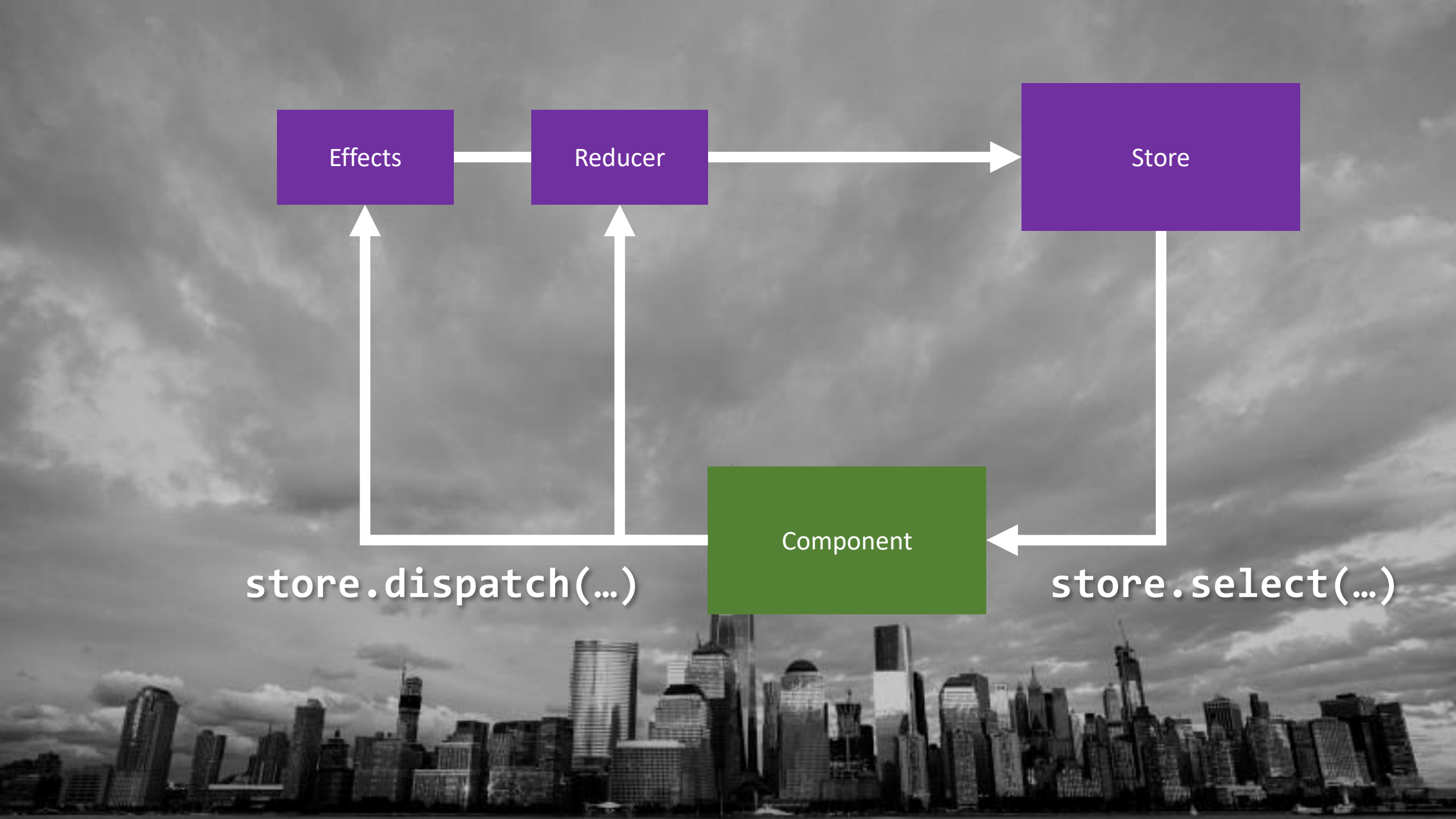
```
export class HomeComponent implements OnInit {

    message: string;
    things: Thing[] = [];
    thing: Thing = new Thing();

    constructor(private dataService: ThingService) {}

    ngOnInit() {
        this.getAllThings();
    }

    addThing() {
        this.dataService
            .add(this.thing)
            .subscribe(() => {
                this.getAllThings();
                this.thing = new Thing();
            }, (error) => {
                console.log(error);
            });
    }

    deleteThing(thing: Thing) {
        this.dataService
            .delete(thing.id)
            .subscribe(() => {
                this.getAllThings();
            }, (error) => {
                console.log(error);
            });
    }

    private getAllThings() {
        this.dataService
            .getAll()
            .subscribe(
                data => this.things = data,
                error => console.log(error),
                () => console.log('Get all complete')
            );
    }
}
```

