

# Api Documentation

## Overview

Api Documentation

## Version information

*Version : 1.0*

## License information

*License : Apache 2.0*

*License URL : <http://www.apache.org/licenses/LICENSE-2.0>*

*Terms of service : urn:tos*

## URI scheme

*Host : localhost:8080*

*BasePath : /*

## Tags

- active-drivers : Active Drivers
- active-drivers-administration : Active Drivers Administration
- authentication : Rest
- cache-endpoint : Cache Endpoint
- campaigns : Campaigns
- campaigns-administration : Campaigns Administration
- car-photos : Car Photos
- car-photos-common : Car Photos Common
- car-types : Car Types
- cards : Cards
- charities : Charities
- cities : Cities
- configs : Configs
- configs-administration : Configs Administration
- configuration-items : Configuration Items
- custom-payments : Custom Payments

- driver-area-queue : Driver Area Queue
- driver-area-queue-administration : Driver Area Queue Administration
- driver-area-queue-common : Driver Area Queue Common
- driver-cars : Driver Cars
- driver-changes : Driver Changes
- driver-email-reminders : Driver Email Reminders
- driver-reports : Driver Reports
- driver-reports-common : Driver Reports Common
- driver-statistics : Driver Statistics
- driver-types : Driver Types
- drivers : Drivers
- drivers-administration : Drivers Administration
- drivers-common : Drivers Common
- drivers-data : Drivers Data
- drivers-documents : Drivers Documents
- drivers-documents-administration : Drivers Documents Administration
- events : Events
- fare-payments : Fare Payments
- forgot-password : Forgot Password
- geolocation-logs : Geolocation Logs
- lost-and-found : Lost And Found
- lost-and-found-administration : Lost And Found Administration
- phone-verification : Phone Verification
- promocodes : Promocodes
- push-notifications : Push Notifications
- rating-updates : Rating Updates
- rating-updates-administration : Rating Updates Administration
- reports : Reports
- ride-cancellation-feedback : Ride Cancellation Feedback
- ride-flow : Ride Flow
- ride-flow-administration : Ride Flow Administration
- ride-reports : Ride Reports
- ride-share-tracking : Ride Share Tracking
- ride-upgrades : Ride Upgrades
- rider-payments : Rider Payments

- rider-promocodes : Rider Promocodes
- riders : Riders
- riders-administration : Riders Administration
- riders-common : Riders Common
- rides : Rides
- rides-administration : Rides Administration
- split-fares : Split Fares
- support : Support
- support-topics : Support Topics
- support-topics-administration : Support Topics Administration
- surge-areas : Surge Areas
- surge-areas-administration : Surge Areas Administration
- tokens : Tokens
- user-photos : User Photos
- user-tracking-stats : User Tracking Stats
- users : Users

## Paths

**Endpoint to be called externally when an user gets reset email**

```
GET /password-reset{?token}
```

### Parameters

Type	Name	Description	Schema
Query	<b>token required</b>	token	string

### Responses

HTTP Code	Description	Schema
200	OK	No Content

HTTP Code	Description	Schema
302	Redirect user to landing page	No Content

## Produces

- \*/\*

## Tags

- forgot-password

## Security

Type	Name	Scopes
basic	basic	scope

## Go offline as a driver

```
DELETE /rest/acdr
```

## Responses

HTTP Code	Description	Schema
200	Driver successfully went offline	No Content
400	Driver tried to go offline while in a ride	No Content

## Produces

- \*/\*

## Tags

- active-drivers

## Security

Type	Name	Scopes
basic	basic	scope

# Get current online driver information as a driver

```
GET /rest/acdr/current
```

## Responses

HTTP Code	Description	Schema
200	Current driver object	<a href="#">CurrentActiveDriverDto</a>
204	Driver is offline	<a href="#">CurrentActiveDriverDto</a>
400	Driver has more than 2 rides assigned simultaneously	No Content
403	User is not a driver	No Content

## Produces

- application/json

## Tags

- active-drivers

## Security

Type	Name	Scopes
basic	<a href="#">basic</a>	scope

# Force end driver offline

```
DELETE /rest/acdr/{id}
```

## Parameters

Type	Name	Description	Schema
Path	<b>id</b> <i>optional</i>	Active Driver ID	integer (int64)

## Responses

HTTP Code	Description	Schema
200	Driver is forced to go offline	No Content
400	Driver is in a ride, so it's not possible to send offline	No Content

## Produces

- \*/\*

## Tags

- active-drivers-administration

## Security

Type	Name	Scopes
basic	basic	scope

## List online drivers for web admin console

```
GET /rest/acdr?avatarType=ADMIN{&cityId,desc,page,pageSize,sort}
```

## Parameters

Type	Name	Description	Schema	Default
Query	avatarType <i>required</i>		enum (ADMIN)	"ADMIN"
Query	cityId <i>optional</i>	City ID to search online drivers in	integer (int64)	
Query	desc <i>optional</i>	Order the result descending	boolean	
Query	page <i>optional</i>	Page number	integer (int32)	

Type	Name	Description	Schema	Default
Query	<b>pageSize</b> <i>optional</i>	Page size	integer (int32)	
Query	<b>sort</b> <i>optional</i>	Columns to sort the result by, comma-separated list	< string array(multi) >	

## Responses

HTTP Code	Description	Schema
200	OK	Page«ActiveDriver Dto»

## Produces

- application/json

## Tags

- active-drivers-administration

## Security

Type	Name	Scopes
basic	<b>basic</b>	scope

## List online drivers for 3rd party API clients

```
GET /rest/acdr?avatarType=API_CLIENT{&desc,page,pageSize,sort}
```

## Parameters

Type	Name	Description	Schema	Default
Query	<b>avatarType</b> <i>required</i>		enum (API_CLIENT)	"API_CLIENT"
Query	<b>desc</b> <i>optional</i>	Order the result descending	boolean	

Type	Name	Description	Schema	Default
Query	<b>page</b> <i>optional</i>	Page number	integer (int32)	
Query	<b>pageSize</b> <i>optional</i>	Page size	integer (int32)	
Query	<b>sort</b> <i>optional</i>	Columns to sort the result by, comma-separated list	< string > array(multi)	

## Responses

HTTP Code	Description	Schema
200	OK	< <a href="#">ActiveDriverLocationDto</a> > array

## Produces

- [application/json](#)

## Tags

- active-drivers-administration

## Security

Type	Name	Scopes
basic	<a href="#">basic</a>	scope

## Search for closest drivers as a driver

```
GET /rest/acdr?avatarType=DRIVER{&cityId,latitude,longitude}
```

## Parameters

Type	Name	Description	Schema	Default
Query	<b>avatarType</b> <i>required</i>		enum (DRIVER)	<a href="#">"DRIVER"</a>

Type	Name	Description	Schema	Default
Query	<b>cityId</b> <i>required</i>	City to look drivers in	integer (int64)	1
Query	<b>latitude</b> <i>required</i>	Current GPS latitude	number (double)	
Query	<b>longitude</b> <i>required</i>	Current GPS longitude	number (double)	

## Responses

HTTP Code	Description	Schema
200	List of driver objects	< <a href="#">CurrentActiveDriverDto</a> > array

## Produces

- [application/json](#)

## Tags

- active-drivers

## Security

Type	Name	Scopes
basic	<a href="#">basic</a>	scope

## Go online as a driver

```
POST  
/rest/acdr{?carCategories,carId,cityId,driverTypes,heading,latitude,longitude,speed}
```

## Parameters

Type	Name	Description	Schema	Default
Query	<b>carCategories</b> <i>required</i>	Comma-separated list of enabled car categories	< string array(multi) >	"REGULAR"
Query	<b>carId</b> <i>optional</i>	Selected car ID	integer (int64)	
Query	<b>cityId</b> <i>required</i>	Selected city ID	integer (int64)	1
Query	<b>driverTypes</b> <i>optional</i>	Comma-separated list of enabled driver types	< string array(multi) >	
Query	<b>heading</b> <i>optional</i>	Current GPS heading	number (double)	
Query	<b>latitude</b> <i>required</i>	Current GPS latitude	number (double)	
Query	<b>longitude</b> <i>required</i>	Current GPS longitude	number (double)	
Query	<b>speed</b> <i>optional</i>	Current GPS speed	number (double)	

## Responses

HTTP Code	Description	Schema
200	Location of going online	<a href="#">LatLng</a>
400	Driver is ineligible to go online using parameters passed	No Content
404	Unknown city ID is passed as a parameter	No Content
412	Driver has not accepted usage terms	No Content
500	Server is misconfigured, further processing is impossible	No Content

## Consumes

- [application/json](#)

## Produces

- application/json

## Tags

- active-drivers

## Security

Type	Name	Scopes
basic	basic	scope

## Update location as a driver

```
PUT  
/rest/acdr{?carCategories,course,driverTypes,heading,latitude,longitude,sequence,speed  
}
```

## Parameters

Type	Name	Description	Schema	Default
Query	<b>carCategories</b> <i>required</i>	Comma-separated list of enabled car categories	< string array(multi) >	"REGULAR"
Query	<b>course</b> <i>optional</i>	Current GPS course	number (double)	
Query	<b>driverTypes</b> <i>optional</i>	Comma-separated list of enabled driver types	< string array(multi) >	
Query	<b>heading</b> <i>optional</i>	Current GPS heading	number (double)	
Query	<b>latitude</b> <i>required</i>	Current GPS latitude	number (double)	
Query	<b>longitude</b> <i>required</i>	Current GPS longitude	number (double)	

Type	Name	Description	Schema	Default
Query	<b>sequence</b> <i>optional</i>	Current device timestamp. Should be sent only when ride is started. Used for ride tracking	integer (int64)	
Query	<b>speed</b> <i>optional</i>	Current GPS speed	number (double)	

## Responses

HTTP Code	Description	Schema
200	Updated location	<a href="#">LatLang</a>
400	Driver is ineligible to be online using parameters passed	No Content
409	Driver is already offline	No Content
500	Server is misconfigured, further processing is impossible	No Content

## Consumes

- [application/json](#)

## Produces

- [application/json](#)

## Tags

- active-drivers

## Security

Type	Name	Scopes
basic	<a href="#">basic</a>	scope

## Search for closest drivers eligible to driver given category and driver type as a rider

```
GET /rest/acdr{?carCategory,cityId,driverType,latitude,longitude}
```

## Parameters

Type	Name	Description	Schema	Default
Query	<b>carCategory</b> <i>required</i>	Requested car category	string	"REGULAR" "
Query	<b>cityId</b> <i>required</i>	City to look drivers in	integer (int64)	1
Query	<b>driverType</b> <i>optional</i>	Requested driver type	string	
Query	<b>latitude</b> <i>required</i>	Current GPS latitude	number (double)	
Query	<b>longitude</b> <i>required</i>	Current GPS longitude	number (double)	

## Responses

HTTP Code	Description	Schema
200	List of driver objects	< <a href="#">CurrentActiveDriverDto</a> > array
401	User is not authenticated	No Content
403	User is not a rider	No Content
404	Current rider is not found	No Content

## Produces

- [application/json](#)

## Tags

- active-drivers

## Security

Type	Name	Scopes
basic	basic	scope

## Reload server caches, available only for admin

POST /rest/cache/reload

### Responses

HTTP Code	Description	Schema
200	OK	No Content

### Consumes

- application/json

### Produces

- \*/\*

### Tags

- cache-endpoint

### Security

Type	Name	Scopes
basic	basic	scope

## Get information on campaigns belonging to a given provider

GET /rest/campaigns/providers/{id}/campaigns

### Parameters

Type	Name	Description	Schema
Path	<b>id</b> <i>optional</i>	Campaign provider ID	integer (int64)

## Responses

HTTP Code	Description	Schema
<b>200</b>	OK	< CampaignDto > array

## Produces

- \*/\*

## Tags

- campaigns

## Security

Type	Name	Scopes
basic	<b>basic</b>	scope

# Get information on campaign providers

```
GET /rest/campaigns/providers{?cityId}
```

## Parameters

Type	Name	Description	Schema	Default
Query	<b>cityId</b> <i>required</i>	Campaign ID	integer (int64)	1

## Responses

HTTP Code	Description	Schema
200	OK	< CampaignProvide rDto > array

## Produces

- \*/\*

## Tags

- campaigns

## Security

Type	Name	Scopes
basic	basic	scope

## List campaigns that the rider is subscribed to

```
GET /rest/campaigns/{?for}
```

## Parameters

Type	Name	Description	Schema
Query	<b>for</b> <i>required</i>	Rider ID	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	< CampaignSubscriptionDto > array

## Produces

- \*/\*

## Tags

- campaigns-administration

## Security

Type	Name	Scopes
basic	basic	scope

## Get information on given campaign

```
GET /rest/campaigns/{id}
```

## Parameters

Type	Name	Description	Schema
Path	<b>id</b> <i>optional</i>	Campaign ID	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	CampaignDto
404	Campaign is not found	No Content

## Produces

- \*/\*

## Tags

- campaigns

## Security

Type	Name	Scopes
basic	basic	scope

# List riders that are subscribed to a campaign

```
GET /rest/campaigns/{id}/subscribers
```

## Parameters

Type	Name	Description	Schema
Path	<b>id</b> <i>optional</i>	Campaign ID	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	< CampaignRiderDt o > array

## Produces

- \*/\*

## Tags

- campaigns-administration

## Security

Type	Name	Scopes
basic	<a href="#">basic</a>	scope

# Unsubscribe a rider from a campaign

```
DELETE /rest/campaigns/{id}/subscribers/{riderId}
```

## Parameters

Type	Name	Description	Schema
Path	<b>id</b> <i>optional</i>	Campaign ID	integer (int64)
Path	<b>riderId</b> <i>optional</i>	Rider ID	integer (int64)

## Responses

HTTP Code	Description	Schema
202	Accepted	No Content

## Produces

- \*/\*

## Tags

- campaigns-administration

## Security

Type	Name	Scopes
basic	<b>basic</b>	scope

## Subscribe a rider to a campaign

```
POST /rest/campaigns/{id}/subscribers{?riderId}
```

## Parameters

Type	Name	Description	Schema
Path	<b>id</b> <i>optional</i>	Campaign ID	integer (int64)
Query	<b>riderId</b> <i>required</i>	Rider ID	integer (int64)

## Responses

HTTP Code	Description	Schema
201	Created	No Content

## Consumes

- application/json

## Produces

- \*/\*

## Tags

- campaigns-administration

## Security

Type	Name	Scopes
basic	basic	scope

## Get list of enabled car categories

```
GET /rest/carTypes{?cityId}
```

## Parameters

Type	Name	Description	Schema	Default
Query	cityId <i>required</i>	City ID	integer (int64)	1

## Responses

HTTP Code	Description	Schema
200	OK	< CityCarTypeDto > array

## Produces

- `*/*`

## Tags

- car-types

## Security

Type	Name	Scopes
basic	basic	scope

## Upload a new car photo as a driver or an admin

```
POST /rest/carphotos/car/{carId}
```

## Parameters

Type	Name	Description	Schema
Path	<b>carId</b> <i>optional</i>	Car ID	integer (int64)
FormData	<b>photo</b> <i>optional</i>	Image file	file
Body	<b>carPhotoType</b> <i>optional</i>	Photo type	enum (LICENSE, INSURANCE, PHOTO, FRONT, BACK, INSIDE, TRUNK, TNC_CARD, CHAUFFEUR_LICENSE, CAR_STICKER)

## Responses

HTTP Code	Description	Schema
200	OK	CarPhotoDto
403	User is not allowed to upload photo	No Content

HTTP Code	Description	Schema
404	Car doesn't exist	No Content
500	Failed to upload file to S3	No Content

## Consumes

- `multipart/form-data`

## Produces

- `application/json`

## Tags

- car-photos-common

## Security

Type	Name	Scopes
basic	basic	scope

## List of car photos to get as a driver

```
GET /rest/carphotos/car/{carId}
```

## Parameters

Type	Name	Description	Schema
Path	<b>carId</b> <i>optional</i>	Car ID	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	< CarPhotoDto > array

## Produces

- `*/*`

## Tags

- car-photos

## Security

Type	Name	Scopes
basic	basic	scope

## Remove existing car photo as a driver or an admin

```
DELETE /rest/carphotos/{carPhotoId}
```

## Parameters

Type	Name	Description	Schema
Path	<b>carPhotoId</b> <i>optional</i>	Photo ID	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	No Content
204	No Content	No Content
403	User is not allowed to upload photo	No Content
404	Car doesn't exist	No Content

## Produces

- `*/*`

## Tags

- car-photos-common

## Security

Type	Name	Scopes
basic	basic	scope

## List all available charity organizations

```
GET /rest/charities{?cityId}
```

## Parameters

Type	Name	Description	Schema	Default
Query	cityId <i>required</i>	City ID	integer (int64)	1

## Responses

HTTP Code	Description	Schema
200	OK	< Charity > array
400	City ID is invalid	No Content

## Produces

- \*/\*

## Tags

- charities

## Security

Type	Name	Scopes
basic	basic	scope

## Get information about all supported cities

```
GET /rest/cities
```

## Responses

HTTP Code	Description	Schema
200	OK	< City > array

## Produces

- \*/\*

## Tags

- cities

## Security

Type	Name	Scopes
basic	basic	scope

## Get information about requested city

```
GET /rest/cities/{id}
```

## Parameters

Type	Name	Description	Schema
Path	<b>id</b> <i>optional</i>	City ID	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	City

## Produces

- \*/\*

## Tags

- cities

## Security

Type	Name	Scopes
basic	basic	scope

## Create new application info

```
POST /rest/configs/app/info
```

## Parameters

Type	Name	Description	Schema
Body	info <i>required</i>	Application info object	<a href="#">AppInfo</a>

## Responses

HTTP Code	Description	Schema
200	OK	<a href="#">AppInfo</a>
400	Application information provided is invalid	No Content

## Consumes

- [application/json](#)

## Produces

- [application/json](#)

## Tags

- configs-administration

## Security

Type	Name	Scopes
basic	basic	scope

## Get latest released application information to enable force/mandatory upgrade on app side

```
GET /rest/configs/app/info/current{?avatarType,platformType}
```

### Parameters

Type	Name	Description	Schema
Query	<b>avatarType</b> <i>required</i>	Application type	enum (RIDER, DRIVER)
Query	<b>platformType</b> <i>required</i>	Platform type	enum (IOS, ANDROID)

### Responses

HTTP Code	Description	Schema
200	OK	<a href="#">AppInfo</a>

### Produces

- application/json

### Tags

- configs

### Security

Type	Name	Scopes
basic	basic	scope

## Update application information as an admin

```
PUT /rest/configs/app/info/{id}
```

## Parameters

Type	Name	Description	Schema
Path	<b>id</b> <i>optional</i>	App info ID	integer (int64)
Body	<b>info</b> <i>optional</i>	Updated application info object	<a href="#">AppInfo</a>

## Responses

HTTP Code	Description	Schema
200	OK	<a href="#">AppInfo</a>
400	Application information provided is invalid	No Content
404	Application information not found	No Content

## Consumes

- [application/json](#)

## Produces

- [application/json](#)

## Tags

- configs-administration

## Security

Type	Name	Scopes
basic	<a href="#">basic</a>	scope

## Delete application information as an admin

```
DELETE /rest/configs/app/info/{id}
```

## Parameters

Type	Name	Description	Schema
Path	<b>id</b> <i>optional</i>	Application info ID	integer (int64)

## Responses

HTTP Code	Description	Schema
204	No Content	No Content

## Produces

- application/json

## Tags

- configs-administration

## Security

Type	Name	Scopes
basic	<b>basic</b>	scope

## Paginated list of application infos available

```
GET
```

```
/rest/configs/app/info{?avatarType,cityId,desc,page,pageSize,platformType,search,sort}
```

## Parameters

Type	Name	Description	Schema
Query	<b>avatarType</b> <i>optional</i>	Application type	enum RIDER, DRIVER

Type	Name	Description	Schema
Query	<b>cityId</b> <i>required</i>	City ID	integer (int64)
Query	<b>desc</b> <i>optional</i>	Order the result descending	boolean
Query	<b>page</b> <i>optional</i>	Page number	integer (int32)
Query	<b>pageSize</b> <i>optional</i>	Page size	integer (int32)
Query	<b>platformType</b> <i>optional</i>	Platform type	enum (IOS, ANDROID)
Query	<b>search</b> <i>optional</i>	Search query - user agent or application version	string
Query	<b>sort</b> <i>optional</i>	Columns to sort the result by, comma-separated list	< string > array(multi)

