Sprint 7 Subrules

Improving usability of entering large amounts of fixed rules

Dynamic Rule Locations

A dynamic rule may optionally have a Departure and Destination location

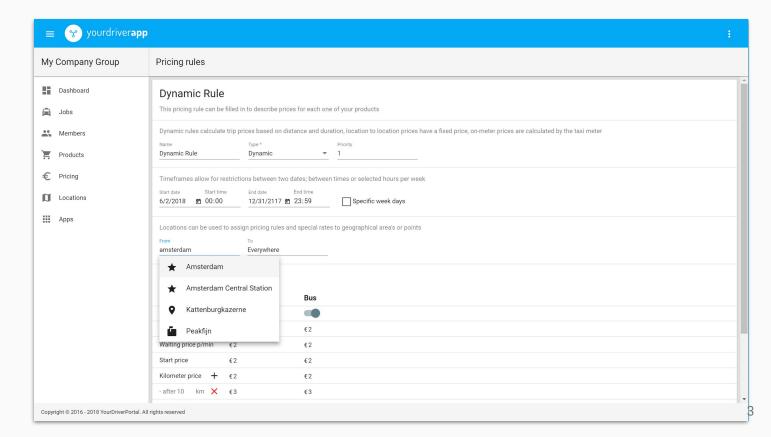
If a location is left empty, the location is assumed to be 'Everywhere'

When the user starts typing in the location field, suggestions of already created locations are displayed, including locations created by developers, which are available to all users

If no suggestion is found, but the user enters the value, a propt asks the user to create a new location

Dynamic Rule Locations

Search suggestions are based on the location name and address



Fixed Rule Locations

A fixed / location to location rule can have multiple subrules to speed up creation of each price definition, having:

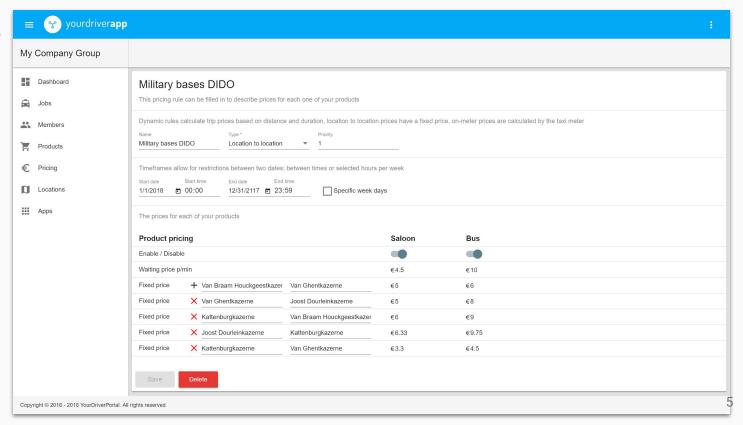
- 1. Departure
- 2. Destination
- 3. Fixed price for each product

Departure and Destination are required properties

Fixed Rule Locations

Like the threshold rows in the dynamic price rules, sub rules can be added by clicking the plus sign

A fixed rule must have both departure and destination locations defined



On-meter

A change to estimations and on-meter prices is bringing the settings to the Apps view

If a location is not provided by the bookings app, no price can be calculated, and therefore users must be able to define whether they would like prices to be calculated in a later stage

The table in the next slide shows eight situations, whether products are shown, whether prices are calculated or estimated, and if meter prices can be used

	= on		= off		= impossible
--	------	--	-------	--	--------------

show - products are shown, regardless of whether prices have been calculated for them

prices - prices have been calculated

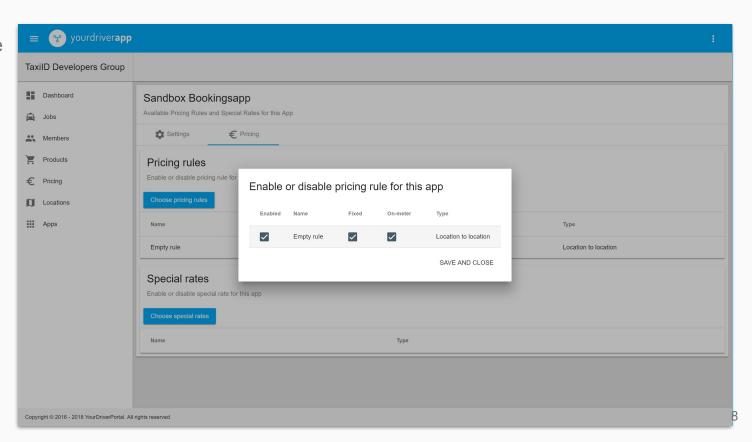
estimation - prices have not been calculated

on-meter - allow app to determine the price on meter

Both locations provided	Is fixed	Is on-meter	
show →	prices	on-meter	
show →	prices		
show →	estimation →	on-meter	
show			
show			

Estimations Select

If a rule is enabled, the fixed and/or the on-meter checkbox must be enabled



Authorization

A token is requested from the core API whenever group routes are being accessed

This token is stored in localstorage

The token is sent to TPS on every request

TPS limits the user's access to resources that are not bound to the JWT payload identity

Authorization - Requesting JWT

```
canActivate(
 route: ActivatedRouteSnapshot
): Observable<boolean> | Promise<boolean> | boolean {
  // Identity information from url and local storage
  const companyId = route.params.id;
  const daAppInstallId = this.getFromStorage(
   vaultPrefix + '.driver',
    'daAppInstallId'
```

```
// See if JWT exists, or request a new one
const jwtName = jwtKeyName(companyId);
if (this.itemIsSet(jwtName)) return true;

// Request new JWT and return the status as a boolean
return this._companyService.getNewJWT(companyId, daAppInstallId)
    .map(response => {
        this._vault.setItem(jwtName, 'Bearer ' + response.jwt);
        return this.itemIsSet(jwtName);
    })
    .catch(() => Observable.of(false));
```

Authorization - Sending JWT in Headers

```
getJWTHeaders = (): HttpHeaders => {
  const companyId =
        this. router.routerState.snapshot.root.firstChild.params['id'];
  const bearer = this._vault.getItem([
    environment.vaultPrefix,
    companyId,
    'jwt',
  ].join('.') || 'Bearer ');
  return new HttpHeaders()
    .set('Accept', 'application/json')
    .set('Authorization', bearer);
```

Authorization - Authorizing Request

The JWT payload id's are added to the query filters to only allow certain resources to be accessible

```
if (!ctx.args.filter) ctx.args.filter = {};
if (!ctx.args.filter.where) ctx.args.filter.where = {};
if (relatedToCompany) {
  if (ctx.args.data) ctx.args.data.companyId = companyId;
  ctx.args.filter.where.companyId = companyId;
// If belongsTo daAppInstall, force JWT token daAppInstallId
if (relatedToDaAppInstall) {
  if (ctx.args.data) ctx.args.data.daAppInstallId = daAppInstallId;
  ctx.args.filter.where.daAppInstallId = daAppInstallId;
```