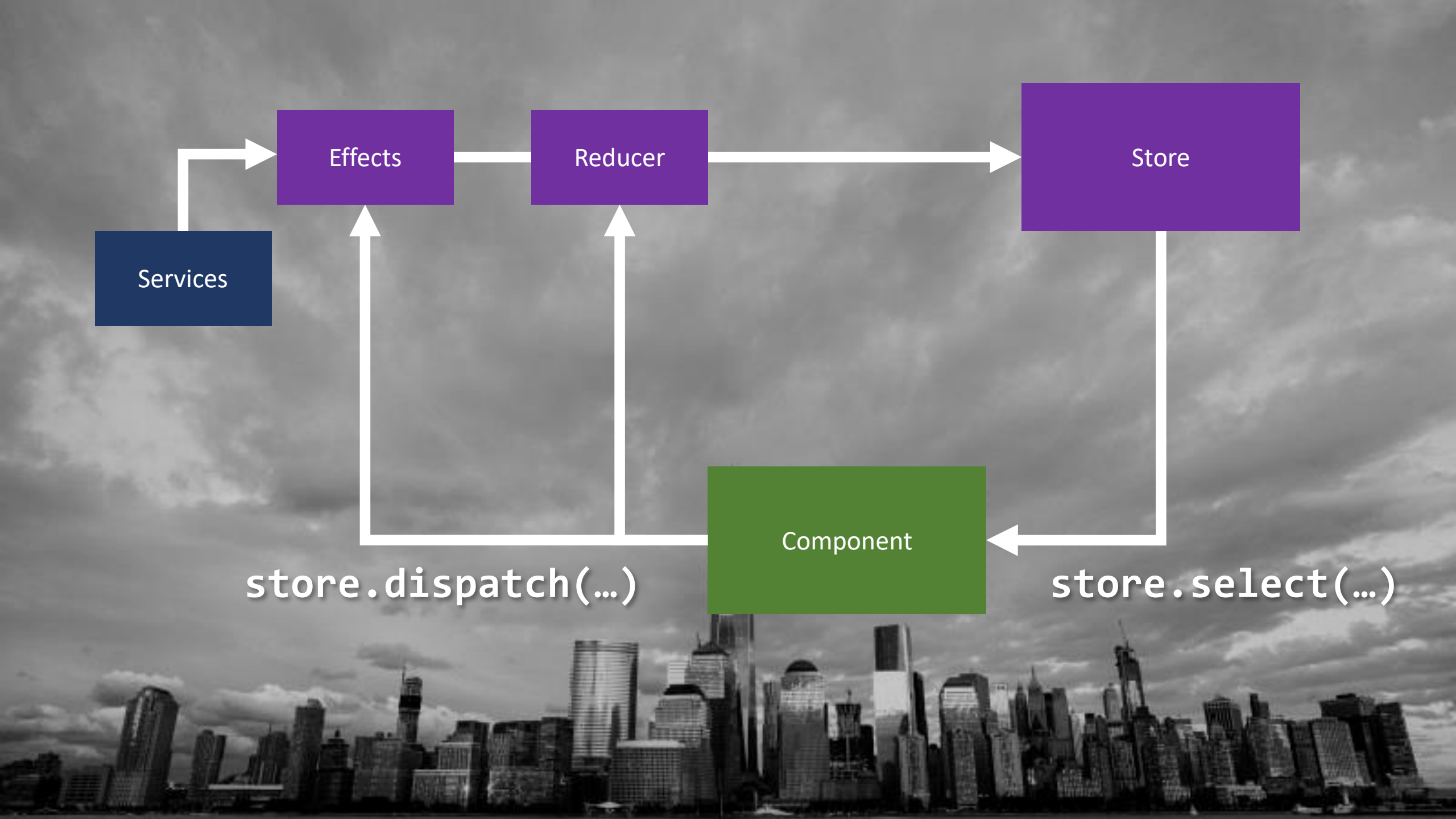
```
@Component({ /* ... */ })
export class MainFoodComponent implements OnInit {
  foods$: Observable<FoodItem[]>;

  constructor(private store: Store<fromStore.FoodState>) {}

  ngOnInit() {
    this.foods$ = this.store.select(fromStore.getAllFoods);
    this.store.dispatch(new fromStore.LoadFoodAction());
  }

  addFood(foodItem: FoodItem) {
    this.store.dispatch(new fromStore.AddFoodAction(foodItem));
  }

  updateFood(foodItem: FoodItem) {
    this.store.dispatch(new fromStore.UpdateFoodAction(foodItem));
  }

  deleteFood(foodItem: FoodItem) {
    this.store.dispatch(new fromStore.DeleteFoodAction(foodItem));
  }
}
```