## Responses

HTTP Code	Description	Schema
200	OK	<a href="#">AppInfo</a>

## Produces

- application/json

## Tags

- configs-administration

## Security

Type	Name	Scopes
basic	<a href="#">basic</a>	scope

# Information on current server build

```
GET /rest/configs/build
```

## Responses

HTTP Code	Description	Schema
200	OK	<a href="#">BuildInfo</a>

## Produces

- `application/json`

## Tags

- configs-administration

## Security

Type	Name	Scopes
basic	<a href="#">basic</a>	scope

# Get all supported cars makes and models to show in driver app

```
GET /rest/configs/cars
```

## Parameters

Type	Name	Description	Schema
Header	<b>If-Modified-Since</b> <i>optional</i>	If-Modified-Since	string (date-time)

## Responses

HTTP Code	Description	Schema
200	OK	< CarModelInfo > array
304	Cars list was not modified since date provided in If-Modified-Since header	No Content

## Produces

- application/json

## Tags

- configs

## Security

Type	Name	Scopes
basic	basic	scope

## Dynamically generated configuration JSON object for driver application

```
GET /rest/configs/driver/global{?cityId,configAttributes,lat,lng}
```

## Parameters

Type	Name	Description	Schema
Query	cityId <i>optional</i>	City ID to get configuration for	integer (int64)
Query	configAttributes <i>optional</i>	Set of requested configuration parameters. Endpoint will return entire configuration object if this parameter is missing	< string > array(multi)
Query	lat <i>required</i>	Current GPS latitude	number (double)
Query	lng <i>required</i>	Current GPS longitude	number (double)

## Responses

HTTP Code	Description	Schema
200	OK	object
400	Latitude or longitude are invalid	No Content
500	Server is misconfigured, configuration JSON can not be parsed	No Content

## Produces

- application/json

## Tags

- configs

## Security

Type	Name	Scopes
basic	basic	scope

## Create a new configuration item

```
POST /rest/configs/items
```

## Parameters

Type	Name	Description	Schema
Body	createRequest <i>optional</i>	New configuration item	CreateRequest

## Responses

HTTP Code	Description	Schema
200	OK	ConfigurationItem Dto

## Consumes

- application/json

## Produces

- application/json

## Tags

- configuration-items

## Security

Type	Name	Scopes
basic	basic	scope

## Get list of db-stored configuration items as an admin

```
GET /rest/configs/items
```

## Responses

HTTP Code	Description	Schema
200	OK	< ConfigurationItem Dto > array

## Produces

- application/json

## Tags

- configuration-items

## Security

Type	Name	Scopes
basic	basic	scope

# Get a single configuration item object

```
GET /rest/configs/items/{id}
```

## Parameters

Type	Name	Description	Schema
Path	<b>id</b> <i>optional</i>	Configuration item ID	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	<a href="#">ConfigurationItem Dto</a>
404	Configuration item not found	No Content

## Produces

- application/json

## Tags

- configuration-items

## Security

Type	Name	Scopes
basic	<a href="#">basic</a>	scope

# Update a configuration item

```
PUT /rest/configs/items/{id}
```

## Parameters

Type	Name	Description	Schema
<b>Path</b>	<b>id</b> <i>optional</i>	Configuration item ID	integer (int64)
<b>Body</b>	<b>value</b> <i>optional</i>	Updated configuration value	<a href="#">UpdateRequest</a>

## Responses

HTTP Code	Description	Schema
<b>200</b>	OK	<a href="#">ConfigurationItem Dto</a>
<b>400</b>	Request payload is invalid	No Content
<b>404</b>	Configuration item not found	No Content

## Consumes

- [application/json](#)

## Produces

- [application/json](#)

## Tags

- configuration-items

## Security

Type	Name	Scopes
<b>basic</b>	<a href="#">basic</a>	scope

## Remove existing configuration item

```
DELETE /rest/configs/items/{id}
```

## Parameters

Type	Name	Description	Schema
Path	<b>id</b> <i>optional</i>	Configuration item ID	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	No Content

## Produces

- application/json

## Tags

- configuration-items

## Security

Type	Name	Scopes
basic	<b>basic</b>	scope

# Dynamically generated configuration JSON object for rider application

```
GET /rest/configs/rider/global{?cityId,configAttributes,lat,lng}
```

## Parameters

Type	Name	Description	Schema
Query	<b>cityId</b> <i>optional</i>	City ID to get configuration for	integer (int64)
Query	<b>configAttributes</b> <i>optional</i>	Set of requested configuration parameters. Endpoint will return entire configuration object if this parameter is missing	< string array(multi) >

Type	Name	Description	Schema
Query	<b>lat</b> <i>required</i>	Current GPS latitude	number (double)
Query	<b>lng</b> <i>required</i>	Current GPS longitude	number (double)

## Responses

HTTP Code	Description	Schema
200	OK	object
400	Latitude or longitude are invalid	No Content
500	Server is misconfigured, configuration JSON can not be parsed	No Content

## Produces

- application/json

## Tags

- configs

## Security

Type	Name	Scopes
basic	<b>basic</b>	scope

## Trigger a custom payment report job

```
GET /rest/custompayment/report{?cityId,createAfter,createBefore,paymentDate,recipient}
```

## Parameters

Type	Name	Description	Schema	Default
Query	<b>cityId</b> <i>required</i>	City ID	integer (int64)	1

Type	Name	Description	Schema	Default
Query	<b>createAfter required</b>	Include payments created after	string (date-time)	
Query	<b>createBefore required</b>	Include payments created before	string (date-time)	
Query	<b>paymentDate optional</b>	Include payments for date	string (date-time)	
Query	<b>recipient required</b>	Comma-separated list of recipient email addresses	< string > array(multi)	

## Responses

HTTP Code	Description	Schema
200	OK	No Content
500	Custom payment report job failed to be scheduled	No Content

## Produces

- application/json

## Tags

- custom-payments

## Security

Type	Name	Scopes
basic	<b>basic</b>	scope

## Get one custom payment object

```
GET /rest/custompayment/{id}
```

## Parameters

Type	Name	Description	Schema
Path	<b>id</b> <i>optional</i>	Custom payment ID	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	<a href="#">CustomPaymentDt0</a>

## Produces

- application/json

## Tags

- custom-payments

## Security

Type	Name	Scopes
basic	<a href="#">basic</a>	scope

## List all custom payments complying with provided search criteria

GET

```
/rest/custompayment{?administratorId,avatarName,category,cityId,createdOnAfter,createdOnBefore,desc,description,driverId,driverName,page,pageSize,sort}
```

## Parameters

Type	Name	Description	Schema
Query	<b>administratorId</b> <i>optional</i>	Admin ID	integer (int64)

Type	Name	Description	Schema
Query	<b>avatarName</b> <i>optional</i>	Creator name	string
Query	<b>category</b> <i>optional</i>	Custom payment type	< enum (INCENTIVE, BONUS, COMPENSATION, REIMBURSEMENT, OTHER) > array(multi)
Query	<b>cityId</b> <i>optional</i>	City ID	integer (int64)
Query	<b>createdOnAfter</b> <i>optional</i>	Payment created after date	string (date-time)
Query	<b>createdOnBefore</b> <i>optional</i>	Payment created before date	string (date-time)
Query	<b>desc</b> <i>optional</i>	Order the result descending	boolean
Query	<b>description</b> <i>optional</i>	Custom payment description	string
Query	<b>driverId</b> <i>optional</i>	Driver ID	integer (int64)
Query	<b>driverName</b> <i>optional</i>	Driver name	string
Query	<b>page</b> <i>optional</i>	Page number	integer (int32)
Query	<b>pageSize</b> <i>optional</i>	Page size	integer (int32)
Query	<b>sort</b> <i>optional</i>	Columns to sort the result by, comma-separated list	< string > array(multi)

## Responses

HTTP Code	Description	Schema
200	OK	< CustomPaymentDt o > array

## Produces

- application/json

## Tags

- custom-payments

## Security

Type	Name	Scopes
basic	basic	scope

## Get a list of changes submitted to a driver profile by date

```
GET /rest/driver-changes{?auditDay,driverId}
```

## Parameters

Type	Name	Description	Schema
Query	<b>auditDay</b> <i>optional</i>	Day to get changes for	string (date-time)
Query	<b>driverId</b> <i>required</i>	Driver ID	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	< ChangeDto > array
404	Driver not found	No Content

## Produces

- application/json

## Tags

- driver-changes

## Security

Type	Name	Scopes
basic	basic	scope

## Get available driver types

```
GET /rest/driverTypes{?cityId}
```

## Parameters

Type	Name	Description	Schema	Default
Query	cityId <i>optional</i>	City ID	integer (int64)	1

## Responses

HTTP Code	Description	Schema
200	OK	< CityDriverTypeDt o > array

## Produces

- \*/\*

## Tags

- driver-types

## Security

Type	Name	Scopes
basic	basic	scope

## Sign up as a driver

```
POST /rest/drivers
```

## Parameters

Type	Name	Description	Schema
FormData	acceptedTermId <i>optional</i>	Terms ID to be accepted	integer (int64)
FormData	driver <i>required</i>	driver	object
FormData	insuranceData <i>required</i>	insuranceData	file
FormData	licenseData <i>required</i>	licenseData	file

## Responses

HTTP Code	Description	Schema
200	OK	MobileDriverDriverDto

HTTP Code	Description	Schema
400	Registration data is invalid	No Content
409	Driver did not accept usage terms	No Content
500	Failed to send signup email	No Content

## Consumes

- `multipart/form-data`

## Produces

- `application/json`

## Tags

- drivers

## Security

Type	Name	Scopes
basic	basic	scope

## Get list of enabled car categories

```
GET /rest/drivers/carTypes{?cityId}
```

## Parameters

Type	Name	Description	Schema	Default
Query	cityId <i>required</i>	City ID	integer (int64)	1

## Responses

HTTP Code	Description	Schema
200	OK	< <a href="#">CityCarTypeDto</a> > array

## Produces

- [\\*/\\*](#)

## Tags

- car-types

## Security

Type	Name	Scopes
basic	<a href="#">basic</a>	scope

## Get Direct Connect driver information as a rider

```
GET /rest/drivers/connect/{id}{?lat,lng}
```

## Parameters

Type	Name	Description	Schema
Path	<b>id</b> <i>optional</i>	Driver DCID	string
Query	<b>lat</b> <i>required</i>	Current GPS latitude	number (double)
Query	<b>lng</b> <i>required</i>	Current GPS longitude	number (double)

## Responses

HTTP Code	Description	Schema
200	OK	<a href="#">DirectConnectDriverDto</a>

HTTP Code	Description	Schema
404	Driver not found by DCID	No Content

## Produces

- application/json

## Tags

- drivers

## Security

Type	Name	Scopes
basic	basic	scope

## Get current driver information

```
GET /rest/drivers/current
```

## Responses

HTTP Code	Description	Schema
200	OK	MobileDriverDriverDto
403	Requesting user is not a driver	No Content

## Produces

- application/json

## Tags

- drivers

## Security

Type	Name	Scopes
basic	basic	scope

## Download a sample CSV to fill in for further uploading

GET /rest/drivers/data/sample

### Responses

HTTP Code	Description	Schema
200	OK	No Content
500	Failed to generate sample file	No Content

### Produces

- application/octet-stream

### Tags

- drivers-data

### Security

Type	Name	Scopes
basic	basic	scope

## CSV bulk upload for driver's data

POST /rest/drivers/data/upload

### Parameters

Type	Name	Description	Schema
FormData	driversDataCs v <i>required</i>	driversDataCsv	file

## Responses

HTTP Code	Description	Schema
200	OK	ResponseEntity
204	Upload success	ResponseEntity
400	Provided data is invalid	No Content

## Consumes

- `multipart/form-data`

## Produces

- `*/*`

## Tags

- drivers-data

## Security

Type	Name	Scopes
basic	basic	scope

## Export filtered list of drivers as an email

```
POST /rest/drivers/export{?Car category,Car inspection status,City ID,City approval  
status,Created after,Created before,Driver ID,Driver activation status,Driver is in  
PENDING onboarding status more than X days,Last communication later than X  
days,Onboarding status,Payoneer status,Signed up after,Signed up before}
```

## Parameters

Type	Name	Schema
Query	<b>Car category</b> <i>optional</i>	< string > array(multi)
Query	<b>Car inspection status</b> <i>optional</i>	< enum (APPROVED, REJECTED, PENDING, NOT_INSPECTED) > array(multi)

Type	Name	Schema
Query	<b>City ID</b> <i>optional</i>	integer (int64)
Query	<b>City approval status</b> <i>optional</i>	< enum (PENDING, NOT_PROVIDED, APPROVED, REJECTED_PHOTO, REJECTED_BY_CITY, EXPIRED) > array(multi)
Query	<b>Created after</b> <i>optional</i>	string (date-time)
Query	<b>Created before</b> <i>optional</i>	string (date-time)
Query	<b>Driver ID</b> <i>optional</i>	integer (int64)
Query	<b>Driver activation status</b> <i>optional</i>	< enum (ACTIVE, REJECTED, SUSPENDED, DEACTIVATED_OTHER, INACTIVE) > array(multi)
Query	<b>Driver is in PENDING onboarding status more than X days</b> <i>optional</i>	integer (int64)
Query	<b>Last communication later than X days</b> <i>optional</i>	integer (int64)
Query	<b>Onboarding status</b> <i>optional</i>	< enum (REJECTED, SUSPENDED, PENDING, FINAL REVIEW, ACTIVE) > array(multi)
Query	<b>Payoneer status</b> <i>optional</i>	< enum (Not registered, Inactive, Active) > array(multi)
Query	<b>Signed up after</b> <i>optional</i>	string (date-time)
Query	<b>Signed up before</b> <i>optional</i>	string (date-time)

## Responses

HTTP Code	Description	Schema
200	OK	No Content
500	Failed to schedule an export job	No Content

## Consumes

- application/json

## Produces

- \*/\*

## Tags

- drivers-administration

## Security

Type	Name	Scopes
basic	basic	scope

## Get a list of drivers

```
GET /rest/drivers/list{?Car category,Car inspection status,City ID,City approval  
status,Created after,Created before,Driver ID,Driver activation status,Driver is in  
PENDING onboarding status more than X days,Last communication later than X  
days,Onboarding status,Payoneer status,Signed up after,Signed up  
before,desc,page,pageSize,sort}
```

## Parameters

Type	Name	Description	Schema
Query	Car category <i>optional</i>		< string > array(multi)

Type	Name	Description	Schema
Query	<b>Car inspection status</b> <i>optional</i>		< enum (APPROVED, REJECTED, PENDING, NOT_INSPECTED) > array(multi)
Query	<b>City ID</b> <i>optional</i>		integer (int64)
Query	<b>City approval status</b> <i>optional</i>		< enum (PENDING, NOT_PROVIDED, APPROVED, REJECTED_PHOTO, REJECTED_BY_CITY, EXPIRED) > array(multi)
Query	<b>Created after</b> <i>optional</i>		string (date-time)
Query	<b>Created before</b> <i>optional</i>		string (date-time)
Query	<b>Driver ID</b> <i>optional</i>		integer (int64)
Query	<b>Driver activation status</b> <i>optional</i>		< enum (ACTIVE, REJECTED, SUSPENDED, DEACTIVATED_OTHER, INACTIVE) > array(multi)
Query	<b>Driver is in PENDING onboarding status more than X days</b> <i>optional</i>		integer (int64)

Type	Name	Description	Schema
Query	<b>Last communication later than X days</b> <i>optional</i>		integer (int64)
Query	<b>Onboarding status</b> <i>optional</i>		< enum (REJECTED, SUSPENDED, PENDING, FINAL REVIEW, ACTIVE) > array(multi)
Query	<b>Payoneer status</b> <i>optional</i>		< enum (Not registered, Inactive, Active) > array(multi)
Query	<b>Signed up after</b> <i>optional</i>		string (date-time)
Query	<b>Signed up before</b> <i>optional</i>		string (date-time)
Query	<b>desc</b> <i>optional</i>	Order the result descending	boolean
Query	<b>page</b> <i>optional</i>	Page number	integer (int32)
Query	<b>pageSize</b> <i>optional</i>	Page size	integer (int32)
Query	<b>sort</b> <i>optional</i>	Columns to sort the result by, comma-separated list	< string > array(multi)

## Responses

HTTP Code	Description	Schema
200	OK	< SimpleDriverDto > array

## Produces

- `*/*`

## Tags

- drivers-administration

## Security

Type	Name	Scopes
basic	<a href="#">basic</a>	scope

# Update Payoneer status for all driver who have pending Payoneer registration

```
GET /rest/drivers/payoneerStatus
```

## Responses

HTTP Code	Description	Schema
200	OK	<a href="#">ResponseEntity</a>
500	Payoneer communication failure	No Content

## Produces

- `*/*`

## Tags

- drivers-administration

## Security

Type	Name	Scopes
basic	basic	scope

## Get a list of available email reminders

GET /rest/drivers/reminders

### Responses

HTTP Code	Description	Schema
200	OK	< DriverEmailReminderDto > array

### Produces

- \*/\*

### Tags

- driver-email-reminders

### Security

Type	Name	Scopes
basic	basic	scope

## Get content of an already sent reminder

GET /rest/drivers/reminders/history/{reminderHistoryId}

### Parameters

Type	Name	Description	Schema
Path	reminderHistoryId <i>optional</i>	Reminder history item ID	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	string
404	History item not found	No Content
500	Failed to render history item template	No Content

## Produces

- \*/\*

## Tags

- driver-email-reminders

## Security

Type	Name	Scopes
basic	basic	scope

## Set a car as the default for a driver

```
PUT /rest/drivers/selected{?carId,driverId}
```

## Parameters

Type	Name	Description	Schema
Query	<b>carId</b> <i>optional</i>	Car ID	integer (int64)
Query	<b>driverId</b> <i>optional</i>	Driver ID	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	CarDto
400	Trying to set unapproved or removed car as the default	No Content
403	Requesting user is not a driver or admin	No Content
404	Driver or car is not found	No Content

## Consumes

- application/json

## Produces

- application/json

## Tags

- driver-cars

## Security

Type	Name	Scopes
basic	basic	scope

## Get counts of drivers in pending statuses by various criteria.

```
GET /rest/drivers/statuses/pending{?cityId}
```

## Parameters

Type	Name	Description	Schema	Default
Query	cityId <i>required</i>	City ID	integer (int64)	1

## Responses

HTTP Code	Description	Schema
200	OK	DriverStatusPendingDto

## Produces

- \*/\*

## Tags

- drivers-administration

## Security

Type	Name	Scopes
basic	basic	scope

## Get count of drivers by their statuses

```
GET /rest/drivers/statuses{?cityId}
```

## Parameters

Type	Name	Description	Schema	Default
Query	cityId <i>required</i>	City ID	integer (int64)	1

## Responses

HTTP Code	Description	Schema
200	OK	DriverStatusDto

## Produces

- \*/\*

## Tags

- drivers-administration

## Security

Type	Name	Scopes
basic	basic	scope

## Accept usage terms

```
PUT /rest/drivers/terms/{termsId}
```

## Parameters

Type	Name	Description	Schema
Path	<b>termsId</b> <i>optional</i>	Terms ID	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	MobileDriverDriverDto
400	Terms have been already accepted	No Content
403	Requesting user is not a driver	No Content
409	Terms not found	No Content

## Consumes

- application/json

## Produces

- \*/\*

## Tags

- drivers

## Security

Type	Name	Scopes
basic	basic	scope

## Get all present driver's cars

```
GET /rest/drivers/{driverId}/allCars
```

## Parameters

Type	Name	Description	Schema
Path	driverId <i>optional</i>	Driver ID	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	<CarDto> array
403	Requesting user is not a driver	No Content
404	Driver not found	No Content

## Produces

- application/json

## Tags

- driver-cars

## Security

Type	Name	Scopes
basic	basic	scope

# Add a new car to driver's profile

POST /rest/drivers/{driverId}/cars

## Parameters

Type	Name	Description	Schema
Path	<b>driverId</b> <i>optional</i>	Driver ID	integer (int64)
FormData	<b>car</b> <i>required</i>	car	object
FormData	<b>insurancePhoto</b> <b>to</b> <i>optional</i>	insurancePhoto	file
FormData	<b>photo</b> <i>optional</i>	photo	file

## Responses

HTTP Code	Description	Schema
200	OK	CarDto
403	Requesting user is not a driver or admin	No Content
404	Driver is not found	No Content
500	Failed to upload car photo	No Content

## Consumes

- `multipart/form-data`

## Produces

- `application/json`

## Tags

- driver-cars

## Security

Type	Name	Scopes
basic	<a href="#">basic</a>	scope

## Update car information

```
PUT /rest/drivers/{driverId}/cars/{carId}
```

### Parameters

Type	Name	Description	Schema
Path	<b>carId</b> <i>optional</i>	Car ID	integer (int64)
Path	<b>driverId</b> <i>optional</i>	Driver ID	integer (int64)
Body	<b>dto</b> <i>required</i>	Updated car object	<a href="#">CarDto</a>

### Responses

HTTP Code	Description	Schema
200	OK	<a href="#">CarDto</a>
403	Requesting user is not a driver or admin	No Content
404	Driver, car or insurance is not found	No Content
500	Failed to send a push notification to driver when car category is changed	No Content

### Consumes

- [application/json](#)

### Produces

- [application/json](#)

## Tags

- driver-cars

## Security

Type	Name	Scopes
basic	basic	scope

## Remove a car

```
DELETE /rest/drivers/{driverId}/cars/{carId}
```

## Parameters

Type	Name	Description	Schema
Path	<b>carId</b> <i>optional</i>	Car ID	integer (int64)
Path	<b>driverId</b> <i>optional</i>	Driver ID	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	No Content
403	Requesting user is not a driver or admin	No Content
404	Driver or car is not found	No Content

## Produces

- application/json

## Tags

- driver-cars

## Security

Type	Name	Scopes
basic	basic	scope

## Get current driver's position in all present queues

```
GET /rest/drivers/{driverId}/queue
```

## Parameters

Type	Name	Description	Schema
Path	<b>driverId</b> <i>optional</i>	Driver ID	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	AreaQueuePositions
403	Requesting user is not a driver	No Content
404	Driver not found	No Content

## Produces

- application/json

## Tags

- driver-area-queue-common

## Security

Type	Name	Scopes
basic	basic	scope

# Get reminder content

```
GET /rest/drivers/{driverId}/reminders/{reminderId}
```

## Parameters

Type	Name	Description	Schema
Path	<b>driverId</b> <i>optional</i>	Driver ID	integer (int64)
Path	<b>reminderId</b> <i>optional</i>	Reminder ID	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	string
404	Driver or reminder item is not found	No Content
500	Failed to render history item template	No Content

## Produces

- \*/\*

## Tags

- driver-email-reminders

## Security

Type	Name	Scopes
basic	basic	scope

# Send a reminder to a driver

```
POST /rest/drivers/{driverId}/reminders/{reminderId}{?content,subject}
```

## Parameters

Type	Name	Description	Schema
Path	<b>driverId</b> <i>optional</i>	Driver ID	integer (int64)
Path	<b>reminderId</b> <i>optional</i>	Reminder ID	integer (int64)
Query	<b>content</b> <i>optional</i>	Reminder content	string
Query	<b>subject</b> <i>optional</i>	Reminder subject	string

## Responses

HTTP Code	Description	Schema
202	OK	No Content
400	Content parameter is required, but was not provided	No Content
500	Failed to send email	No Content

## Consumes

- application/json

## Produces

- \*/\*

## Tags

- driver-email-reminders

## Security

Type	Name	Scopes
basic	basic	scope

# Get acceptance statistics for a driver

```
GET /rest/drivers/{driverId}/stats
```

## Parameters

Type	Name	Description	Schema
Path	<b>driverId</b> <i>optional</i>	Driver ID	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	< DriverStatisticDto > array
400	Stats not found for a driver	No Content

## Produces

- application/json

## Tags

- driver-statistics

## Security

Type	Name	Scopes
basic	<b>basic</b>	scope

# Get information about a driver

```
GET /rest/drivers/{id}
```

## Parameters

Type	Name	Description	Schema
Path	<b>id</b> <i>optional</i>	Driver ID	integer (int64)

## Responses

HTTP Code	Description	Schema
<b>200</b>	OK	<a href="#">ConsoleDriverDto</a>
<b>404</b>	Driver not found	No Content
<b>500</b>	Failed to update payoneer status	No Content

## Produces

- `*/*`

## Tags

- drivers-administration

## Security

Type	Name	Scopes
basic	<a href="#">basic</a>	scope

## Update driver information

```
PUT /rest/drivers/{id}
```

## Parameters

Type	Name	Description	Schema
Path	<b>id</b> <i>optional</i>	Driver ID	integer (int64)
Body	<b>driver</b> <i>optional</i>	driver	<a href="#">ConsoleDriverDto</a>

## Responses

HTTP Code	Description	Schema
200	OK	ConsoleDriverDto
400	Provided data is invalid	No Content
404	Driver not found	No Content

## Consumes

- application/json; charset=UTF-8

## Produces

- \*/\*

## Tags

- drivers-administration

## Security

Type	Name	Scopes
basic	basic	scope

## Send an activation email to a driver

```
POST /rest/drivers/{id}/activationEmail
```

## Parameters

Type	Name	Description	Schema
Path	<b>id</b> <i>optional</i>	Driver ID	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	No Content
400	Driver is not activated	No Content
404	Driver not found	No Content
500	Failed to send an email	No Content

## Consumes

- application/json

## Produces

- \*/\*

## Tags

- drivers-administration

## Security

Type	Name	Scopes
basic	basic	scope

## Create a new custom payment for a driver

```
POST /rest/drivers/{id}/custompayment{?description,paymentDate,type,value}
```

## Parameters

Type	Name	Description	Schema
Path	<b>id</b> <i>optional</i>	Driver ID	integer (int64)
Query	<b>description</b> <i>required</i>	Custom payment description	string

Type	Name	Description	Schema
Query	<b>paymentDate required</b>	Custom payment date	string (date-time)
Query	<b>type required</b>	Type of custom payment	enum (INCENTIVE, BONUS, COMPENSATION, REIMBURSEMENT, OTHER)
Query	<b>value required</b>	Custom payment amount	number (double)

## Responses

HTTP Code	Description	Schema
200	OK	CustomPaymentDt o

## Consumes

- application/json

## Produces

- application/json

## Tags

- custom-payments

## Security

Type	Name	Scopes
basic	basic	scope

## Assign new Direct Connect ID

```
GET /rest/drivers/{id}/dcid
```

## Parameters

Type	Name	Description	Schema
Path	<b>id</b> <i>optional</i>	Driver ID	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	<a href="#">DirectConnectDto</a>
400	New DCID can not be assigned	No Content

## Produces

- application/json

## Tags

- drivers

## Security

Type	Name	Scopes
basic	<a href="#">basic</a>	scope

## Assign driver types to a driver

```
PUT /rest/drivers/{id}/driverTypes{?driverTypes}
```

## Parameters

Type	Name	Description	Schema
Path	<b>id</b> <i>optional</i>	Driver ID	integer (int64)
Query	<b>driverTypes</b> <i>optional</i>	driverTypes	< string > array(multi)

## Responses

HTTP Code	Description	Schema
200	OK	No Content
404	Driver not found	No Content

## Consumes

- application/json

## Produces

- \*/\*

## Tags

- drivers-administration

## Security

Type	Name	Scopes
basic	basic	scope

## Send a earnings statement for a driver

```
POST /rest/drivers/{id}/earnings{?date,recipient}
```

## Parameters

Type	Name	Description	Schema
Path	<b>id</b> <i>optional</i>	Driver ID	integer (int64)
Query	<b>date</b> <i>optional</i>	Date to get the earnings statement for	string (date)
Query	<b>recipient</b> <i>optional</i>	List of recipient emails	< string > array(multi)

## Responses

HTTP Code	Description	Schema
200	OK	No Content
400	Failed to schedule	No Content
404	Driver not found	No Content

## Consumes

- application/json

## Produces

- \*/\*

## Tags

- drivers-common

## Security

Type	Name	Scopes
basic	basic	scope

## Get statistics on online time for a driver

```
GET /rest/drivers/{id}/online{?from,to}
```

## Parameters

Type	Name	Description	Schema
Path	<b>id</b> <i>optional</i>	Driver ID	integer (int64)
Query	<b>from</b> <i>required</i>	Report start time	string (date-time)
Query	<b>to</b> <i>required</i>	Report end time	string (date-time)

## Responses

HTTP Code	Description	Schema
200	OK	DriverOnline
403	Requesting user is not a driver	No Content
404	Driver not found	No Content

## Produces

- application/json

## Tags

- driver-reports

## Security

Type	Name	Scopes
basic	basic	scope

## Update driver's photo

```
POST /rest/drivers/{id}/photo
```

## Parameters

Type	Name	Description	Schema
Path	<b>id</b> <i>optional</i>	Driver ID	integer (int64)
FormData	<b>photoData</b> <i>optional</i>	Image file	file

## Responses

HTTP Code	Description	Schema
200	OK	MobileDriverDriverDto
404	Driver not found	No Content
500	Failed to upload a photo	No Content

## Consumes

- `multipart/form-data`

## Produces

- `application/json`

## Tags

- drivers-common

## Security

Type	Name	Scopes
basic	basic	scope

## Disable driver immediately and set driver offline

```
DELETE /rest/drivers/{id}/quickdisable
```

## Parameters

Type	Name	Description	Schema
Path	<b>id</b> <i>optional</i>	Driver ID	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	No Content
500	Failed to send push notification	No Content

## Produces

- \*/\*

## Tags

- drivers-administration

## Security

Type	Name	Scopes
basic	basic	scope

## Send a referral email to invite a new driver

```
POST /rest/drivers/{id}/referAFriendByEmail{?cityId,email}
```

## Parameters

Type	Name	Description	Schema
Path	<b>id</b> <i>optional</i>	Driver ID	integer (int64)
Query	<b>cityId</b> <i>optional</i>	City ID	integer (int64)
Query	<b>email</b> <i>required</i>	Referred email	string

## Responses

HTTP Code	Description	Schema
200	OK	No Content

HTTP Code	Description	Schema
400	Provided email is invalid	No Content
404	Driver not found	No Content
500	Failed to send an email	No Content

## Consumes

- application/json

## Produces

- application/json

## Tags

- drivers-common

## Security

Type	Name	Scopes
basic	basic	scope

## Send a referral text message to invite a new driver

```
POST /rest/drivers/{id}/referAFriendBySMS{?cityId,phoneNumber}
```

## Parameters

Type	Name	Description	Schema
Path	<b>id</b> <i>optional</i>	Driver ID	integer (int64)
Query	<b>cityId</b> <i>optional</i>	City ID	integer (int64)
Query	<b>phoneNumber</b> <i>required</i>	Referred SMS	string

## Responses

HTTP Code	Description	Schema
200	OK	No Content
400	Provided phone number is invalid	No Content
404	Driver not found	No Content
500	Failed to send an SMS	No Content

## Consumes

- application/json

## Produces

- application/json

## Tags

- drivers-common

## Security

Type	Name	Scopes
basic	basic	scope

## Make driver available if it's inconsistently set to RIDING state

```
POST /rest/drivers/{id}/release
```

## Parameters

Type	Name	Description	Schema
Path	<b>id</b> <i>optional</i>	Driver ID	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	No Content
404	Driver not found	No Content

## Consumes

- application/json

## Produces

- \*/\*

## Tags

- drivers-administration

## Security

Type	Name	Scopes
basic	basic	scope

## Get a report on rides performed by a driver

GET

/rest/drivers/{id}/rides{?completedOnAfter,completedOnBefore,desc,page,pageSize,sort}

## Parameters

Type	Name	Description	Schema
Path	<b>id</b> <i>optional</i>	Driver ID	integer (int64)
Query	<b>completedOnAfter</b> <i>optional</i>	Report start datetime	string (date-time)

Type	Name	Description	Schema
Query	<b>completedOnBefore</b> <i>optional</i>	Report end datetime	string (date-time)
Query	<b>desc</b> <i>optional</i>	Order the result descending	boolean
Query	<b>page</b> <i>optional</i>	Page number	integer (int32)
Query	<b>pageSize</b> <i>optional</i>	Page size	integer (int32)
Query	<b>sort</b> <i>optional</i>	Columns to sort the result by, comma-separated list	< string array(multi) >

## Responses

HTTP Code	Description	Schema
200	OK	< DriverRide > array
403	Requesting user is not a driver or admin	No Content
404	Driver not found	No Content

## Produces

- application/json

## Tags

- driver-reports-common

## Security

Type	Name	Scopes
basic	<b>basic</b>	scope

# Update existing document

```
PUT /rest/driversDocuments/{documentId}
```

## Parameters

Type	Name	Description	Schema
Path	<b>documentId</b> <i>optional</i>	Document ID	integer (int64)
Body	<b>document</b> <i>optional</i>	document	<a href="#">Document</a>

## Responses

HTTP Code	Description	Schema
200	OK	<a href="#">Document</a>
404	Document not found	No Content

## Consumes

- [application/json](#)

## Produces

- [application/json](#)

## Tags

- drivers-documents

## Security

Type	Name	Scopes
basic	<a href="#">basic</a>	scope

# Remove document

```
DELETE /rest/driversDocuments/{documentId}
```

## Parameters

Type	Name	Description	Schema
Path	<b>documentId</b> <i>optional</i>	Document ID	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	No Content
404	Document not found	No Content

## Produces

- application/json

## Tags

- drivers-documents-administration

## Security

Type	Name	Scopes
basic	<b>basic</b>	scope

## List all available documents for a car

```
GET  
/rest/driversDocuments/{driverId}/cars/{carId}{?carId,cityId,documentType,driverId}
```

## Parameters

Type	Name	Description	Schema
Path	<b>carId</b> <i>optional</i>	Car ID	integer (int64)

Type	Name	Description	Schema
Path	<b>driverId</b> <i>optional</i>	Driver ID	integer (int64)
Query	<b>carId</b> <i>optional</i>	Car ID	integer (int64)
Query	<b>cityId</b> <i>optional</i>	City ID	integer (int64)
Query	<b>documentType</b> <i>optional</i>	Document type	enum (LICENSE, INSURANCE, PHOTO, FRONT, BACK, INSIDE, TRUNK, TNC_CARD, CHAUFFEUR_LICENSE, CAR_STICKER)
Query	<b>driverId</b> <i>optional</i>	Driver ID	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	< DocumentDto > array
403	Requesting user is not the same driver as requested one	No Content

## Produces

- application/json

## Tags

- drivers-documents

## Security

Type	Name	Scopes
basic	<b>basic</b>	scope

# List all available documents for a driver

```
GET /rest/driversDocuments/{driverId}{?avatarId,cityId,documentType}
```

## Parameters

Type	Name	Description	Schema
Path	<b>driverId</b> <i>optional</i>	Driver ID	integer (int64)
Query	<b>avatarId</b> <i>optional</i>	Avatar ID	integer (int64)
Query	<b>cityId</b> <i>optional</i>	City ID	integer (int64)
Query	<b>documentType</b> <i>optional</i>	Document type	enum (LICENSE, INSURANCE, PHOTO, FRONT, BACK, INSIDE, TRUNK, TNC_CARD, CHAUFFEUR_LICENSE, CAR_STICKER)

## Responses

HTTP Code	Description	Schema
200	OK	< DocumentDto > array
403	Requesting user is not the same driver as requested one	No Content

## Produces

- application/json

## Tags

- drivers-documents

## Security

Type	Name	Scopes
basic	basic	scope

## Add a new document

```
POST /rest/driversDocuments/{driverId}{?carId,cityId,driverPhotoType,validityDate}
```

### Parameters

Type	Name	Description	Schema
Path	<b>driverId</b> <i>optional</i>	Driver ID	integer (int64)
Query	<b>carId</b> <i>optional</i>	Car ID	integer (int64)
Query	<b>cityId</b> <i>optional</i>	City ID	integer (int64)
Query	<b>driverPhotoType</b> <i>required</i>	Document type	enum (LICENSE, INSURANCE, PHOTO, FRONT, BACK, INSIDE, TRUNK, TNC_CARD, CHAUFFEUR_LICENSE, CAR_STICKER)
Query	<b>validityDate</b> <i>optional</i>	Validity date	string (date-time)
FormData	<b>fileData</b> <i>required</i>	Image file	file

### Responses

HTTP Code	Description	Schema
200	OK	MobileDriverDriverDto

HTTP Code	Description	Schema
400	File is not provided or parameters are invalid	No Content
404	Driver not found	No Content
500	Failed to upload image file	No Content

## Consumes

- `multipart/form-data`

## Produces

- `application/json`

## Tags

- drivers-documents

## Security

Type	Name	Scopes
basic	basic	scope

## Long-polling list of events

```
GET /rest/events{?avatarType,lastReceivedEvent}
```

## Parameters

Type	Name	Description	Schema
Query	<b>avatarType</b> <i>optional</i>	Avatar type	enum (RIDER, DRIVER)
Query	<b>lastReceivedEvent</b> <i>optional</i>	ID of last received event	string

## Responses

HTTP Code	Description	Schema
200	OK	< EventDto > array
403	Requesting user doesn't have avatar set	No Content

## Produces

- \*/\*

## Tags

- events

## Security

Type	Name	Scopes
basic	basic	scope

## List payments performed for a ride

```
GET /rest/farepayments/{rideId}/list
```

## Parameters

Type	Name	Description	Schema
Path	rideId <i>optional</i>	Ride ID	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	< FarePaymentDto > array

## Produces

- application/json

## Tags

- fare-payments

## Security

Type	Name	Scopes
basic	basic	scope

## Reset a password as an admin

```
POST /rest/forgot?avatarType=ADMIN&email={email}
```

## Parameters

Type	Name	Description	Schema	Default
Query	<b>avatarType</b> <i>required</i>		enum (ADMIN)	"ADMIN"
Query	<b>email</b> <i>required</i>	User email to reset a password	string	

## Responses

HTTP Code	Description	Schema
200	OK	No Content
404	User not found	No Content
500	Failed to send email	No Content

## Consumes

- application/json

## Produces

- \*/\*

## Tags

- forgot-password

## Security

Type	Name	Scopes
basic	basic	scope

## Request a password reminder email

```
POST /rest/forgot{?email}
```

## Parameters

Type	Name	Description	Schema
Query	<b>email</b> <i>required</i>	User email to reset a password	string

## Responses

HTTP Code	Description	Schema
200	OK	No Content
404	User not found	No Content
500	Failed to send email	No Content

## Consumes

- application/json

## Produces

- \*/\*

## Tags

- forgot-password

## Security

Type	Name	Scopes
basic	basic	scope

## Store rider's position to determine higher demand areas

```
POST /rest/geolog{?locationLat,locationLng,riderId}
```

## Parameters

Type	Name	Description	Schema
Query	<b>locationLat</b> <i>required</i>	Current GPS latitude	number (double)
Query	<b>locationLng</b> <i>required</i>	Current GPS longitude	number (double)
Query	<b>riderId</b> <i>required</i>	Rider ID	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	<a href="#">GeolocationLogDt o</a>
403	Requesting user is trying to add location for some other user	No Content

## Consumes

- [application/json](#)

## Produces

- [application/json](#)

## Tags

- geolocation-logs

## Security

Type	Name	Scopes
basic	basic	scope

## Authenticate as an user

POST /rest/login

## Responses

HTTP Code	Description	Schema
200	OK	<a href="#">AuthenticationToken</a>
401	Credentials are incorrect	No Content
403	Device blocked	No Content
500	Failed to login	No Content

## Consumes

- application/json

## Produces

- \*/\*

## Tags

- authentication

## Security

Type	Name	Scopes
basic	basic	scope

# Log out

```
POST /rest/logout
```

## Responses

HTTP Code	Description	Schema
200	OK	No Content
400	Logging out is forbidden while in a ride	No Content

## Consumes

- application/json

## Produces

- \*/\*

## Tags

- authentication

## Security

Type	Name	Scopes
basic	basic	scope

# Contact the driver about the lost item

```
POST /rest/lostandfound/contact{?phone,rideId}
```

## Parameters

Type	Name	Description	Schema
Query	phone <i>required</i>	Phone number	string

Type	Name	Description	Schema
Query	<b>rideId</b> <i>required</i>	ID of the ride while which the item was lost	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	<a href="#">ContactSuccessResponse</a>
500	Failed to contact the driver	string

## Consumes

- [application/json](#)

## Produces

- [application/json](#)

## Tags

- lost-and-found

## Security

Type	Name	Scopes
basic	<a href="#">basic</a>	scope

## Report found item as a driver

```
POST /rest/lostandfound/Found
```

## Parameters

Type	Name	Description	Schema
FormData	<b>image</b> <i>optional</i>	Found item image file	file

Type	Name	Description	Schema
<b>FormData</b>	<b>item</b> <i>optional</i>	item	object