Smart component
Pres. component
store.select(…)
@Input(…)
store.dispatch(…)
@Output(…)

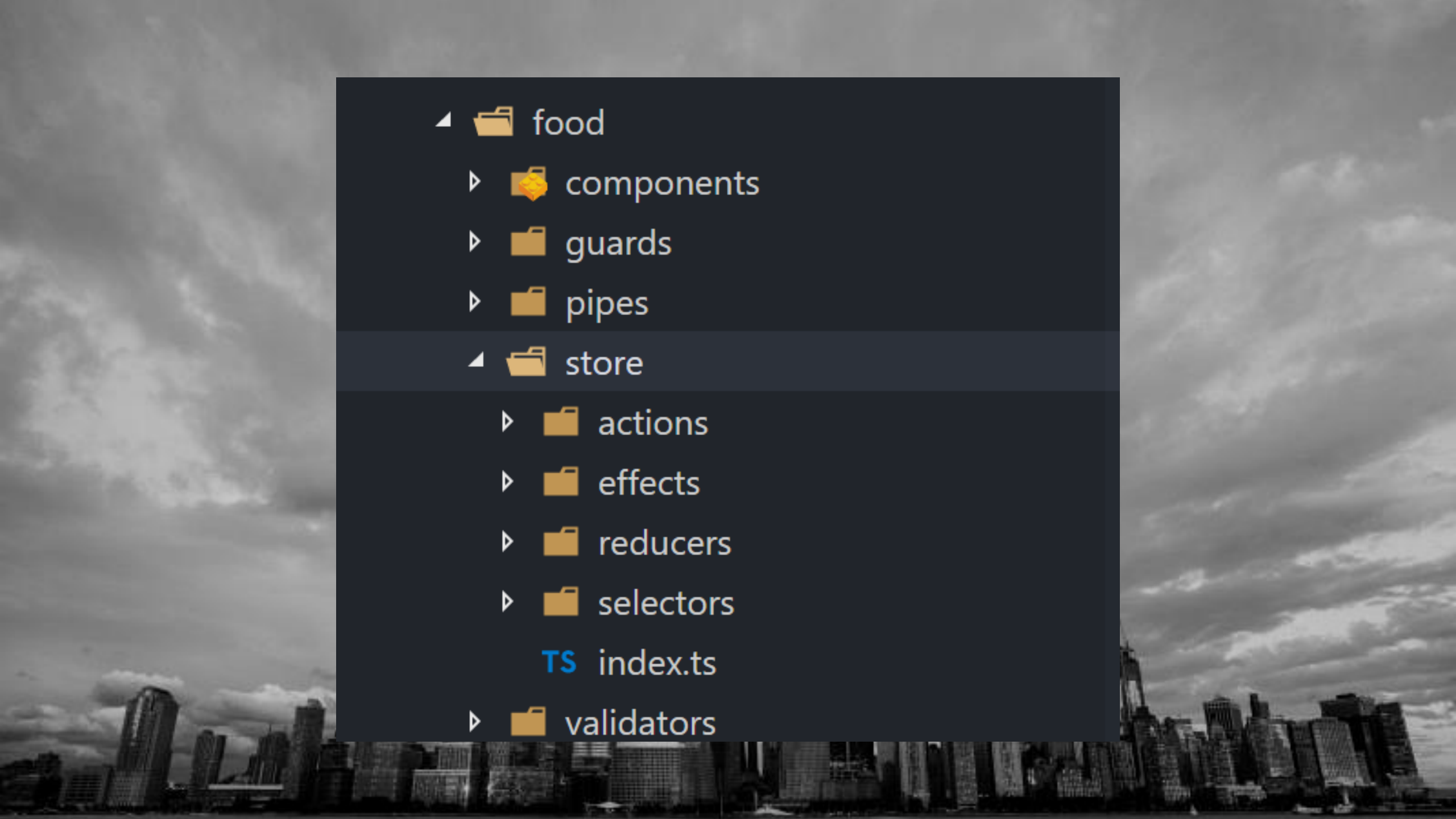

Effects
Reducer
Store
Component
store.dispatch(…)
store.select(…)


@NgModule({
imports: [
StoreModule.forFeature('food', reducers),
EffectsModule.forFeature(effects)
],
declarations: [
// ...
],
providers: [],
exports: []
})
export class FoodModule {}


Effects
Reducer
Store
Component
store.dispatch(…)
store.select(…)


Effects
Reducer
Store
Services
Component
store.dispatch(…)
store.select(…)