## Responses

HTTP Code	Description	Schema
<b>200</b>	OK	<a href="#">LostSuccessMessage</a>
<b>202</b>	Report accepted	<a href="#">LostSuccessMessage</a>
<b>403</b>	Driver tries to report lost item for someone else's ride	No Content
<b>404</b>	Ride not found	No Content

## Consumes

- [multipart/form-data](#)

## Produces

- [\\*/\\*](#)

## Tags

- lost-and-found

## Security

Type	Name	Scopes
<b>basic</b>	<a href="#">basic</a>	scope

## Report lost item as a rider

```
POST /rest/lostandfound/lost{?description,details,phone,rideId}
```

## Parameters

Type	Name	Description	Schema
Query	<b>description required</b>	Lost item description	string
Query	<b>details required</b>	Extra details about the lost item	string
Query	<b>phone required</b>	Phone number	string
Query	<b>rideId required</b>	ID of the ride while which the item was lost	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	<a href="#">LostSuccessMessage</a>
202	Report accepted	<a href="#">LostSuccessMessage</a>
403	Rider tries to report lost item for someone other's ride	No Content
404	Ride not found	No Content

## Consumes

- application/json

## Produces

- application/json

## Tags

- lost-and-found

## Security

Type	Name	Scopes
basic	<a href="#">basic</a>	scope

# Get a list of all lost/found item requests per user

```
GET /rest/lostandfound/{avatarId}/requests
```

## Parameters

Type	Name	Description	Schema
Path	<b>avatarId</b> <i>optional</i>	Avatar ID	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	< LostAndFoundRequestDto> array

## Produces

- \*/\*

## Tags

- lost-and-found-administration

## Security

Type	Name	Scopes
basic	<a href="#">basic</a>	scope

# List of available notification topics

```
GET /rest/notifications
```

## Responses

HTTP Code	Description	Schema
200	OK	< Topic > array

## Produces

- application/json

## Tags

- push-notifications

## Security

Type	Name	Scopes
basic	basic	scope

## Push a notification to a specific user

```
POST /rest/notifications/{avatarType}/{avatarId}{?message}
```

## Parameters

Type	Name	Description	Schema
Path	avatarId <i>optional</i>	Avatar ID	integer (int64)
Path	avatarType <i>optional</i>	Avatar type	enum (RIDER, DRIVER)
Query	message <i>optional</i>	Notification content	string

## Responses

HTTP Code	Description	Schema
200	OK	No Content
404	User not found	No Content

## Consumes

- application/json

## Produces

- \*/\*

## Tags

- push-notifications

## Security

Type	Name	Scopes
basic	basic	scope

## Push a notification to a topic

```
POST /rest/notifications/{topicId}{?message}
```

## Parameters

Type	Name	Description	Schema
Path	<b>topicId</b> <i>optional</i>	Topic ID	integer (int64)
Query	<b>message</b> <i>optional</i>	Notification content	string

## Responses

HTTP Code	Description	Schema
200	OK	No Content

## Consumes

- application/json

## Produces

- application/json

## Tags

- push-notifications

## Security

Type	Name	Scopes
basic	basic	scope

## Change password as an user

```
POST /rest/password{?password}
```

## Parameters

Type	Name	Description	Schema
Query	<b>password</b> <i>required</i>	New password	string

## Responses

HTTP Code	Description	Schema
200	OK	No Content

## Consumes

- application/json

## Produces

- \*/\*

## Tags

- authentication

## Security

Type	Name	Scopes
basic	basic	scope

## Request a code to be sent to user's phone

```
POST /rest/phoneVerification/requestCode{?phoneNumber}
```

## Parameters

Type	Name	Description	Schema
Query	<b>phoneNumber</b> <i>required</i>	Phone number	string

## Responses

HTTP Code	Description	Schema
200	OK	AuthenticationToken
400	Phone number is invalid or failure to send a message	No Content

## Consumes

- application/json

## Produces

- \*/\*

## Tags

- phone-verification

## Security

Type	Name	Scopes
basic	basic	scope

# Verify received code

```
POST /rest/phoneVerification/verify{?authToken,code}
```

## Parameters

Type	Name	Description	Schema
Query	<b>authToken</b> <i>required</i>	Token received as a response to /requestCode call	string
Query	<b>code</b> <i>required</i>	Code received in SMS	string

## Responses

HTTP Code	Description	Schema
200	OK	No Content
400	SMS verification failed	No Content

## Consumes

- application/json

## Produces

- \*/\*

## Tags

- phone-verification

## Security

Type	Name	Scopes
basic	basic	scope

# Upload new user profile photo

```
POST /rest/photos
```

## Parameters

Type	Name	Description	Schema
FormData	<b>file</b> <i>required</i>	file	file

## Responses

HTTP Code	Description	Schema
200	OK	<a href="#">URI</a>
201	Upload successful	<a href="#">URI</a>
400	Image file is missing	No Content
500	Failed to upload file	No Content

## Consumes

- `multipart/form-data`

## Produces

- `*/*`

## Tags

- user-photos

## Security

Type	Name	Scopes
basic	<b>basic</b>	scope

## Add a new promocode

```
POST /rest/promocodes
```

## Parameters

Type	Name	Description	Schema
Body	<b>promocodeDt o</b> <i>required</i>	promocodeDto	PromocodeDto

## Responses

HTTP Code	Description	Schema
200	OK	PromocodeDto
400	Provided data is invalid	No Content

## Consumes

- application/json

## Produces

- application/json

## Tags

- promocodes

## Security

Type	Name	Scopes
basic	basic	scope

## Get promocode information

```
GET /rest/promocodes/{promocodeId}
```

## Parameters

Type	Name	Description	Schema
Path	<b>promocodeId</b> <i>optional</i>	Promocode ID	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	PromocodeDto

## Produces

- \*/\*

## Tags

- promocodes

## Security

Type	Name	Scopes
basic	basic	scope

## Update existing promocode

```
PUT /rest/promocodes/{promocodeId}
```

## Parameters

Type	Name	Description	Schema
Path	<b>promocodeId</b> <i>optional</i>	Promocode ID	integer (int64)
Body	<b>promocodeDt</b> <b>o</b> <i>optional</i>	promocodeDto	PromocodeDto

## Responses

HTTP Code	Description	Schema
200	OK	PromocodeDto
400	Provided data is invalid	No Content

## Consumes

- application/json

## Produces

- application/json

## Tags

- promocodes

## Security

Type	Name	Scopes
basic	basic	scope

## Get usage count for a promocode

```
GET /rest/promocodes/{promocodeId}/usage
```

## Parameters

Type	Name	Description	Schema
Path	<b>promocodeId</b> <i>optional</i>	Promocode ID	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	integer (int64)

## Produces

- \*/\*

## Tags

- promocodes

## Security

Type	Name	Scopes
basic	basic	scope

## List available promocodes

```
GET /rest/promocodes{?cityBitMask,codeLiteral,desc,page,pageSize,sort}
```

### Parameters

Type	Name	Description	Schema
Query	<b>cityBitMask</b> <i>optional</i>		integer (int32)
Query	<b>codeLiteral</b> <i>optional</i>		string
Query	<b>desc</b> <i>optional</i>	Order the result descending	boolean
Query	<b>page</b> <i>optional</i>	Page number	integer (int32)
Query	<b>pageSize</b> <i>optional</i>	Page size	integer (int32)
Query	<b>sort</b> <i>optional</i>	Columns to sort the result by, comma-separated list	< string array(multi) >

### Responses

HTTP Code	Description	Schema
200	OK	< ListPromocodeDto > array

### Produces

- \*/\*

## Tags

- promocodes

## Security

Type	Name	Scopes
basic	basic	scope

## Get information on queue-enabled area as an admin

```
GET /rest/queue/{areaId}/info
```

## Parameters

Type	Name	Description	Schema
Path	<b>areaId</b> <i>required</i>	areaId	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	<a href="#">AreaQueueInfo</a>
404	Queue-enabled area is not found	No Content

## Produces

- \*/\*

## Tags

- driver-area-queue-administration

## Security

Type	Name	Scopes
basic	basic	scope

# Get lengths and driver positions for the queue-enabled areas

```
GET /rest/queues{?cityId}
```

## Parameters

Type	Name	Description	Schema	Default
Query	<b>cityId</b> <i>required</i>	City ID	integer (int64)	1

## Responses

HTTP Code	Description	Schema
200	OK	< AreaQueuePositio ns > array

## Produces

- application/json

## Tags

- driver-area-queue

## Security

Type	Name	Scopes
basic	<a href="#">basic</a>	scope

# Get a list of rating updates submitted as a driver / by a driver

```
GET /rest/ratingupdates/byDriver/{driverId}
```

## Parameters

Type	Name	Description	Schema
Path	<b>driverId</b> <i>optional</i>	Driver ID	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	< RatingUpdateDto > array
404	Driver not found	No Content

## Produces

- application/json

## Tags

- rating-updates

## Security

Type	Name	Scopes
basic	<b>basic</b>	scope

## Get a list of rating updates submitted as a rider / by a rider

```
GET /rest/ratingupdates/byRider/{riderId}
```

## Parameters

Type	Name	Description	Schema
Path	<b>riderId</b> <i>optional</i>	Rider ID	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	< RatingUpdateDto > array
404	Rider not found	No Content

## Produces

- application/json

## Tags

- rating-updates

## Security

Type	Name	Scopes
basic	basic	scope

## Get a list of rating updates as a driver / for a driver

```
GET /rest/ratingupdates/forDriver/{driverId}
```

## Parameters

Type	Name	Description	Schema
Path	driverId <i>optional</i>	Driver ID	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	< RatingUpdateDto > array

HTTP Code	Description	Schema
404	Driver not found	No Content

## Produces

- application/json

## Tags

- rating-updates

## Security

Type	Name	Scopes
basic	basic	scope

## Get a list of rating updates as a rider / for a rider

```
GET /rest/ratingupdates/forRider/{riderId}
```

## Parameters

Type	Name	Description	Schema
Path	riderId <i>optional</i>	Rider ID	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	< RatingUpdateDto > array
404	Rider not found	No Content

## Produces

- application/json

## Tags

- rating-updates

## Security

Type	Name	Scopes
basic	basic	scope

## Remove rating update

```
DELETE /rest/ratingupdates/{id}
```

## Parameters

Type	Name	Description	Schema
Path	<b>id</b> <i>optional</i>	Rating update ID	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	No Content

## Produces

- \*/\*

## Tags

- rating-updates-administration

## Security

Type	Name	Scopes
basic	basic	scope

# Recalculate rating for an user

```
PATCH /rest/ratingupdates/{id}/recalculate{?type}
```

## Parameters

Type	Name	Description	Schema
Path	<b>id</b> <i>optional</i>	Avatar ID	integer (int64)
Query	<b>type</b> <i>required</i>	Avatar type	enum (RIDER, DRIVER)

## Responses

HTTP Code	Description	Schema
200	OK	No Content

## Consumes

- application/json

## Produces

- \*/\*

## Tags

- rating-updates-administration

## Security

Type	Name	Scopes
basic	basic	scope

# Update rating update

```
POST /rest/ratingupdates/{id}{?value}
```

## Parameters

Type	Name	Description	Schema
Path	<b>id</b> <i>optional</i>	Rating update ID	integer (int64)
Query	<b>value</b> <i>optional</i>	New rating value	number (double)

## Responses

HTTP Code	Description	Schema
200	OK	<a href="#">RatingUpdateDto</a>

## Consumes

- application/json

## Produces

- \*/\*

## Tags

- rating-updates-administration

## Security

Type	Name	Scopes
basic	<a href="#">basic</a>	scope

## Get cumulative rides report, combining rides stats and ride fulfillment per driver

GET

```
/rest/reports/cumulativeRidesReport{?cityId,completedOnAfter,completedOnBefore,desc,page,pageSize,sort,zipCode}
```

## Parameters

Type	Name	Description	Schema
Query	<b>cityId</b> <i>optional</i>	City ID	integer (int64)
Query	<b>completedOnAfter</b> <i>required</i>	Report start date	string (date-time)
Query	<b>completedOnBefore</b> <i>required</i>	Report end date	string (date-time)
Query	<b>desc</b> <i>optional</i>	Order the result descending	boolean
Query	<b>page</b> <i>optional</i>	Page number	integer (int32)
Query	<b>pageSize</b> <i>optional</i>	Page size	integer (int32)
Query	<b>sort</b> <i>optional</i>	Columns to sort the result by, comma-separated list	< string array(multi) >
Query	<b>zipCode</b> <i>optional</i>	Zip code	string

## Responses

HTTP Code	Description	Schema
200	OK	<a href="#">CumulativeRidesReportEntry</a>

## Produces

- [\\*/\\*](#)

## Tags

- ride-reports

## Security

Type	Name	Scopes
basic	basic	scope

## Get report on ride fulfillment per driver

GET

/rest/reports/driversRidesReport{?cityId,completedOnAfter,completedOnBefore,desc,page,pageSize,sort,zipCode}

## Parameters

Type	Name	Description	Schema
Query	<b>cityId</b> <i>optional</i>	City ID	integer (int64)
Query	<b>completedOnAfter</b> <i>required</i>	Report start date	string (date-time)
Query	<b>completedOnBefore</b> <i>required</i>	Report end date	string (date-time)
Query	<b>desc</b> <i>optional</i>	Order the result descending	boolean
Query	<b>page</b> <i>optional</i>	Page number	integer (int32)
Query	<b>pageSize</b> <i>optional</i>	Page size	integer (int32)
Query	<b>sort</b> <i>optional</i>	Columns to sort the result by, comma-separated list	< string array(multi) >
Query	<b>zipCode</b> <i>optional</i>	Zip code	string

## Responses

HTTP Code	Description	Schema
200	OK	< DriverRidesReport Entry > array

## Produces

- \*/\*

## Tags

- ride-reports

## Security

Type	Name	Scopes
basic	basic	scope

## Get ride stats report (total rides, mileage, cancellations, etc)

```
GET  
/rest/reports/ridesReport{?cityId,completedOnAfter,completedOnBefore,timeZoneOffset,zipCode}
```

## Parameters

Type	Name	Description	Schema
Query	<b>cityId</b> <i>optional</i>	City ID	integer (int64)
Query	<b>completedOn After</b> <i>required</i>	Report start date	string (date-time)
Query	<b>completedOn Before</b> <i>required</i>	Report end date	string (date-time)

Type	Name	Description	Schema
Query	<b>timeZoneOffset</b> <i>optional</i>	Timezone offset	string
Query	<b>zipCode</b> <i>optional</i>	Zip code	string

## Responses

HTTP Code	Description	Schema
200	OK	< RideReportEntry > array

## Produces

- \*/\*

## Tags

- ride-reports

## Security

Type	Name	Scopes
basic	<b>basic</b>	scope

## Get rides report per zip code

```
GET
/rest/reports/ridesZipCodeReport{?completedOnAfter,completedOnBefore,desc,page,pageSize,sort,zipCode}
```

## Parameters

Type	Name	Description	Schema
Query	<b>completedOnAfter</b> <i>required</i>	Report start date	string (date-time)

Type	Name	Description	Schema
Query	<b>completedOnBefore</b> <i>required</i>	Report end date	string (date-time)
Query	<b>desc</b> <i>optional</i>	Order the result descending	boolean
Query	<b>page</b> <i>optional</i>	Page number	integer (int32)
Query	<b>pageSize</b> <i>optional</i>	Page size	integer (int32)
Query	<b>sort</b> <i>optional</i>	Columns to sort the result by, comma-separated list	< string array(multi) >
Query	<b>zipCode</b> <i>optional</i>	Zip code	string

## Responses

HTTP Code	Description	Schema
200	OK	< ZipCodeReportEntry> array

## Produces

- \*/\*

## Tags

- ride-reports

## Security

Type	Name	Scopes
basic	<a href="#">basic</a>	scope

# Get report about users who signed up via promo campaign

GET

/rest/reports/trackingReport{?completedOnAfter,completedOnBefore,desc,page,pageSize,sort}

## Parameters

Type	Name	Description	Schema
Query	<b>completedOnAfter</b> <i>optional</i>	Report start date	string (date-time)
Query	<b>completedOnBefore</b> <i>optional</i>	Report end date	string (date-time)
Query	<b>desc</b> <i>optional</i>	Order the result descending	boolean
Query	<b>page</b> <i>optional</i>	Page number	integer (int32)
Query	<b>pageSize</b> <i>optional</i>	Page size	integer (int32)
Query	<b>sort</b> <i>optional</i>	Columns to sort the result by, comma-separated list	< string array(multi) >

## Responses

HTTP Code	Description	Schema
200	OK	< UserTrackStatsDto > array

## Produces

- \*/\*

## Tags

- user-tracking-stats

## Security

Type	Name	Scopes
basic	basic	scope

## Get report metadata

```
GET /rest/reports/{reportId}
```

## Parameters

Type	Name	Description	Schema
Path	reportId <i>optional</i>	Report ID	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	ReportMetadata
404	Report not found	No Content

## Produces

- \*/\*

## Tags

- reports

## Security

Type	Name	Scopes
basic	basic	scope

# Trigger report execution job

POST /rest/reports/{reportId}/execute

## Parameters

Type	Name	Description	Schema
Path	<b>reportId</b> <i>optional</i>	Report ID	integer (int64)
Body	<b>parametersJs</b> <b>on</b> <i>optional</i>	Report parameters, json-formatted	string

## Responses

HTTP Code	Description	Schema
202	Job is triggered	No Content
500	Failed to trigger job	No Content

## Consumes

- application/json

## Produces

- \*/\*

## Tags

- reports

## Security

Type	Name	Scopes
basic	basic	scope

# Get list of report parameters

```
GET /rest/reports/{reportId}/parameters
```

## Parameters

Type	Name	Description	Schema
Path	<b>reportId</b> <i>optional</i>	Report ID	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	< ReportParameter > array

## Produces

- \*/\*

## Tags

- reports

## Security

Type	Name	Scopes
basic	<b>basic</b>	scope

## Get list of available reports

```
GET /rest/reports{?desc,page,pageSize,reportName,sort}
```

## Parameters

Type	Name	Description	Schema
Query	<b>desc</b> <i>optional</i>	Order the result descending	boolean

Type	Name	Description	Schema
Query	<b>page</b> <i>optional</i>	Page number	integer (int32)
Query	<b>pageSize</b> <i>optional</i>	Page size	integer (int32)
Query	<b>reportName</b> <i>optional</i>		string
Query	<b>sort</b> <i>optional</i>	Columns to sort the result by, comma-separated list	< string array(multi) >

## Responses

HTTP Code	Description	Schema
200	OK	< ReportMetadata > array

## Produces

- \*/\*

## Tags

- reports

## Security

Type	Name	Scopes
basic	<b>basic</b>	scope

## Get current rider information

```
GET /rest/riders/current
```

## Responses

HTTP Code	Description	Schema
200	OK	<a href="#">CurrentRiderDto</a>
403	Current user is not a rider	No Content
500	Error while getting rider cards information from Stripe	No Content

## Produces

- `*/*`

## Tags

- riders

## Security

Type	Name	Scopes
basic	<b>basic</b>	scope

## Export rider list

```
POST /rest/riders/export{?cityId,riderId}
```

## Parameters

Type	Name	Description	Schema
Query	<b>cityId</b> <i>optional</i>	City ID	integer (int64)
Query	<b>riderId</b> <i>optional</i>	Rider ID	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	No Content

HTTP Code	Description	Schema
500	Failed to trigger report job	No Content

## Consumes

- application/json

## Produces

- \*/\*

## Tags

- riders-administration

## Security

Type	Name	Scopes
basic	basic	scope

## Get a paginated list of rider information

```
GET /rest/riders/list{?cityId,desc,page,pageSize,riderId,sort}
```

## Parameters

Type	Name	Description	Schema
Query	<b>cityId</b> <i>optional</i>	City ID	integer (int64)
Query	<b>desc</b> <i>optional</i>	Order the result descending	boolean
Query	<b>page</b> <i>optional</i>	Page number	integer (int32)
Query	<b>pageSize</b> <i>optional</i>	Page size	integer (int32)
Query	<b>riderId</b> <i>optional</i>	Rider ID	integer (int64)

Type	Name	Description	Schema
Query	<b>sort</b> <i>optional</i>	Columns to sort the result by, comma-separated list	< string > array(multi)

## Responses

HTTP Code	Description	Schema
200	OK	< SimpleRiderDto > array

## Produces

- \*/\*

## Tags

- riders-administration

## Security

Type	Name	Scopes
basic	<a href="#">basic</a>	scope

## Get rider information as a rider or an administrator

```
GET /rest/riders/{id}
```

## Parameters

Type	Name	Description	Schema
Path	<b>id</b> <i>optional</i>	Rider ID	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	<a href="#">RiderDto</a>

HTTP Code	Description	Schema
404	Rider not found	No Content

## Produces

- \*/\*

## Tags

- riders-common

## Security

Type	Name	Scopes
basic	basic	scope

# Update rider profile as a rider or an administrator

```
PUT /rest/riders/{id}
```

## Parameters

Type	Name	Description	Schema
Path	<b>id</b> <i>optional</i>	Rider ID	integer (int64)
Body	<b>riderDTO</b> <i>optional</i>	Rider object	<a href="#">RiderDto</a>

## Responses

HTTP Code	Description	Schema
200	OK	<a href="#">RiderDto</a>
400	Provided data is invalid	No Content
404	Rider not found	No Content

## Consumes

- application/json

## Produces

- \*/\*

## Tags

- riders-common

## Security

Type	Name	Scopes
basic	basic	scope

## Get a list of pending payments

```
GET /rest/riders/{id}/payments/pending
```

## Parameters

Type	Name	Description	Schema
Path	<b>id</b> <i>optional</i>	Rider ID	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	< PendingPaymentD to > array
403	User is not permitted to perform the request	No Content

## Produces

- \*/\*

## Tags

- rider-payments

## Security

Type	Name	Scopes
basic	basic	scope

## Process pending payment, pay outstanding balance as a rider

```
POST /rest/riders/{id}/payments/pending{?applePayToken,rideId}
```

## Parameters

Type	Name	Description	Schema
Path	<b>id</b> <i>optional</i>	Rider ID	integer (int64)
Query	<b>applePayToke</b> <b>n</b> <i>optional</i>	Apple pay token	string
Query	<b>rideId</b> <i>optional</i>	Ride ID	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	No Content
400	Failed to process payment	No Content
500	Failed to process payment	No Content

## Consumes

- application/json

## Produces

- `*/*`

## Tags

- rider-payments

## Security

Type	Name	Scopes
basic	basic	scope

## Get payments history as a rider or administrator

```
GET /rest/riders/{id}/payments{?desc,page,pageSize,sort,status}
```

## Parameters

Type	Name	Description	Schema
Path	<b>id</b> <i>optional</i>	Rider ID	integer (int64)
Query	<b>desc</b> <i>optional</i>	Order the result descending	boolean
Query	<b>page</b> <i>optional</i>	Page number	integer (int32)
Query	<b>pageSize</b> <i>optional</i>	Page size	integer (int32)
Query	<b>sort</b> <i>optional</i>	Columns to sort the result by, comma-separated list	< string > array(multi)

Type	Name	Description	Schema
Query	<b>status</b> <i>optional</i>	Ride status	enum (REQUEST_QUEUED, REQUEST_QUEUE_EXPIRED, REQUESTED, RIDER_CANCELLED, DRIVER_ASSIGNED, DRIVER_CANCELLED, DRIVER_REACHED, ACTIVE, NO_AVAILABLE_DRIVER, COMPLETED, ADMIN_CANCELLED )

## Responses

HTTP Code	Description	Schema
200	OK	< <a href="#">PaymentHistoryDt</a> o > array

## Produces

- \*/\*

## Tags

- rider-payments

## Security

Type	Name	Scopes
basic	<a href="#">basic</a>	scope

## Unlock rider's cards

```
POST /rest/riders/{id}/unlock
```

## Parameters

Type	Name	Description	Schema
Path	<b>id</b> <i>optional</i>	Rider ID	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	No Content

## Consumes

- application/json

## Produces

- \*/\*

## Tags

- riders-administration

## Security

Type	Name	Scopes
basic	basic	scope

## List rider's cards

```
GET /rest/riders/{riderId}/cards
```

## Parameters

Type	Name	Description	Schema
Path	<b>riderId</b> <i>optional</i>	Rider ID	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	< RiderCardDto > array
403	User is not allowed to list rider's cards	No Content
404	Rider is not found	No Content
500	Error while communicating with Stripe	No Content

## Produces

- application/json

## Tags

- cards

## Security

Type	Name	Scopes
basic	basic	scope

## Delete a card

```
DELETE /rest/riders/{riderId}/cards/{cardId}
```

## Parameters

Type	Name	Description	Schema
Path	<b>cardId</b> <i>optional</i>	Card ID	integer (int64)
Path	<b>riderId</b> <i>optional</i>	Rider ID	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	No Content
400	User can't delete primary card	No Content
403	User is not allowed to delete rider's card	No Content
404	Rider is not found	No Content
500	Error while communicating with Stripe	No Content

## Produces

- \*/\*

## Tags

- cards

## Security

Type	Name	Scopes
basic	basic	scope

## Update card details

```
PUT /rest/riders/{riderId}/cards/{cardId}{?expMonth,expYear,primary}
```

## Parameters

Type	Name	Description	Schema
Path	cardId <i>optional</i>	Card ID	integer (int64)
Path	riderId <i>optional</i>	Rider ID	integer (int64)

Type	Name	Description	Schema
Query	<b>expMonth</b> <i>optional</i>	Expiration month	string
Query	<b>expYear</b> <i>optional</i>	Expiration year	string
Query	<b>primary</b> <i>required</i>	Set the card as primary source	boolean

## Responses

HTTP Code	Description	Schema
<b>200</b>	OK	No Content
<b>400</b>	Expiration month or year is invalid	No Content
<b>403</b>	User is not allowed to update rider's card	No Content
<b>404</b>	Rider is not found	No Content
<b>500</b>	Error while communicating with Stripe	No Content

## Consumes

- `application/json`

## Produces

- `*/*`

## Tags

- cards

## Security

Type	Name	Scopes
<b>basic</b>	<b>basic</b>	scope

# Add new card

```
POST /rest/riders/{riderId}/cards{?token}
```

## Parameters

Type	Name	Description	Schema
Path	<b>riderId</b> <i>optional</i>	Rider ID	integer (int64)
Query	<b>token</b> <i>optional</i>	Card token generated by Stripe SDK	string

## Responses

HTTP Code	Description	Schema
200	OK	<a href="#">RiderCardDto</a>
403	User is not allowed to list rider's cards	No Content
404	Rider is not found	No Content
500	Error while communicating with Stripe	No Content

## Consumes

- [application/json](#)

## Produces

- [application/json](#)

## Tags

- cards

## Security

Type	Name	Scopes
basic	<b>basic</b>	scope

# Redeem promocode

POST /rest/riders/{riderId}/promocode

## Parameters

Type	Name	Description	Schema
Path	<b>riderId</b> <i>optional</i>	Rider ID	integer (int64)
Body	<b>promocode</b> <i>optional</i>	Promocode object	<a href="#">RedeemPromocodeD to</a>

## Responses

HTTP Code	Description	Schema
200	OK	<a href="#">RiderPromoCodeD to</a>
400	Invalid promocode	No Content

## Consumes

- [application/json](#)

## Produces

- [application/json](#)

## Tags

- rider-promocodes

## Security

Type	Name	Scopes
basic	<a href="#">basic</a>	scope

# Get information about personal promocode

```
GET /rest/riders/{riderId}/promocode
```

## Parameters

Type	Name	Description	Schema
Path	riderId <i>required</i>	riderId	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	<a href="#">RiderPromoCodeD to</a>
400	Invalid promocode	No Content

## Produces

- [application/json](#)

## Tags

- rider-promocodes

## Security

Type	Name	Scopes
basic	<a href="#">basic</a>	scope

## Get a list of promocode redemptions

```
GET /rest/riders/{riderId}/promocode/redemptions
```

## Parameters

Type	Name	Description	Schema
Path	riderId <i>required</i>	riderId	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	< PromocodeRedemptionDTO > array
403	Rider is not permitted to view other rider's redemption history	No Content

## Produces

- application/json

## Tags

- rider-promocodes

## Security

Type	Name	Scopes
basic	basic	scope

## Get remaining free credit

```
GET /rest/riders/{riderId}/promocode/remainder
```

## Parameters

Type	Name	Description	Schema
Path	riderId <i>optional</i>	Rider ID	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	RemainderResponse

HTTP Code	Description	Schema
403	Rider is not permitted to view other rider's remainder	No Content

## Produces

- application/json

## Tags

- rider-promocodes

## Security

Type	Name	Scopes
basic	basic	scope

## Submit feedback as a rider after cancelling the ride

```
POST /rest/rides/cancellation/{rideId}{?comment,reason}
```

## Parameters

Type	Name	Description	Schema
Path	<b>rideId</b> <i>optional</i>	Ride ID	integer (int64)
Query	<b>comment</b> <i>optional</i>	Cancellation comment	string
Query	<b>reason</b> <i>required</i>	Cancellation reason code	enum (CHANGE_BOOKING, CHANGE_MIND, ANOTHER_RIDE, MISTAKE, TOO_LONG)

## Responses

HTTP Code	Description	Schema
200	OK	No Content
400	Feedback can't be submitted	No Content

## Consumes

- application/json

## Produces

- \*/\*

## Tags

- ride-cancellation-feedback

## Security

Type	Name	Scopes
basic	basic	scope

## List available cancellation reasons

```
GET /rest/rides/cancellation{?avatarType,cityId}
```

## Parameters

Type	Name	Description	Schema	Default
Query	avatarType <i>optional</i>	Avatar type	enum (RIDER, DRIVER)	"RIDER"
Query	cityId <i>optional</i>	City ID	integer (int64)	1

## Responses

HTTP Code	Description	Schema
200	OK	< CancellationReasonDto > array

## Produces

- \*/\*

## Tags

- ride-cancellation-feedback

## Security

Type	Name	Scopes
basic	basic	scope

## Get current ride as a dispatcher

```
GET /rest/rides/current?avatarType=DISPATCHER
```

## Parameters

Type	Name	Schema	Default
Query	avatarType <i>required</i>	enum (DISPATCHER)	"DISPATCHER"

## Responses

HTTP Code	Description	Schema
200	OK	< DispatcherAccountRideDto > array
403	Requesting rider is not a dispatcher	No Content

## Produces

- application/json

## Tags

- rides

## Security

Type	Name	Scopes
basic	basic	scope

## Get current ride as a driver

```
GET /rest/rides/current?avatarType=DRIVER
```

## Parameters

Type	Name	Schema	Default
Query	avatarType <i>required</i>	enum (DRIVER)	"DRIVER"

## Responses

HTTP Code	Description	Schema
200	OK	MobileDriverRide Dto
400	Current driver is offline	No Content

## Produces

- application/json

## Tags

- rides

## Security

Type	Name	Scopes
basic	basic	scope

## Get current ride as a driver

```
GET /rest/rides/current?avatarType=RIDER
```

## Parameters

Type	Name	Schema	Default
Query	avatarType <i>required</i>	enum (RIDER)	"RIDER"

## Responses

HTTP Code	Description	Schema
200	OK	MobileRiderRideD to

## Produces

- application/json

## Tags

- rides

## Security

Type	Name	Scopes
basic	basic	scope

## Fare estimate

```
GET /rest/rides/estimate{?carCategory,cityId,endLat,endLong,startLat,startLong}
```

## Description

Estimates fare between start and end location. No value should be provided for cityId to estimate in Austin. No value should be provided for carCategory if we estimate for STANDARD car category

## Parameters

Type	Name	Description	Schema	Default
Query	<b>carCategory</b> <i>optional</i>	Car category	string	"REGULAR" "
Query	<b>cityId</b> <i>optional</i>	City ID	integer (int64)	1
Query	<b>endLat</b> <i>required</i>	End location latitude	number (double)	
Query	<b>endLong</b> <i>required</i>	End location longitude	number (double)	
Query	<b>startLat</b> <i>required</i>	Start location latitude	number (double)	
Query	<b>startLong</b> <i>required</i>	Start location longitude	number (double)	

## Responses

HTTP Code	Description	Schema
200	OK	<a href="#">EstimatedFareDTO</a>
400	Car category or city is invalid; estimation failed	No Content

## Produces

- [application/json](#)

## Tags

- rides

## Security

Type	Name	Scopes
basic	basic	scope

## Process events cached on app side

POST /rest/rides/events

## Parameters

Type	Name	Description	Schema
Body	<b>rideEvents</b> <i>required</i>	rideEvents	RideEvents

## Responses

HTTP Code	Description	Schema
200	OK	object

## Consumes

- application/json

## Produces

- application/json

## Tags

- rides

## Security

Type	Name	Scopes
basic	basic	scope

# Trigger extended rides report. Same as POST /rest/rides/report, include all the statuses, email to current user

```
POST  
/rest/rides/export{?cancelledOnAfter,cancelledOnBefore,charged,cityId,completedOnAfter,  
,completedOnBefore,createdOnAfter,createdOnBefore,driverEmail,driverId,phoneNumber,riderEmail,riderId,status,zipCode}
```

## Parameters

Type	Name	Description	Schema
Query	<b>cancelledOnAfter</b> <i>optional</i>	Rides cancelled after	string (date-time)
Query	<b>cancelledOnBefore</b> <i>optional</i>	Rides cancelled before	string (date-time)
Query	<b>charged</b> <i>optional</i>	Show charged rides only	boolean
Query	<b>cityId</b> <i>optional</i>	City ID	integer (int64)
Query	<b>completedOnAfter</b> <i>optional</i>	Rides completed after	string (date-time)
Query	<b>completedOnBefore</b> <i>optional</i>	Rides completed before	string (date-time)
Query	<b>createdOnAfter</b> <i>optional</i>	Rides created after	string (date-time)
Query	<b>createdOnBefore</b> <i>optional</i>	Rides created before	string (date-time)

Type	Name	Description	Schema
Query	<b>driverEmail</b> <i>optional</i>	Driver email	string
Query	<b>driverId</b> <i>optional</i>	Driver ID	integer (int64)
Query	<b>phoneNumber</b> <i>optional</i>	Rider phone number	string
Query	<b>riderEmail</b> <i>optional</i>	Rider email	string
Query	<b>riderId</b> <i>optional</i>	Rider ID	integer (int64)
Query	<b>status</b> <i>optional</i>	List of ride statuses to be included	< enum (REQUEST_QUEUED, REQUEST_QUEUE_EXPIRED, REQUESTED, RIDER_CANCELLED, DRIVER_ASSIGNED, DRIVER_CANCELLED, DRIVER_REACHED, ACTIVE, NO_AVAILABLE_DRIVER, COMPLETED, ADMIN_CANCELLED ) > array(multi)
Query	<b>zipCode</b> <i>optional</i>	Zip code	string

## Responses

HTTP Code	Description	Schema
200	OK	No Content
500	Server is either misconfigured, or failed to trigger report job	No Content

## Consumes

- application/json

## Produces

- \*/\*

## Tags

- ride-reports

## Security

Type	Name	Scopes
basic	basic	scope

## Get history of Direct Connect rides

```
GET /rest/rides/history/direct{?riderId}
```

## Parameters

Type	Name	Description	Schema
Query	riderId <i>optional</i>	Rider ID	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	< DirectConnectHistoryDto> array

## Produces

- \*/\*

## Tags

- rides

## Security

Type	Name	Scopes
basic	basic	scope

## Get last unrated ride as a rider

```
GET /rest/rides/last
```

## Responses

HTTP Code	Description	Schema
200	OK	<a href="#">MobileRiderRideD to</a>

## Produces

- \*/\*

## Tags

- rides

## Security

Type	Name	Scopes
basic	basic	scope

## getLastRides

```
GET /rest/rides/last?avatarType=DISPATCHER
```

## Parameters

Type	Name	Schema	Default
Query	avatarType <i>required</i>	enum (DISPATCHER)	"DISPATCHER"

## Responses

HTTP Code	Description	Schema
200	OK	< DispatcherAccoun tRideDto > array

## Produces

- \*/\*

## Tags

- rides

## Security

Type	Name	Scopes
basic	basic	scope

## Get a list of rides (extended format) as an administrator

GET

```
/rest/rides/list{?cancelledOnAfter,cancelledOnBefore,charged,cityId,completedOnAfter,c  
ompletedOnBefore,createdOnAfter,createdOnBefore,desc,driverEmail,driverId,page,pageSize,phoneNumber,riderEmail,riderId,sort,status,zipCode}
```

## Parameters

Type	Name	Description	Schema
Query	<b>cancelledOnA fter</b> <i>optional</i>	Rides cancelled after	string (date-time)
Query	<b>cancelledOnB efore</b> <i>optional</i>	Rides cancelled before	string (date-time)

Type	Name	Description	Schema
Query	<b>charged</b> <i>optional</i>	Show charged rides only	boolean
Query	<b>cityId</b> <i>optional</i>	City ID	integer (int64)
Query	<b>completedOnAfter</b> <i>optional</i>	Rides completed after	string (date-time)
Query	<b>completedOnBefore</b> <i>optional</i>	Rides completed before	string (date-time)
Query	<b>createdOnAfter</b> <i>optional</i>	Rides created after	string (date-time)
Query	<b>createdOnBefore</b> <i>optional</i>	Rides created before	string (date-time)
Query	<b>desc</b> <i>optional</i>	Order the result descending	boolean
Query	<b>driverEmail</b> <i>optional</i>	Driver email	string
Query	<b>driverId</b> <i>optional</i>	Driver ID	integer (int64)
Query	<b>page</b> <i>optional</i>	Page number	integer (int32)
Query	<b>pageSize</b> <i>optional</i>	Page size	integer (int32)
Query	<b>phoneNumber</b> <i>optional</i>	Rider phone number	string
Query	<b>riderEmail</b> <i>optional</i>	Rider email	string

Type	Name	Description	Schema
Query	<b>riderId</b> <i>optional</i>	Rider ID	integer (int64)
Query	<b>sort</b> <i>optional</i>	Columns to sort the result by, comma-separated list	< string > array(multi)
Query	<b>status</b> <i>optional</i>	List of ride statuses to be included	< enum (REQUEST_QUEUED, REQUEST_QUEUE_EXPIRED, REQUESTED, RIDER_CANCELLED, DRIVER_ASSIGNED, DRIVER_CANCELLED, DRIVER_REACHED, ACTIVE, NO_AVAILABLE_DRIVER, COMPLETED, ADMIN_CANCELLED ) > array(multi)
Query	<b>zipCode</b> <i>optional</i>	Zip code	string

## Responses

HTTP Code	Description	Schema
200	OK	< <a href="#">ExtendedRideDto</a> > array

## Produces

- [application/json](#)

## Tags

- rides-administration

## Security

Type	Name	Scopes
basic	basic	scope

## Get map information

```
GET /rest/rides/map{?cityId}
```

## Parameters

Type	Name	Description	Schema	Default
Query	cityId <i>required</i>	City ID	integer (int64)	1

## Responses

HTTP Code	Description	Schema
200	OK	< MapInfoDto > array

## Produces

- \*/\*

## Tags

- rides-administration

## Security

Type	Name	Scopes
basic	basic	scope

## Associate a ride with a queued token as a rider

```
POST /rest/rides/queue/{token}
```

## Parameters

Type	Name	Description	Schema
Path	<b>token</b> <i>optional</i>	Queued ride token	string

## Responses

HTTP Code	Description	Schema
<b>200</b>	OK	<a href="#">MobileRiderRideD to</a>
<b>400</b>	Ride token expired	No Content
<b>404</b>	Ride token not found	No Content

## Consumes

- [application/json](#)

## Produces

- [\\*/\\*](#)

## Tags

- ride-flow

## Security

Type	Name	Scopes
basic	<a href="#">basic</a>	scope

## Trigger extended rides report

POST

```
/rest/rides/report{?cancelledOnAfter,cancelledOnBefore,charged,cityId,completedOnAfter,  
,completedOnBefore,createdOnAfter,createdOnBefore,driverEmail,driverId,phoneNumber,rec  
ipient,riderEmail,riderId,status,zipCode}
```

## Parameters

Type	Name	Description	Schema
Query	<b>cancelledOnAfter</b> <i>optional</i>	Rides cancelled after	string (date-time)
Query	<b>cancelledOnBefore</b> <i>optional</i>	Rides cancelled before	string (date-time)
Query	<b>charged</b> <i>optional</i>	Show charged rides only	boolean
Query	<b>cityId</b> <i>optional</i>	City ID	integer (int64)
Query	<b>completedOnAfter</b> <i>optional</i>	Rides completed after	string (date-time)
Query	<b>completedOnBefore</b> <i>optional</i>	Rides completed before	string (date-time)
Query	<b>createdOnAfter</b> <i>optional</i>	Rides created after	string (date-time)
Query	<b>createdOnBefore</b> <i>optional</i>	Rides created before	string (date-time)
Query	<b>driverEmail</b> <i>optional</i>	Driver email	string
Query	<b>driverId</b> <i>optional</i>	Driver ID	integer (int64)
Query	<b>phoneNumber</b> <i>optional</i>	Rider phone number	string
Query	<b>recipient</b> <i>optional</i>	List of emails to receive the report	< string > array(multi)

Type	Name	Description	Schema
Query	<b>riderEmail</b> <i>optional</i>	Rider email	string
Query	<b>riderId</b> <i>optional</i>	Rider ID	integer (int64)
Query	<b>status</b> <i>optional</i>	List of ride statuses to be included	< enum (REQUEST_QUEUED, REQUEST_QUEUE_EXPIRED, REQUESTED, RIDER_CANCELLED, DRIVER_ASSIGNED, DRIVER_CANCELLED, DRIVER_REACHED, ACTIVE, NO_AVAILABLE_DRIVER, COMPLETED, ADMIN_CANCELLED ) > array(multi)
Query	<b>status</b> <i>optional</i>	Statuses to include in report	< enum (REQUEST_QUEUED, REQUEST_QUEUE_EXPIRED, REQUESTED, RIDER_CANCELLED, DRIVER_ASSIGNED, DRIVER_CANCELLED, DRIVER_REACHED, ACTIVE, NO_AVAILABLE_DRIVER, COMPLETED, ADMIN_CANCELLED ) > array(multi)
Query	<b>zipCode</b> <i>optional</i>	Zip code	string

## Responses

HTTP Code	Description	Schema
200	OK	No Content
500	Server is either misconfigured, or failed to trigger report job	No Content

## Consumes

- application/json

## Produces

- \*/\*

## Tags

- ride-reports

## Security

Type	Name	Scopes
basic	basic	scope

## specialFees

```
GET /rest/rides/specialFees{?startLat,startLong}
```

## Description

Set of special fees - applicable for specified pickup location.