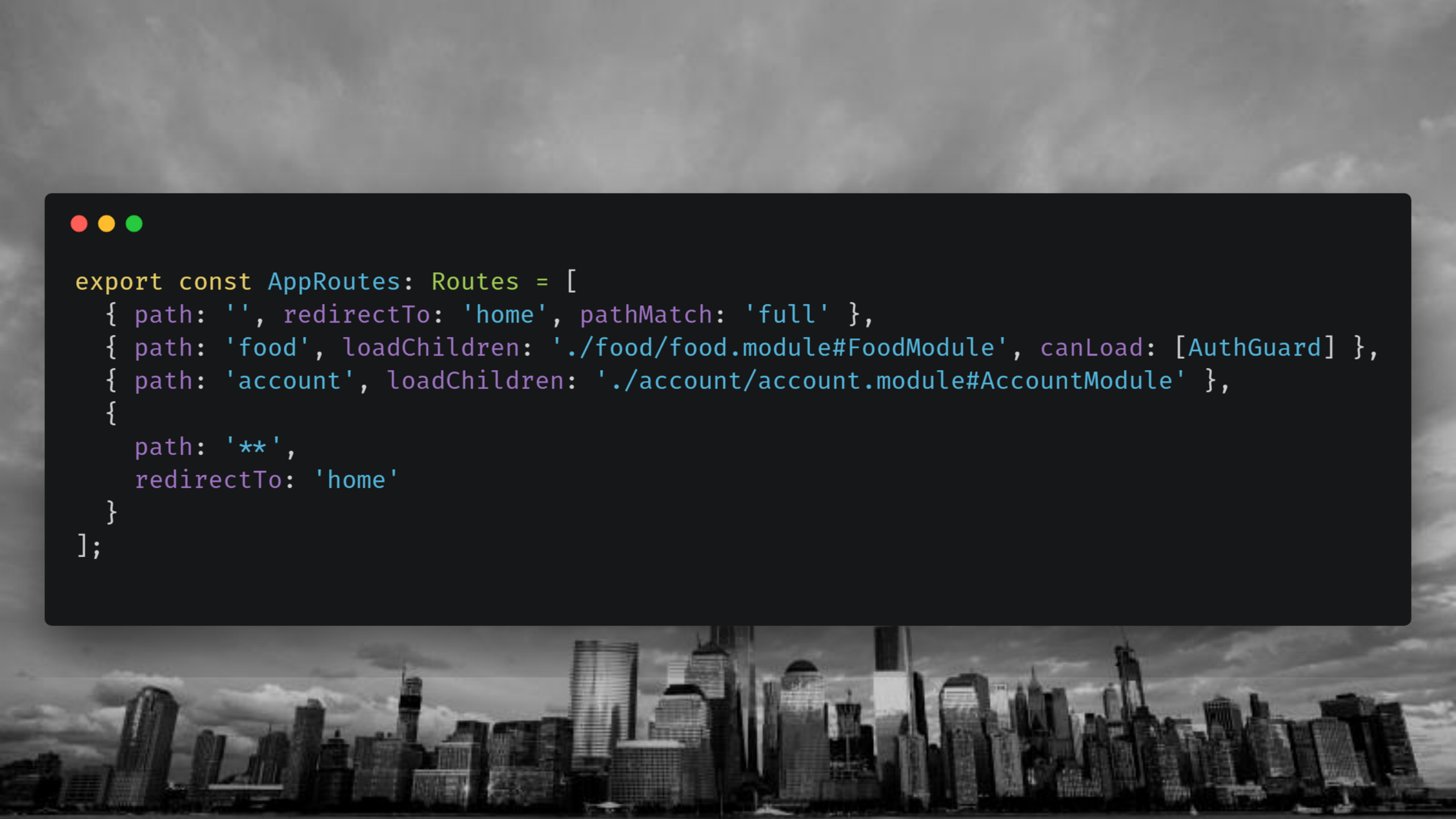

Ben Lesh
@BenLesh
Following
Oh.. and on the anti-redux-because-boilerplate front:
That "boilerplate" is directly the result of separation of concerns: You have to define many separate things. It's more than you're used to, because you weren't separating concerns before when your app was a big pile of code.
3:44 AM - 13 Feb 2018
26 Retweets 108 Likes
6
26
108
Tweet your reply
Ben Lesh @BenLesh · Feb 13
Tangling state and rendering together into one big, freaky component, or many tiny, cute, little, freaky components doesn't mean you've separated concerns.
2
3
25


food
components
guards
pipes
store
actions
effects
reducers
selectors
TS
index.ts
validators


export * from './selectors';
export * from './reducers';
export * from './actions';
export * from './effects';

```
import * as fromStore from '../../store';

@Component({ /* ... */ })
export class MainFoodComponent implements OnInit {
  constructor(private store: Store<fromStore.FoodState>) {}

  ngOnInit() {
    this.foods$ = this.store.select(fromStore.getAllFoods);
    this.store.dispatch(new fromStore.LoadFoodAction());
  }

  addFood(foodItem: FoodItem) {
    this.store.dispatch(new fromStore.AddFoodAction(foodItem));
  }

  updateFood(foodItem: FoodItem) {
    this.store.dispatch(new fromStore.UpdateFoodAction(foodItem));
  }

  deleteFood(foodItem: FoodItem) {
    this.store.dispatch(new fromStore.DeleteFoodAction(foodItem));
  }
}
```