## Parameters

Type	Name	Description	Schema
Query	<b>startLat</b> <i>required</i>	Latitude	number (double)
Query	<b>startLong</b> <i>required</i>	Longitude	number (double)

## Responses

HTTP Code	Description	Schema
200	OK	< SpecialFee > array

## Produces

- application/json

## Tags

- rides

## Security

Type	Name	Scopes
basic	basic	scope

## Accept ride upgrade as a rider

```
POST /rest/rides/upgrade/accept
```

## Responses

HTTP Code	Description	Schema
200	OK	RideUpgradeResponse
403	Requesting user is not a rider	No Content
404	Upgrade request not found	No Content

## Consumes

- application/json

## Produces

- \*/\*

## Tags

- ride-upgrades

## Security

Type	Name	Scopes
basic	basic	scope

## Decline ride upgrade as a dispatcher

```
POST /rest/rides/upgrade/decline?avatarType=DISPATCHER
```

## Parameters

Type	Name	Schema	Default
Query	avatarType <i>required</i>	enum (DISPATCHER)	"DISPATCHER"

## Responses

HTTP Code	Description	Schema
200	OK	RideUpgradeResponse
403	Requesting user is not a rider	No Content
404	Upgrade request not found	No Content

## Consumes

- application/json

## Produces

- \*/\*

## Tags

- ride-upgrades

## Security

Type	Name	Scopes
basic	basic	scope

## Cancel ride upgrade as a driver

```
POST /rest/rides/upgrade/decline?avatarType=DRIVER
```

## Parameters

Type	Name	Schema	Default
Query	<b>avatarType</b> <i>required</i>	enum (DRIVER)	"DRIVER"

## Responses

HTTP Code	Description	Schema
200	OK	RideUpgradeResponse
403	Requesting user is not a driver	No Content
404	Upgrade request not found	No Content

## Consumes

- application/json

## Produces

- \*/\*

## Tags

- ride-upgrades

## Security

Type	Name	Scopes
basic	basic	scope

## Decline ride upgrade as a rider

POST /rest/rides/upgrade/decline?avatarType=RIDER

### Parameters

Type	Name	Schema	Default
Query	<b>avatarType</b> <i>required</i>	enum (RIDER)	"RIDER"

### Responses

HTTP Code	Description	Schema
200	OK	RideUpgradeResponse
403	Requesting user is not a rider	No Content
404	Upgrade request not found	No Content

### Consumes

- application/json

### Produces

- \*/\*

### Tags

- ride-upgrades

### Security

Type	Name	Scopes
basic	basic	scope

# Request ride upgrade as a driver

```
POST /rest/rides/upgrade/request?target=SUV
```

## Parameters

Type	Name	Description	Schema
Query	<b>target</b> <i>required</i>	Target car type	enum (SUV)

## Responses

HTTP Code	Description	Schema
200	OK	RideUpgradeResp onse
400	Upgrade can not be requested	No Content

## Consumes

- application/json

## Produces

- \*/\*

## Tags

- ride-upgrades

## Security

Type	Name	Scopes
basic	basic	scope

# Accept ride request

```
POST /rest/rides/{id}/accept
```

## Parameters

Type	Name	Description	Schema
Path	<b>id</b> <i>optional</i>	Ride ID	integer (int64)

## Responses

HTTP Code	Description	Schema
<b>200</b>	OK	object
<b>400</b>	Failed to process request	No Content

## Consumes

- application/json

## Produces

- application/json

## Tags

- ride-flow

## Security

Type	Name	Scopes
basic	<b>basic</b>	scope

## Decline ride request

```
DELETE /rest/rides/{id}/decline
```

## Parameters

Type	Name	Description	Schema
Path	<b>id</b> <i>optional</i>	Ride ID	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	object
400	Failed to process request	No Content

## Produces

- application/json

## Tags

- ride-flow

## Security

Type	Name	Scopes
basic	basic	scope

## End the ride

```
POST  
/rest/rides/{id}/end{?endAddress,endGooglePlaceId,endLocationLat,endLocationLong,endZipCode}
```

## Parameters

Type	Name	Description	Schema
Path	<b>id</b> <i>optional</i>	Ride ID	integer (int64)
Query	<b>endAddress</b> <i>optional</i>	End location address	string
Query	<b>endGooglePlaceId</b> <i>optional</i>	End location Google Place ID	string

Type	Name	Description	Schema
Query	<b>endLocationLat</b> <i>required</i>	End location latitude	number (double)
Query	<b>endLocationLong</b> <i>required</i>	End location longitude	number (double)
Query	<b>endZipCode</b> <i>optional</i>	End location zipcode	string

## Responses

HTTP Code	Description	Schema
200	OK	<a href="#">MobileDriverRide Dto</a>
400	Failed to process request	No Content

## Consumes

- [application/json](#)

## Produces

- [application/json](#)

## Tags

- ride-flow

## Security

Type	Name	Scopes
basic	<a href="#">basic</a>	scope

## Get sharing token for a ride

```
POST /rest/rides/{id}/getShareToken
```

## Parameters

Type	Name	Description	Schema
Path	<b>id</b> <i>optional</i>	Ride ID	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	TrackingShareToken
403	Ride doesn't belong requesting rider	No Content
404	Ride not found	No Content

## Consumes

- application/json

## Produces

- application/json

## Tags

- ride-share-tracking

## Security

Type	Name	Scopes
basic	basic	scope

## Get ride map URL

```
GET /rest/rides/{id}/map
```

## Parameters

Type	Name	Description	Schema
Path	<b>id</b> <i>optional</i>	Ride ID	integer (int64)

## Responses

HTTP Code	Description	Schema
<b>200</b>	OK	<a href="#">UrlDto</a>
<b>403</b>	Requesting rider is not an owner of the ride	No Content
<b>404</b>	Ride not found	No Content

## Produces

- [application/json](#)

## Tags

- rides

## Security

Type	Name	Scopes
basic	<a href="#">basic</a>	scope

## Recreate missing ride map

```
PUT /rest/rides/{id}/map/recreate
```

## Parameters

Type	Name	Description	Schema
Path	<b>id</b> <i>optional</i>	Ride ID	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	No Content
400	Ride is not completed	No Content

## Consumes

- application/json

## Produces

- application/json

## Tags

- rides-administration

## Security

Type	Name	Scopes
basic	basic	scope

## Submit ride rating and/or tip, and/or comment

```
PUT /rest/rides/{id}/rating{?comment,paymentProvider,rating,tip}
```

## Parameters

Type	Name	Description	Schema
Path	<b>id</b> <i>optional</i>	Ride ID	integer (int64)
Query	<b>comment</b> <i>optional</i>	Ride completion comment	string
Query	<b>paymentProvider</b> <i>optional</i>	Payment provider used to pay for ride	enum (CREDIT_CARD, APPLE_PAY)
Query	<b>rating</b> <i>required</i>	Rating value	number

Type	Name	Description	Schema
Query	<b>tip</b> <i>optional</i>	Ride tip amount	number

## Responses

HTTP Code	Description	Schema
200	OK	<a href="#">MobileRiderRideD to</a>
400	Failed to process request	No Content

## Consumes

- [application/json](#)

## Produces

- [application/json](#)

## Tags

- ride-flow

## Security

Type	Name	Scopes
basic	<a href="#">basic</a>	scope

## Mark ride request as DRIVER\_REACHED

```
POST /rest/rides/{id}/reached
```

## Parameters

Type	Name	Description	Schema
Path	<b>id</b> <i>optional</i>	Ride ID	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	object
400	Failed to process request	No Content

## Consumes

- application/json

## Produces

- application/json

## Tags

- ride-flow

## Security

Type	Name	Scopes
basic	basic	scope

## Resend a receipt

```
POST /rest/rides/{id}/receipt
```

## Parameters

Type	Name	Description	Schema
Path	<b>id</b> <i>required</i>	id	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	No Content

HTTP Code	Description	Schema
202	Accepted	No Content
500	Failed to send email	No Content

## Consumes

- application/json

## Produces

- \*/\*

## Tags

- rides-administration

## Security

Type	Name	Scopes
basic	basic	scope

# Acknowledge handshake received by driver app for a ride

```
POST /rest/rides/{id}/received
```

## Parameters

Type	Name	Description	Schema
Path	<b>id</b> <i>optional</i>	Ride ID	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	object

HTTP Code	Description	Schema
400	Failed to process request	No Content

## Consumes

- application/json

## Produces

- application/json

## Tags

- ride-flow

## Security

Type	Name	Scopes
basic	basic	scope

## Send an email with a link to share the ride route

```
POST /rest/rides/{id}/shareToEmail{?emailAddress}
```

## Parameters

Type	Name	Description	Schema
Path	<b>id</b> <i>optional</i>	Ride ID	integer (int64)
Query	<b>emailAddress</b> <i>required</i>	Recipient email	string

## Responses

HTTP Code	Description	Schema
200	OK	No Content

HTTP Code	Description	Schema
403	Ride doesn't belong requesting rider	No Content
404	Ride not found	No Content
500	Failed to send email	No Content

## Consumes

- application/json

## Produces

- application/json

## Tags

- ride-share-tracking

## Security

Type	Name	Scopes
basic	basic	scope

## Start the ride

```
POST /rest/rides/{id}/start
```

## Parameters

Type	Name	Description	Schema
Path	<b>id</b> <i>optional</i>	Ride ID	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	object

HTTP Code	Description	Schema
400	Failed to process request	No Content

## Consumes

- application/json

## Produces

- application/json

## Tags

- ride-flow

## Security

Type	Name	Scopes
basic	basic	scope

## Get a ride object as an administrator

```
GET /rest/rides/{id}?avatarType=ADMIN
```

## Parameters

Type	Name	Description	Schema	Default
Path	<b>id</b> <i>optional</i>	Ride ID	integer (int64)	
Query	<b>avatarType</b> <i>required</i>		enum (ADMIN)	"ADMIN"

## Responses

HTTP Code	Description	Schema
200	OK	ConsoleRideDto

## Produces

- application/json

## Tags

- rides-administration

## Security

Type	Name	Scopes
basic	basic	scope

## Cancel the ride as an administrator

```
DELETE /rest/rides/{id}?avatarType=ADMIN
```

## Parameters

Type	Name	Description	Schema	Default
Path	<b>id</b> <i>optional</i>	Ride ID	integer (int64)	
Query	<b>avatarType</b> <i>required</i>		enum (ADMIN)	"ADMIN"

## Responses

HTTP Code	Description	Schema
200	OK	No Content
400	Failed to process request	No Content

## Produces

- application/json

## Tags

- ride-flow-administration

## Security

Type	Name	Scopes
basic	basic	scope

## Get ride object as a dispatcher

```
GET /rest/rides/{id}?avatarType=DISPATCHER
```

## Parameters

Type	Name	Description	Schema	Default
Path	<b>id</b> <i>optional</i>	Ride ID	integer (int64)	
Query	<b>avatarType</b> <i>required</i>		enum (DISPATCHER)	"DISPATCHER"

## Responses

HTTP Code	Description	Schema
200	OK	DispatcherAccountRideDto
403	Requesting rider is not an owner of the ride	No Content

## Produces

- application/json

## Tags

- rides

## Security

Type	Name	Scopes
basic	basic	scope

# Cancel ride as a dispatcher

```
DELETE /rest/rides/{id}?avatarType=DISPATCHER
```

## Parameters

Type	Name	Description	Schema	Default
Path	<b>id</b> <i>optional</i>	Ride ID	integer (int64)	
Query	<b>avatarType</b> <i>required</i>		enum (DISPATCHER)	"DISPATCHER"

## Responses

HTTP Code	Description	Schema
200	OK	object
400	Failed to process request	No Content

## Produces

- application/json

## Tags

- ride-flow

## Security

Type	Name	Scopes
basic	<b>basic</b>	scope

# Get ride object as a driver

```
GET /rest/rides/{id}?avatarType=DRIVER
```

## Parameters

Type	Name	Description	Schema	Default
Path	<b>id</b> <i>optional</i>	Ride ID	integer (int64)	
Query	<b>avatarType</b> <i>required</i>		enum (DRIVER)	"DRIVER"

## Responses

HTTP Code	Description	Schema
200	OK	<a href="#">MobileDriverRide Dto</a>

## Produces

- [application/json](#)

## Tags

- rides

## Security

Type	Name	Scopes
basic	<a href="#">basic</a>	scope

## Cancel ride as a driver

```
DELETE /rest/rides/{id}?avatarType=DRIVER{&comment,reason}
```

## Parameters

Type	Name	Description	Schema	Default
Path	<b>id</b> <i>optional</i>	Ride ID	integer (int64)	

Type	Name	Description	Schema	Default
Query	<b>avatarType</b> <i>required</i>		enum (DRIVER)	"DRIVER"
Query	<b>comment</b> <i>optional</i>	Cancellation comment	string	
Query	<b>reason</b> <i>optional</i>	Cancellation reason code	enum (NO_SHOW, WRONG_GPS, TOO_MANY_RIDE RS)	

## Responses

HTTP Code	Description	Schema
200	OK	object
400	Failed to process request	No Content

## Produces

- application/json

## Tags

- ride-flow

## Security

Type	Name	Scopes
basic	basic	scope

## Cancel ride as a rider

```
DELETE /rest/rides/{id}?avatarType=RIDER
```

## Parameters

Type	Name	Description	Schema	Default
Path	<b>id</b> <i>optional</i>	Ride ID	integer (int64)	
Query	<b>avatarType</b> <i>required</i>		enum (RIDER)	"RIDER"

## Responses

HTTP Code	Description	Schema
200	OK	object
400	Failed to process request	No Content

## Produces

- application/json

## Tags

- ride-flow

## Security

Type	Name	Scopes
basic	basic	scope

## Get ride object as a rider

```
GET /rest/rides/{id}?avatarType=RIDER{&lat,lng}
```

## Parameters

Type	Name	Description	Schema	Default
Path	<b>id</b> <i>optional</i>	Ride ID	integer (int64)	
Query	<b>avatarType</b> <i>required</i>		enum (RIDER)	"RIDER"

Type	Name	Description	Schema	Default
Query	<code>lat required</code>	Current GPS latitude	number (double)	
Query	<code>lng required</code>	Current GPS longitude	number (double)	

## Responses

HTTP Code	Description	Schema
<b>200</b>	OK	<a href="#">MobileRiderRideD to</a>
<b>403</b>	Requesting rider is not an owner of the ride	No Content

## Produces

- [application/json](#)

## Tags

- rides

## Security

Type	Name	Scopes
basic	<a href="#">basic</a>	scope

## Update ride destination and/or comment

```
PUT
/rest/rides/{id}{?comment,endAddress,endGooglePlaceId,endLocationLat,endLocationLong,endZipCode}
```

## Parameters

Type	Name	Description	Schema
Path	<code>id optional</code>	Ride ID	integer (int64)

Type	Name	Description	Schema
Query	<b>comment</b> <i>optional</i>	New ride comment	string
Query	<b>endAddress</b> <i>optional</i>	End location address	string
Query	<b>endGooglePlaceId</b> <i>optional</i>	End location Google Place ID	string
Query	<b>endLocationLat</b> <i>required</i>	End location latitude	number (double)
Query	<b>endLocationLong</b> <i>required</i>	End location longitude	number (double)
Query	<b>endZipCode</b> <i>optional</i>	End location zipcode	string

## Responses

HTTP Code	Description	Schema
200	OK	object
400	Failed to process request	No Content

## Consumes

- application/json

## Produces

- application/json

## Tags

- ride-flow

## Security

Type	Name	Scopes
basic	basic	scope

## Get current ride track for web sharing console page

```
GET /rest/rides/{trackingKey}/allTrackers
```

## Parameters

Type	Name	Description	Schema
Path	trackingKey <i>optional</i>	Sharing tracking key	string

## Responses

HTTP Code	Description	Schema
200	OK	RideTrackingShareDto
404	Ride not found	No Content

## Produces

- \*/\*

## Tags

- ride-share-tracking

## Security

Type	Name	Scopes
basic	basic	scope

# Get a list of rides (compact format) as an administrator

GET

```
/rest/rides?avatarType=ADMIN&format=compact{&cancelledOnAfter,cancelledOnBefore,charge  
d,cityId,completedOnAfter,completedOnBefore,createdOnAfter,createdOnBefore,desc,driver  
Email,driverId,page,pageSize,phoneNumber,riderEmail,riderId,sort,status,zipCode}
```

## Parameters

Type	Name	Description	Schema	Default
Query	<b>avatarType</b> <i>required</i>		enum (ADMIN)	"ADMIN"
Query	<b>cancelledOnAfter</b> <i>optional</i>	Rides cancelled after	string (date-time)	
Query	<b>cancelledOnBefore</b> <i>optional</i>	Rides cancelled before	string (date-time)	
Query	<b>charged</b> <i>optional</i>	Show charged rides only	boolean	
Query	<b>cityId</b> <i>optional</i>	City ID	integer (int64)	
Query	<b>completedOnAfter</b> <i>optional</i>	Rides completed after	string (date-time)	
Query	<b>completedOnBefore</b> <i>optional</i>	Rides completed before	string (date-time)	
Query	<b>createdOnAfter</b> <i>optional</i>	Rides created after	string (date-time)	
Query	<b>createdOnBefore</b> <i>optional</i>	Rides created before	string (date-time)	

Type	Name	Description	Schema	Default
Query	<b>desc</b> <i>optional</i>	Order the result descending	boolean	
Query	<b>driverEmail</b> <i>optional</i>	Driver email	string	
Query	<b>driverId</b> <i>optional</i>	Driver ID	integer (int64)	
Query	<b>format</b> <i>required</i>		enum (compact)	"compact" "
Query	<b>page</b> <i>optional</i>	Page number	integer (int32)	
Query	<b>pageSize</b> <i>optional</i>	Page size	integer (int32)	
Query	<b>phoneNum ber</b> <i>optional</i>	Rider phone number	string	
Query	<b>riderEmail</b> <i>optional</i>	Rider email	string	
Query	<b>riderId</b> <i>optional</i>	Rider ID	integer (int64)	
Query	<b>sort</b> <i>optional</i>	Columns to sort the result by, comma-separated list	< string array(multi) >	

Type	Name	Description	Schema	Default
Query	<b>status</b> <i>optional</i>	List of ride statuses to be included	< enum (REQUEST_QUEUE_D, REQUEST_QUEUE_EXPIRED, REQUESTED, RIDER_CANCELLED, DRIVER_ASSIGNED, DRIVER_CANCELLED, DRIVER_REACHED, , ACTIVE, NO_AVAILABLE_DRIVER, COMPLETED, ADMIN_CANCELLED) > array(multi)	
Query	<b>zipCode</b> <i>optional</i>	Zip code	string	