Lazy Loading



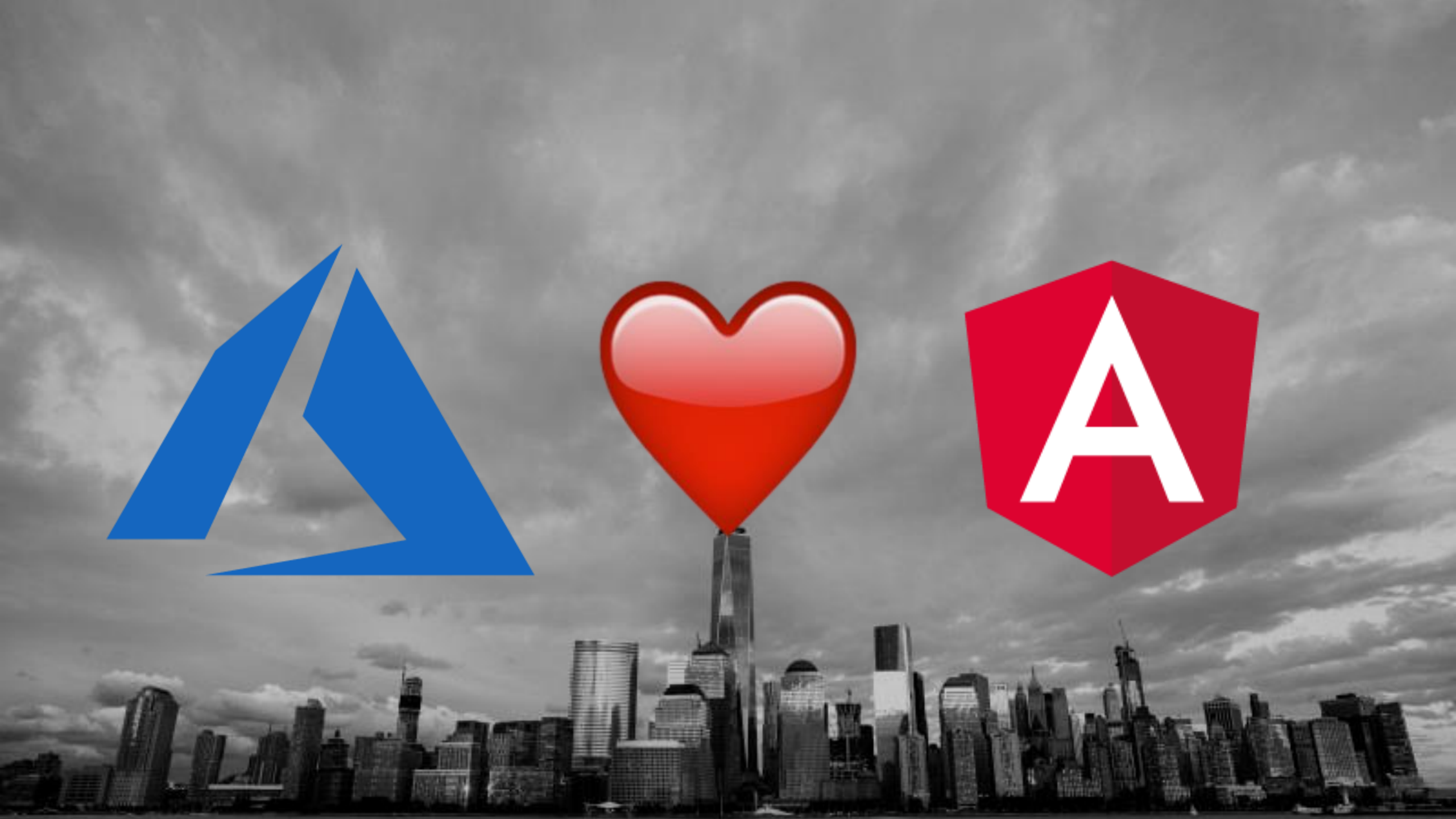
```
export const AppRoutes: Routes = [
  { path: '', redirectTo: 'home', pathMatch: 'full' },
  { path: 'food', loadChildren: './food/food.module#FoodModule', canLoad: [AuthGuard] },
  { path: 'account', loadChildren: './account/account.module#AccountModule' },
  {
    path: '**',
    redirectTo: 'home'
  }
];
```


Do
a great
production build


Do
treeshaking


Do
ahead of time
compilation

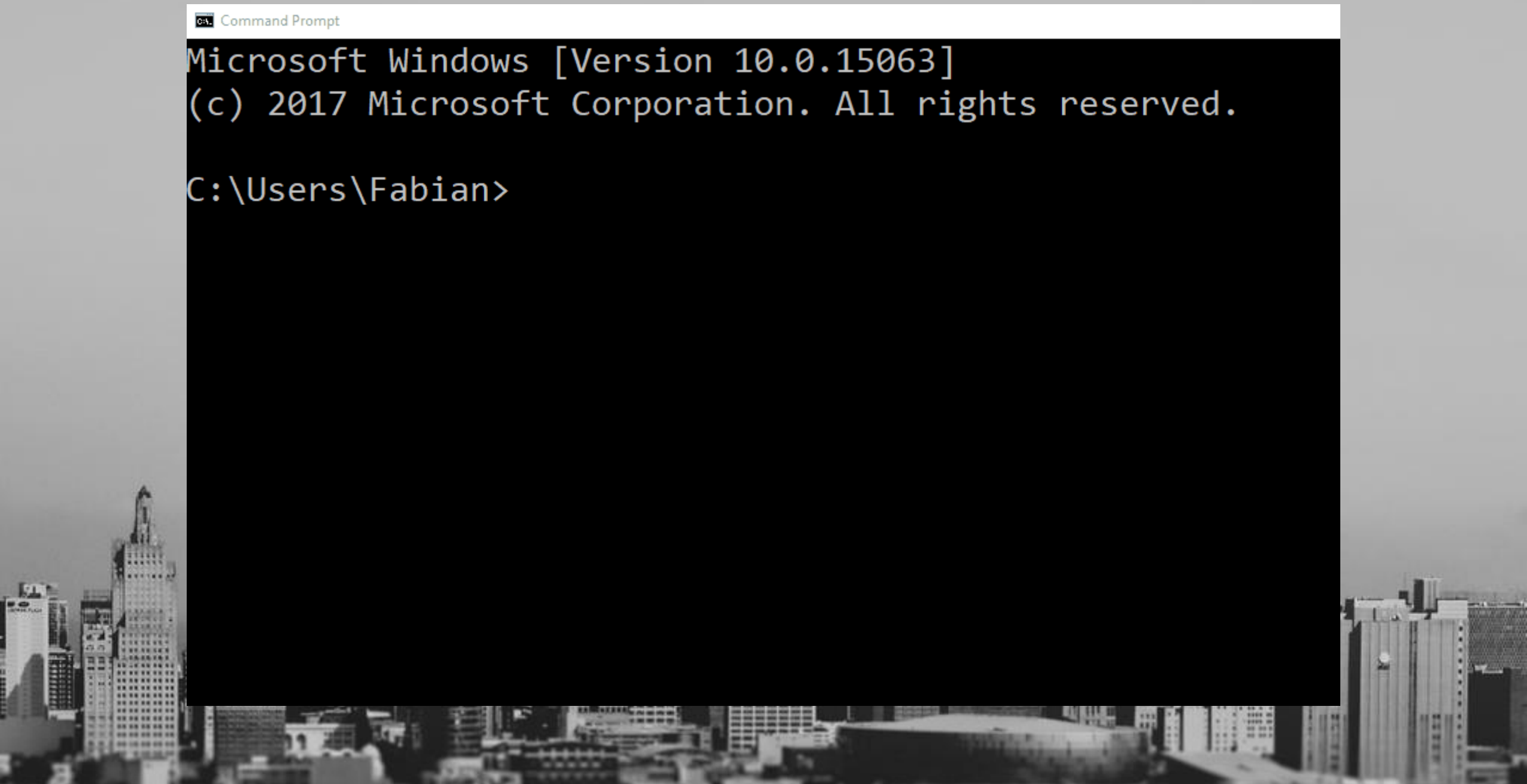

> ng build --prod




Demo


Change
your code


Automate
everything


Command Prompt
Microsoft Windows [Version 10.0.15063]
(c) 2017 Microsoft Corporation. All rights reserved.
C:\Users\Fabian>


Le Fin


@FabianGosebrink
OFFERING SOLUTIONS SOFTWARE
A<A>EMY
https://offering.solutions
https://angular-academy.ch
https://swissangular.com
https://github.com/FabianGosebrink/ASPNETCore-Angular-Ngrx