## Responses

HTTP Code	Description	Schema
200	OK	< CompactRideDto > array

## Produces

- application/json

## Tags

- rides-administration

## Security

Type	Name	Scopes
basic	<b>basic</b>	scope

# Get a list of rides (history format) as an administrator

GET

```
/rest/rides?avatarType=ADMIN{&cancelledOnAfter,cancelledOnBefore,charged,cityId,completedOnAfter,completedOnBefore,createdOnAfter,createdOnBefore,desc,driverEmail,driverId,page,pageSize,phoneNumber,riderEmail,riderId,sort,status,zipCode}
```

## Parameters

Type	Name	Description	Schema	Default
Query	<b>avatarType</b> <i>required</i>		enum (ADMIN)	"ADMIN"
Query	<b>cancelledOnAfter</b> <i>optional</i>	Rides cancelled after	string (date-time)	
Query	<b>cancelledOnBefore</b> <i>optional</i>	Rides cancelled before	string (date-time)	
Query	<b>charged</b> <i>optional</i>	Show charged rides only	boolean	
Query	<b>cityId</b> <i>optional</i>	City ID	integer (int64)	
Query	<b>completedOnAfter</b> <i>optional</i>	Rides completed after	string (date-time)	
Query	<b>completedOnBefore</b> <i>optional</i>	Rides completed before	string (date-time)	
Query	<b>createdOnAfter</b> <i>optional</i>	Rides created after	string (date-time)	
Query	<b>createdOnBefore</b> <i>optional</i>	Rides created before	string (date-time)	

Type	Name	Description	Schema	Default
Query	<b>desc</b> <i>optional</i>	Order the result descending	boolean	
Query	<b>driverEmail</b> <i>optional</i>	Driver email	string	
Query	<b>driverId</b> <i>optional</i>	Driver ID	integer (int64)	
Query	<b>page</b> <i>optional</i>	Page number	integer (int32)	
Query	<b>pageSize</b> <i>optional</i>	Page size	integer (int32)	
Query	<b>phoneNumbr</b> <i>optional</i>	Rider phone number	string	
Query	<b>riderEmail</b> <i>optional</i>	Rider email	string	
Query	<b>riderId</b> <i>optional</i>	Rider ID	integer (int64)	
Query	<b>sort</b> <i>optional</i>	Columns to sort the result by, comma-separated list	< string array(multi) >	

Type	Name	Description	Schema	Default
Query	<b>status</b> <i>optional</i>	List of ride statuses to be included	< enum (REQUEST_QUEUE_D, REQUEST_QUEUE_EXPIRED, REQUESTED, RIDER_CANCELLED, DRIVER_ASSIGNED, DRIVER_CANCELLED, DRIVER_REACHED, , ACTIVE, NO_AVAILABLE_RIVER, COMPLETED, ADMIN_CANCELLED) > array(multi)	
Query	<b>zipCode</b> <i>optional</i>	Zip code	string	

## Responses

HTTP Code	Description	Schema
200	OK	< RideHistoryDto > array

## Produces

- application/json

## Tags

- rides-administration

## Security

Type	Name	Scopes
basic	<b>basic</b>	scope

# Request a ride

POST

```
/rest/rides{?applePayToken,avatarType,carCategory,cityId,comment,directConnectId,drive  
rType,endAddress,endGooglePlaceId,endLocationLat,endLocationLong,endZipCode,inSurgeAre  
a,riderFirstName,riderLastName,riderPhoneNumber,startAddress,startGooglePlaceId,startL  
ocationLat,startLocationLong,startZipCode}
```

## Description

Request a ride from startLocation to endLocation with given car category

## Parameters

Type	Name	Description	Schema	Default
Query	<b>applePayTok en</b> <i>optional</i>	Apple pay token obtained from Stripe SDK	string	
Query	<b>avatarType</b> <i>optional</i>	Requesting avatar type	enum (RIDER, DRIVER, ADMIN, API_CLIENT, DISPATCHER)	
Query	<b>carCategory</b> <i>optional</i>	Car category	enum (REGULAR, SUV, PREMIUM)	"REGULAR"
Query	<b>cityId</b> <i>required</i>	City ID	integer (int64)	1
Query	<b>comment</b> <i>optional</i>	Ride comment	string	
Query	<b>directConne ctId</b> <i>optional</i>	Driver Direct Connect ID, makes sense only for requests with driverType=DIRECT_CONNECT	string	
Query	<b>driverType</b> <i>optional</i>	Requested driver type	enum (DIRECT_CONNECT, WOMAN_ONLY, FINGERPRINTED)	
Query	<b>endAddress</b> <i>optional</i>	End location address	string	

Type	Name	Description	Schema	Default
Query	<b>endGooglePlaceId</b> <i>optional</i>	End location Google Place ID	string	
Query	<b>endLocationLat</b> <i>required</i>	End location latitude	number (double)	
Query	<b>endLocationLong</b> <i>required</i>	End location longitude	number (double)	
Query	<b>endZipCode</b> <i>optional</i>	End location zipcode	string	
Query	<b>inSurgeArea</b> <i>required</i>	Is request originating in surge area	boolean	"false"
Query	<b>riderFirstName</b> <i>me optional</i>	Overridden rider first name	string	
Query	<b>riderLastName</b> <i>me optional</i>	Overridden rider last name	string	
Query	<b>riderPhoneNumber</b> <i>optional</i>	Overridden rider phone number	string	
Query	<b>startAddress</b> <i>optional</i>	Start location address	string	
Query	<b>startGooglePlaceId</b> <i>optional</i>	Start location Google Place ID	string	
Query	<b>startLocationLat</b> <i>required</i>	Start location latitude	number (double)	

Type	Name	Description	Schema	Default
Query	<b>startLocationLongitude</b> <i>required</i>	Start location longitude	number (double)	
Query	<b>startZipCode</b> <i>optional</i>	Start location zipcode	string	

## Responses

HTTP Code	Description	Schema
<b>200</b>	OK	<a href="#">MobileRiderRideDTO</a>
<b>400</b>	Provided parameters are invalid	No Content
<b>402</b>	Requesting user has an outstanding balance to be paid	No Content
<b>403</b>	Requesting user is not a rider or api client	No Content
<b>500</b>	Server is misconfigured	No Content

## Consumes

- [application/json](#)

## Produces

- [application/json](#)

## Tags

- ride-flow

## Security

Type	Name	Scopes
<b>basic</b>	<a href="#">basic</a>	scope

# Get a list of split fare requests per ride

```
GET /rest/splitfares/{rideId}/list
```

## Parameters

Type	Name	Description	Schema
Path	<b>rideId</b> <i>optional</i>	Ride ID	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	< SplitFareDto > array
403	Requesting rider is not a participant of the ride	No Content
404	Ride not found	No Content

## Produces

- application/json

## Tags

- split-fares

## Security

Type	Name	Scopes
basic	<b>basic</b>	scope

# Send split fare request for a ride

```
POST /rest/splitfares/{rideId}{?phoneNumbers}
```

## Parameters

Type	Name	Description	Schema
Path	<b>rideId</b> <i>optional</i>	Ride ID	integer (int64)
Query	<b>phoneNumbers</b> <i>required</i>	Comma-separated list of recipient phone numbers	< string array(multi) >

## Responses

HTTP Code	Description	Schema
200	OK	<a href="#">SplitFareDto</a>
403	Fare can't be split	No Content
404	Rider not found	No Content

## Consumes

- application/json

## Produces

- \*/\*

## Tags

- split-fares

## Security

Type	Name	Scopes
basic	<a href="#">basic</a>	scope

## Get a list of pending split fare requests per rider

```
GET /rest/splitfares/{riderId}/requested
```

## Parameters

Type	Name	Description	Schema
Path	riderId <i>optional</i>	Rider ID	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	< <a href="#">SplitFareDto</a> > array
403	Requesting rider is not a participant of the ride	No Content
404	Rider not found	No Content

## Produces

- application/json

## Tags

- split-fares

## Security

Type	Name	Scopes
basic	<a href="#">basic</a>	scope

## Remove split fare request and associated rider from the ride

```
DELETE /rest/splitfares/{splitFareId}
```

## Parameters

Type	Name	Description	Schema
Path	splitFareId <i>optional</i>	Split fare request ID	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	No Content
403	Requesting rider is not a participant of the ride	No Content
404	Ride or split fare request not found	No Content

## Produces

- application/json

## Tags

- split-fares

## Security

Type	Name	Scopes
basic	basic	scope

## Accept or decline split fare request

```
POST /rest/splitfares/{splitFareId}/accept{?acceptance}
```

## Parameters

Type	Name	Description	Schema
Path	<b>splitFareId</b> <i>optional</i>	Split fare request ID	integer (int64)
Query	<b>acceptance</b> <i>optional</i>	Accept or decline request	boolean

## Responses

HTTP Code	Description	Schema
200	OK	<a href="#">SplitFareDto</a>
403	Request can't be accepted or declined	No Content
404	Request not found	No Content

## Consumes

- [application/json](#)

## Produces

- [application/json](#)

## Tags

- split-fares

## Security

Type	Name	Scopes
basic	<a href="#">basic</a>	scope

## Send a generic customer support message

```
POST /rest/support/default
```

## Parameters

Type	Name	Description	Schema
Body	<b>supportRequestDto</b> <i>optional</i>	Support request object	<a href="#">SupportRequestDto</a>

## Responses

HTTP Code	Description	Schema
200	OK	No Content
500	Failed to send email message	No Content

## Consumes

- application/json

## Produces

- application/json

## Tags

- support

## Security

Type	Name	Scopes
basic	basic	scope

## Create new support topic

```
POST /rest/supporttopics
```

## Parameters

Type	Name	Description	Schema
Body	<b>supportTopic</b> Dto <i>optional</i>	Support topic object	SupportTopicDto

## Responses

HTTP Code	Description	Schema
200	OK	SupportTopicDto

HTTP Code	Description	Schema
400	Provided data invalid	No Content

## Consumes

- application/json

## Produces

- application/json

## Tags

- support-topics-administration

## Security

Type	Name	Scopes
basic	basic	scope

## Get a list of available support topics for an avatar type

```
GET /rest/supporttopics/list/{avatarType}
```

## Parameters

Type	Name	Description	Schema
Path	avatarType <i>optional</i>	Avatar type	enum (RIDER, DRIVER)

## Responses

HTTP Code	Description	Schema
200	OK	< SupportTopicDto > array

## Produces

- \*/\*

## Tags

- support-topics

## Security

Type	Name	Scopes
basic	basic	scope

## Get a list of child support topics

```
GET /rest/supporttopics/{parentTopicId}/children
```

## Parameters

Type	Name	Description	Schema
Path	<b>parentTopicId</b> <i>optional</i>	Parent topic ID	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	< SupportTopicDto > array

## Produces

- \*/\*

## Tags

- support-topics

## Security

Type	Name	Scopes
basic	basic	scope

# Get support topic object as an administrator

```
GET /rest/supporttopics/{supportTopicId}
```

## Parameters

Type	Name	Description	Schema
Path	<b>supportTopicId</b> <i>optional</i>	Support topic ID	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	<a href="#">SupportTopicDto</a>
404	Support topic not found	No Content

## Produces

- \*/\*

## Tags

- support-topics-administration

## Security

Type	Name	Scopes
basic	<a href="#">basic</a>	scope

# Update support topic

```
PUT /rest/supporttopics/{supportTopicId}
```

## Parameters

Type	Name	Description	Schema
Path	<b>supportTopicId</b> <i>optional</i>	Support topic ID	integer (int64)
Body	<b>supportTopicDto</b> <i>optional</i>	Support topic object	<a href="#">SupportTopicDto</a>

## Responses

HTTP Code	Description	Schema
<b>200</b>	OK	<a href="#">SupportTopicDto</a>
<b>400</b>	Provided data invalid	No Content
<b>404</b>	Support topic not found	No Content

## Consumes

- [application/json](#)

## Produces

- [application/json](#)

## Tags

- support-topics-administration

## Security

Type	Name	Scopes
basic	<a href="#">basic</a>	scope

## Remove a support topic

```
DELETE /rest/supporttopics/{supportTopicId}
```

## Parameters

Type	Name	Description	Schema
Path	<b>supportTopicId</b> <i>optional</i>	Support topic ID	integer (int64)

## Responses

HTTP Code	Description	Schema
<b>204</b>	OK	No Content
<b>404</b>	Support topic not found	No Content

## Produces

- \*/\*

## Tags

- support-topics-administration

## Security

Type	Name	Scopes
basic	basic	scope

## Get a support topic form

```
GET /rest/supporttopics/{supportTopicId}/form
```

## Parameters

Type	Name	Description	Schema
Path	<b>supportTopicId</b> <i>optional</i>	Support topic ID	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	<a href="#">SupportTopicFor mDto</a>
404	Support topic doesn't have a form	No Content

## Produces

- \*/\*

## Tags

- support-topics

## Security

Type	Name	Scopes
basic	<a href="#">basic</a>	scope

## Get a list of available support topics

```
GET /rest/supporttopics{?active,avatarType,description,parentTopicId}
```

## Parameters

Type	Name	Description	Schema
Query	<b>active</b> <i>optional</i>	List active support topics	boolean
Query	<b>avatarType</b> <i>optional</i>	Avatar type	enum (RIDER, DRIVER, ADMIN, API_CLIENT, DISPATCHER)
Query	<b>description</b> <i>optional</i>	Topic description	string

Type	Name	Description	Schema
Query	<b>parentTopicId</b> <i>optional</i>	Parent topic ID	integer (int64)

## Responses

HTTP Code	Description	Schema
<b>200</b>	OK	< <a href="#">SupportTopicDto</a> > array

## Produces

- `*/*`

## Tags

- support-topics

## Security

Type	Name	Scopes
basic	<a href="#">basic</a>	scope

## Send a message to customer support regarding a ride

```
POST /rest/support{?cityId,message,rideId}
```

## Parameters

Type	Name	Description	Schema
Query	<b>cityId</b> <i>optional</i>	City ID	integer (int64)
Query	<b>message</b> <i>required</i>	Support message content	string
Query	<b>rideId</b> <i>optional</i>	Ride ID	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	No Content
404	Ride not found	No Content
500	Failed to send support message	No Content

## Consumes

- application/json

## Produces

- application/json

## Tags

- support

## Security

Type	Name	Scopes
basic	basic	scope

## Create a new surge area

```
POST /rest/surgeareas
```

## Parameters

Type	Name	Description	Schema
Body	surgeArea <i>optional</i>	Surge area object	SurgeAreaDto

## Responses

HTTP Code	Description	Schema
200	OK	<a href="#">SurgeAreaDto</a>
400	Provided data is invalid	No Content

## Consumes

- application/json

## Produces

- application/json

## Tags

- surge-areas-administration

## Security

Type	Name	Scopes
basic	<a href="#">basic</a>	scope

## Bulk update surge areas

```
PUT /rest/surgeareas
```

## Parameters

Type	Name	Description	Schema
Body	<b>surgeAreas</b> <i>optional</i>	List of surge area objects	< <a href="#">SurgeAreaDto</a> > array

## Responses

HTTP Code	Description	Schema
200	OK. List of surge area names that failed to update	< string > array

## Consumes

- application/json

## Produces

- application/json

## Tags

- surge-areas-administration

## Security

Type	Name	Scopes
basic	<a href="#">basic</a>	scope

## Update surge recalculation mode

```
PATCH /rest/surgeareas/config
```

## Parameters

Type	Name	Description	Schema
Body	<b>globalPriority</b> <b>FareDto</b> <i>optional</i>	Global priority config object	<a href="#">GlobalPriorityFareDto</a>

## Responses

HTTP Code	Description	Schema
200	OK	<a href="#">GlobalPriorityFareDto</a>
400	Provided config is invalid	No Content

## Consumes

- application/json

## Produces

- `*/*`

## Tags

- surge-areas-administration

## Security

Type	Name	Scopes
basic	basic	scope

## Update surge recalculation configuration

```
POST /rest/surgeareas/config{?cityId}
```

## Parameters

Type	Name	Description	Schema	Default
Query	<b>cityId</b> <i>required</i>	City ID	integer (int64)	1
Body	<b>config</b> <i>optional</i>	Config object	SurgeRecalculatio nConfig	

## Responses

HTTP Code	Description	Schema
200	OK	SurgeRecalculatio nConfig
400	Provided config is invalid	No Content

## Consumes

- `application/json`

## Produces

- `*/*`

## Tags

- surge-areas-administration

## Security

Type	Name	Scopes
basic	basic	scope

## Get current surge recalculation configuration

```
GET /rest/surgeareas/config{?cityId}
```

## Parameters

Type	Name	Description	Schema	Default
Query	cityId <i>required</i>	City ID	integer (int64)	1

## Responses

HTTP Code	Description	Schema
200	OK	SurgeRecalculatio nConfig

## Produces

- \*/\*

## Tags

- surge-areas-administration

## Security

Type	Name	Scopes
basic	basic	scope

# Update surge area properties

```
PUT /rest/surgeareas/{surgeAreaId}
```

## Parameters

Type	Name	Description	Schema
Path	<b>surgeAreaId</b> <i>optional</i>	Surge area ID	integer (int64)
Body	<b>surgeAreaDto</b> <i>optional</i>	Surge area object	SurgeAreaDto

## Responses

HTTP Code	Description	Schema
200	OK	SurgeAreaDto
400	Provided data is invalid	No Content
404	Surge area not found	No Content

## Consumes

- application/json

## Produces

- application/json

## Tags

- surge-areas-administration

## Security

Type	Name	Scopes
basic	basic	scope

# Remove existing surge area

```
DELETE /rest/surgeareas/{surgeAreaId}
```

## Parameters

Type	Name	Description	Schema
Path	<b>surgeAreaId</b> <i>optional</i>	Surge area ID	integer (int64)

## Responses

HTTP Code	Description	Schema
<b>204</b>	OK	No Content
<b>404</b>	Surge area not found	No Content

## Produces

- \*/\*

## Tags

- surge-areas-administration

## Security

Type	Name	Scopes
basic	basic	scope

# Get a paginated list of active surge areas

```
GET /rest/surgeareas{?cityId,desc,latitude,longitude,name,names,page,pageSize,sort}
```

## Parameters

Type	Name	Description	Schema
Query	<b>cityId</b> <i>optional</i>	City ID	integer (int64)
Query	<b>desc</b> <i>optional</i>	Order the result descending	boolean
Query	<b>latitude</b> <i>optional</i>	Location latitude to search surge area at	number (double)
Query	<b>longitude</b> <i>optional</i>	Location longitude to search surge area at	number (double)
Query	<b>name</b> <i>optional</i>	Surge area name	string
Query	<b>names</b> <i>optional</i>	Set of surge area names	< string > array(multi)
Query	<b>page</b> <i>optional</i>	Page number	integer (int32)
Query	<b>pageSize</b> <i>optional</i>	Page size	integer (int32)
Query	<b>sort</b> <i>optional</i>	Columns to sort the result by, comma-separated list	< string > array(multi)

## Responses

HTTP Code	Description	Schema
200	OK	< <a href="#">RedisSurgeArea</a> > array

## Produces

- `*/*`

## Tags

- surge-areas

## Security

Type	Name	Scopes
basic	basic	scope

## Subscribe to push notifications

POST /rest/tokens

## Parameters

Type	Name	Description	Schema	Default
FormData	avatarType <i>required</i>	Avatar type	enum (RIDER, DRIVER)	"RIDER"
FormData	type <i>optional</i>	Token type	enum (APPLE, GOOGLE, UNDEFINED)	
FormData	value <i>required</i>	Token value	string	

## Responses

HTTP Code	Description	Schema
200	OK	No Content
500	Failed to subscribe	No Content

## Consumes

- application/x-www-form-urlencoded

## Produces

- \*/\*

## Tags

- tokens

## Security

Type	Name	Scopes
basic	basic	scope

## Create a new rider account

POST /rest/users

### Parameters

Type	Name	Description	Schema	Default
FormDa ta	cityId <i>optional</i>	City ID	integer (int64)	1
FormDa ta	data <i>optional</i>	Base64-encoded photo image	string	
FormDa ta	email <i>required</i>	Email	string	
FormDa ta	firstname <i>required</i>	First name	string	
FormDa ta	lastname <i>required</i>	Last name	string	
FormDa ta	password <i>required</i>	Password, either hashed or plain-text	string	
FormDa ta	phonenum ber <i>required</i>	Phone number	string	
FormDa ta	phonenum berVerified <i>optional</i>	Was the phone number verified	boolean	
FormDa ta	socialId <i>optional</i>	Facebook ID	string	

## Responses

HTTP Code	Description	Schema
200	OK	<a href="#">Rider</a>
400	Phone number is of invalid format or is forbidden to use	No Content
422	Provided data is invalid	No Content
500	Failed to get phone number status from Twilio	No Content

## Consumes

- [application/x-www-form-urlencoded](#)

## Produces

- [application/json](#)

## Tags

- users

## Security

Type	Name	Scopes
basic	<a href="#">basic</a>	scope

## Get current user information

```
GET /rest/users/current
```

## Responses

HTTP Code	Description	Schema
200	OK	<a href="#">User</a>

## Produces

- [application/json](#)

## Tags

- users

## Security

Type	Name	Scopes
basic	basic	scope

## Check if the user with requested email and/or phone number already exists

```
POST /rest/users/exists{?email,phoneNumber}
```

## Parameters

Type	Name	Description	Schema
Query	<b>email</b> <i>optional</i>	Email	string
Query	<b>phoneNumber</b> <i>optional</i>	Phone number	string

## Responses

HTTP Code	Description	Schema
200	OK. Email/phone number is not registered in the database	No Content
400	Email/phone number is already registered in the database	No Content
500	Failed to get phone number status from Twilio	No Content

## Consumes

- application/json

## Produces

- \*/\*

## Tags

- users

## Security

Type	Name	Scopes
basic	basic	scope

## Get user information

```
GET /rest/users/{id}
```

## Parameters

Type	Name	Description	Schema
Path	id <i>required</i>	id	integer (int64)

## Responses

HTTP Code	Description	Schema
200	OK	User
404	User not found	No Content

## Produces

- application/json

## Tags

- users

## Security

Type	Name	Scopes
basic	basic	scope

# Update user information

```
PUT /rest/users/{id}
```

## Parameters

Type	Name	Description	Schema
Path	<b>id</b> <i>required</i>	id	integer (int64)
Body	<b>user</b> <i>required</i>	user	<a href="#">UpdateUserDto</a>

## Responses

HTTP Code	Description	Schema
200	OK	<a href="#">User</a>
400	Phone number is of invalid format or is forbidden to use	No Content
403	Updating other user's details is forbidden	No Content
500	Failed to get phone number status from Twilio	No Content

## Consumes

- [application/json](#)

## Produces

- [application/json](#)

## Tags

- users

## Security

Type	Name	Scopes
basic	<b>basic</b>	scope

# Definitions

## ActiveDriverDto

Name	Schema
<b>carCategories</b> <i>required</i>	< string > array
<b>course</b> <i>required</i>	number (double)
<b>driver</b> <i>required</i>	<a href="#">MobileRiderDriverDto</a>
<b>drivingTimeToRider</b> <i>required</i>	integer (int64)
<b>id</b> <i>required</i>	integer (int64)
<b>latitude</b> <i>required</i>	number (double)
<b>longitude</b> <i>required</i>	number (double)
<b>selectedCar</b> <i>required</i>	<a href="#">MobileRiderCarDto</a>

## ActiveDriverInfo

Name	Schema
<b>driver</b> <i>required</i>	<a href="#">DriverInfo</a>
<b>latitude</b> <i>required</i>	number (double)
<b>longitude</b> <i>required</i>	number (double)

Name	Schema
<b>status</b> <i>required</i>	enum (AVAILABLE, REQUESTED, RIDING, INACTIVE, AWAY)

## ActiveDriverLocationDto

Name	Schema
<b>carCategories</b> <i>optional</i>	< string > array
<b>lat</b> <i>optional</i>	number (double)
<b>lng</b> <i>optional</i>	number (double)

## Address

Name	Schema
<b>address</b> <i>required</i>	string
<b>lat</b> <i>required</i>	number (double)
<b>lng</b> <i>required</i>	number (double)
<b>zipCode</b> <i>optional</i>	string

## AddressDto

Name	Schema
<b>address</b> <i>required</i>	string
<b>zipCode</b> <i>required</i>	string

## AppInfo

Name	Description	Schema
<b>avatarType</b> <i>required</i>		enum (RIDER, DRIVER, ADMIN, API_CLIENT, DISPATCHER)
<b>build</b> <i>required</i>		integer (int32)
<b>cityId</b> <i>required</i>	<b>Example :</b> 1	integer (int64)
<b>downloadUrl</b> <i>required</i>		string
<b>id</b> <i>optional</i>		integer (int64)
<b>mandatoryUpGrade</b> <i>required</i>		boolean
<b>platformType</b> <i>required</i>		enum (IOS, ANDROID)
<b>userAgentHeader</b> <i>required</i>		string
<b>version</b> <i>required</i>		string

## AreaDto

Name	Schema
<b>areaPolygon</b> <i>required</i>	< <a href="#">LatLang</a> > array
<b>name</b> <i>required</i>	string

## AreaGeometry

Name	Schema
<b>bottomRightCornerLat</b> <i>optional</i>	number (double)
<b>bottomRightCornerLng</b> <i>optional</i>	number (double)
<b>centerPointLat</b> <i>optional</i>	number (double)
<b>centerPointLng</b> <i>optional</i>	number (double)
<b>csvGeometry</b> <i>optional</i>	string
<b>id</b> <i>optional</i>	integer (int64)
<b>labelLat</b> <i>optional</i>	number (double)
<b>labelLng</b> <i>optional</i>	number (double)
<b>name</b> <i>optional</i>	string
<b>polygon</b> <i>optional</i>	Polygon
<b>topLeftCornerLat</b> <i>optional</i>	number (double)
<b>topLeftCornerLng</b> <i>optional</i>	number (double)

## AreaQueueDriverInfo

Name	Description	Schema
<b>createdOn</b> <i>required</i>		string
<b>driver</b> <i>required</i>	<b>Example :</b> 1	integer (int64)
<b>lastUpdated</b> <i>required</i>		string
<b>name</b> <i>required</i>		string

## AreaQueueInfo

Name	Schema
<b>entries</b> <i>required</i>	< string, < <a href="#">AreaQueueDriverInfo</a> > array > map

## AreaQueuePositions

Name	Schema
<b>areaQueueName</b> <i>required</i>	string
<b>iconUrl</b> <i>required</i>	string
<b>lengths</b> <i>required</i>	< string, integer (int32) > map
<b>mapDisplayConfig</b> <i>required</i>	<a href="#">MapDisplayConfig</a>
<b>positions</b> <i>required</i>	< string, integer (int32) > map

## AuthenticationToken

Name	Description	Schema
<b>token</b> <i>optional</i>	Authentication token to be used as X-Auth-Token header content	string

## Avatar

Name	Schema
<b>active</b> <i>required</i>	boolean
<b>id</b> <i>required</i>	integer (int64)
<b>type</b> <i>required</i>	enum (RIDER, DRIVER, ADMIN, API_CLIENT, DISPATCHER)

## AvatarDto

Name	Schema
<b>active</b> <i>required</i>	boolean
<b>id</b> <i>required</i>	integer (int64)
<b>type</b> <i>required</i>	enum (RIDER, DRIVER, ADMIN, API_CLIENT, DISPATCHER)

## BuildInfo

Name	Schema
<b>buildNumber</b> <i>required</i>	string
<b>gitBranch</b> <i>required</i>	string
<b>gitCommit</b> <i>required</i>	string

Name	Schema
<b>projectVersion</b> <i>required</i>	string

## CampaignAreaDto

Name	Schema
<b>boundary</b> <i>required</i>	< <a href="#">LatLang</a> > array
<b>color</b> <i>required</i>	string
<b>name</b> <i>required</i>	string

## CampaignBannerDto

Name	Description	Schema
<b>bannerIcon</b> <i>optional</i>	Icon to be shown in rider app	string
<b>bannerText</b> <i>optional</i>	Text to be shown in rider app	string
<b>estimatedFare</b> <i>optional</i>	Total estimated fare after campaign discounts have been applied	number
<b>id</b> <i>optional</i>	Unique ID of a ride campaign	integer (int64)
<b>shouldShowDetail</b> <i>optional</i>	Flag to indicate whether campaign details should be shown in rider app	boolean
<b>shouldShowMap</b> <i>optional</i>	Flag to indicate whether campaign area map should be shown in rider app	boolean

## CampaignDto

Name	Schema
<b>areas</b> <i>required</i>	< <a href="#">CampaignAreaDto</a> > array
<b>body</b> <i>required</i>	string
<b>footer</b> <i>required</i>	string
<b>headerIcon</b> <i>required</i>	string
<b>headerTitle</b> <i>required</i>	string

## CampaignProviderDto

Name	Schema
<b>id</b> <i>required</i>	integer (int64)
<b>menuIcon</b> <i>required</i>	string
<b>menuTitle</b> <i>required</i>	string

## CampaignRiderDto

Name	Schema
<b>email</b> <i>required</i>	string
<b>firstName</b> <i>required</i>	string
<b>lastName</b> <i>required</i>	string

## CampaignSubscriptionDto

Name	Schema
<b>id</b> <i>required</i>	integer (int64)
<b>name</b> <i>required</i>	string
<b>subscribed</b> <i>required</i>	boolean

## CancellationReasonDto

Name	Schema
<b>code</b> <i>optional</i>	enum (CHANGE_BOOKING, CHANGE_MIND, ANOTHER_RIDE, MISTAKE, TOO_LONG, RIDER_RESERVED_1, NO_SHOW, WRONG_GPS, TOO_MANY RIDERS, DRIVER_RESERVED_1, OTHER)
<b>description</b> <i>optional</i>	string

## Car

Name	Schema
<b>carCategories</b> <i>optional</i>	< string > array
<b>color</b> <i>optional</i>	string
<b>id</b> <i>optional</i>	integer (int64)
<b>inspectionNotes</b> <i>optional</i>	string
<b>inspectionStatus</b> <i>optional</i>	enum (APPROVED, REJECTED, PENDING, NOT_INSPECTED)

Name	Schema
<b>inspectionSticker</b> <i>optional</i>	boolean
<b>license</b> <i>optional</i>	string
<b>make</b> <i>optional</i>	string
<b>model</b> <i>optional</i>	string
<b>removed</b> <i>optional</i>	boolean
<b>selected</b> <i>optional</i>	boolean
<b>year</b> <i>optional</i>	string

## CarCategoryEstimate

Name	Description	Schema
<b>campaignInfo</b> <i>optional</i>	Information on available ride campaign	<a href="#">CampaignBannerDto</a>
<b>totalFare</b> <i>optional</i>	Total estimated fare	number

## CarDto

Name	Schema
<b>carCategories</b> <i>required</i>	< string > array
<b>color</b> <i>required</i>	string

Name	Schema
<b>id</b> <i>required</i>	integer (int64)
<b>inspectionNotes</b> <i>optional</i>	string
<b>inspectionStatus</b> <i>required</i>	enum (APPROVED, REJECTED, PENDING, NOT_INSPECTED)
<b>insuranceExpiryDate</b> <i>required</i>	string
<b>insurancePictureUrl</b> <i>required</i>	string
<b>license</b> <i>required</i>	string
<b>make</b> <i>required</i>	string
<b>model</b> <i>required</i>	string
<b>photoURL</b> <i>optional</i>	string
<b>photoUrl</b> <i>required</i>	string
<b>removed</b> <i>required</i>	boolean
<b>selected</b> <i>required</i>	boolean
<b>year</b> <i>required</i>	string

## CarModelInfo

Name	Schema
<b>make</b> <i>required</i>	string
<b>model</b> <i>required</i>	string
<b>year</b> <i>required</i>	string

## CarPhotoDto

Name	Schema
<b>carPhotoType</b> <i>required</i>	enum (CAR_PHOTO_FRONT, CAR_PHOTO_BACK, CAR_PHOTO_INSIDE, CAR_PHOTO_TRUNK)
<b>id</b> <i>required</i>	integer (int64)
<b>photoUrl</b> <i>required</i>	string

## CarType

Name	Schema
<b>carCategory</b> <i>optional</i>	string
<b>cityCarTypes</b> <i>optional</i>	< <a href="#">CityCarType</a> > array
<b>configuration</b> <i>optional</i>	string
<b>defaultConfiguration</b> <i>optional</i>	< string, object > map
<b>description</b> <i>optional</i>	string

Name	Schema
<b>fullIconUrl</b> <i>optional</i>	string
<b>iconUrl</b> <i>optional</i>	string
<b>mapIconUrl</b> <i>optional</i>	string
<b>maxPersons</b> <i>optional</i>	integer (int32)
<b>order</b> <i>optional</i>	integer (int32)
<b>plainIconUrl</b> <i>optional</i>	string
<b>selectedFemaleIconUrl</b> <i>optional</i>	string
<b>selectedIconUrl</b> <i>optional</i>	string
<b>title</b> <i>optional</i>	string
<b>unselectedIconUrl</b> <i>optional</i>	string

## ChangeDto

Name	Schema
<b>changedBy</b> <i>required</i>	string
<b>changedFieldName</b> <i>required</i>	string
<b>entityId</b> <i>required</i>	integer (int64)

Name	Schema
<b>entityName</b> <i>required</i>	string
<b>newValue</b> <i>required</i>	string
<b>previousValue</b> <i>required</i>	string
<b>revision</b> <i>required</i>	integer (int64)
<b>revisionDate</b> <i>required</i>	string (date-time)

## Charity

Name	Description	Schema
<b>cityBitmask</b> <i>required</i>		integer (int32)
<b>description</b> <i>optional</i>		string
<b>enabled</b> <i>required</i>		boolean
<b>id</b> <i>required</i>	Example : 1	integer (int64)
<b>imageUrl</b> <i>required</i>		string
<b>name</b> <i>required</i>		string
<b>order</b> <i>required</i>		integer (int32)

## CharityDto

Name	Schema
<b>description</b> <i>required</i>	string
<b>id</b> <i>required</i>	integer (int64)
<b>imageUrl</b> <i>required</i>	string
<b>name</b> <i>required</i>	string

## City

Name	Schema
<b>appName</b> <i>required</i>	string
<b>appStoreLink</b> <i>required</i>	string
<b>areaGeometry</b> <i>required</i>	AreaGeometry
<b>contactEmail</b> <i>required</i>	string
<b>documentsEmail</b> <i>required</i>	string
<b>driversEmail</b> <i>required</i>	string
<b>enabled</b> <i>required</i>	boolean
<b>id</b> <i>optional</i>	integer (int64)

Name	Schema
<b>logoUrl</b> <i>required</i>	string
<b>logoUrlDark</b> <i>required</i>	string
<b>name</b> <i>required</i>	string
<b>office</b> <i>required</i>	string
<b>onboardingEmail</b> <i>required</i>	string
<b>pageUrl</b> <i>required</i>	string
<b>playStoreLink</b> <i>required</i>	string
<b>supportEmail</b> <i>required</i>	string
<b>unsubscribeLink</b> <i>optional</i>	string
<b>updatePreferencesLink</b> <i>optional</i>	string

## CityCarType

Name	Schema
<b>baseFare</b> <i>optional</i>	number
<b>bookingFee</b> <i>optional</i>	number
<b>cancellationFee</b> <i>optional</i>	number

Name	Schema
<b>carType</b> <i>optional</i>	<a href="#">CarType</a>
<b>cityFeeRate</b> <i>optional</i>	number
<b>cityId</b> <i>optional</i>	integer (int64)
<b>configuration</b> <i>optional</i>	string
<b>defaultConfiguration</b> <i>optional</i>	< string, object > map
<b>enabled</b> <i>optional</i>	boolean
<b>fixedRAFee</b> <i>optional</i>	number
<b>id</b> <i>optional</i>	integer (int64)
<b>minimumFare</b> <i>optional</i>	number
<b>processingFeeFixedPart</b> <i>optional</i>	number
<b>processingFeeMinimum</b> <i>optional</i>	number
<b>processingFeeRate</b> <i>optional</i>	number
<b>processingFeeText</b> <i>optional</i>	string
<b>ratePerMile</b> <i>optional</i>	number

Name	Schema
<b>ratePerMinute</b> <i>optional</i>	number

## CityCarTypeDto

Name	Description	Schema
<b>active</b> <i>required</i>		boolean
<b>baseFare</b> <i>required</i>		number
<b>bookingFee</b> <i>required</i>		number
<b>cancellationFee</b> <i>required</i>		number
<b>carCategory</b> <i>required</i>		string
<b>cityId</b> <i>required</i>	Example : 1	integer (int64)
<b>configuration</b> <i>required</i>		string
<b>description</b> <i>required</i>		string
<b>fullIconUrl</b> <i>required</i>		string
<b>iconUrl</b> <i>required</i>		string
<b>mapIconUrl</b> <i>required</i>		string

Name	Description	Schema
<b>maxPersons required</b>	<b>Example : 4</b>	integer (int32)
<b>minimumFare required</b>		number
<b>order required</b>	<b>Example : 1</b>	integer (int32)
<b>plainIconUrl required</b>		string
<b>processingFee required</b>		string
<b>processingFee Rate required</b>		number
<b>processingFee Text required</b>		string
<b>raFixedFee required</b>		number
<b>ratePerMile required</b>		number
<b>ratePerMinute required</b>		number
<b>selectedFemaleIconUrl required</b>		string
<b>selectedIconUrl required</b>		string
<b>title required</b>		string

Name	Description	Schema
<b>tncFeeRate</b> <i>required</i>		number
<b>unselectedIco</b> <b>nUrl</b> <i>required</i>		string

## CityDriverTypeDto

Name	Schema
<b>availableInCategories</b> <i>required</i>	< string > array
<b>cityId</b> <i>required</i>	integer (int64)
<b>description</b> <i>required</i>	string
<b>eligibleCategories</b> <i>required</i>	< string > array
<b>name</b> <i>optional</i>	string

## CompactActiveDriverDto

Name	Schema
<b>course</b> <i>optional</i>	number (double)
<b>driver</b> <i>required</i>	<a href="#">DriverDto</a>
<b>drivingTimeToRider</b> <i>optional</i>	integer (int64)
<b>id</b> <i>optional</i>	integer (int64)

Name	Schema
<b>latitude</b> <i>required</i>	number (double)
<b>longitude</b> <i>required</i>	number (double)
<b>ride</b> <i>optional</i>	<a href="#">MobileDriverRideDto</a>
<b>speed</b> <i>optional</i>	number (double)
<b>status</b> <i>required</i>	enum (AVAILABLE, REQUESTED, RIDING, INACTIVE, AWAY)

## CompactRideDto

Name	Schema
<b>cancelledOn</b> <i>optional</i>	string (date-time)
<b>carCategory</b> <i>required</i>	string
<b>completedOn</b> <i>optional</i>	string (date-time)
<b>distanceTravelled</b> <i>optional</i>	number
<b>driverFullscreen</b> <i>required</i>	string
<b>driverId</b> <i>required</i>	integer (int64)
<b>id</b> <i>required</i>	integer (int64)
<b>riderFullscreen</b> <i>required</i>	string

Name	Schema
<b>riderId</b> <i>required</i>	integer (int64)
<b>startedOn</b> <i>optional</i>	string (date-time)
<b>tip</b> <i>optional</i>	number
<b>tippedOn</b> <i>optional</i>	string (date-time)

## ConfigurationItemDto

Name	Description	Schema
<b>cityId</b> <i>required</i>	<b>Example : 1</b>	integer (int64)
<b>clientType</b> <i>required</i>		enum (RIDER, DRIVER, CONSOLE, UNKNOWN)
<b>configuration</b> <b>Key</b> <i>required</i>		string
<b>configuration</b> <b>Value</b> <i>required</i>		object
<b>default</b> <i>optional</i>		boolean
<b>environment</b> <i>optional</i>		string
<b>id</b> <i>required</i>	<b>Example : 1</b>	integer (int64)

## ConsoleActiveDriverDto

Name	Schema
<b>driver</b> <i>required</i>	<a href="#">ConsoleAvatarDto</a>

## ConsoleAvatarDto

Name	Schema
<b>appVersion</b> <i>required</i>	string
<b>fullName</b> <i>required</i>	string
<b>id</b> <i>required</i>	integer (int64)
<b>photoUrl</b> <i>required</i>	string

## ConsoleDriverDto

Name	Schema
<b>activationNotes</b> <i>required</i>	string
<b>activationStatus</b> <i>required</i>	enum (ACTIVE, REJECTED, SUSPENDED, DEACTIVATED_OTHER, INACTIVE)
<b>cars</b> <i>required</i>	< <a href="#">CarDto</a> > array
<b>chauffeurLicense</b> <i>optional</i>	boolean
<b>cityApprovalStatus</b> <i>required</i>	enum (PENDING, NOT_PROVIDED, APPROVED, REJECTED_PHOTO, REJECTED_BY_CITY, EXPIRED)
<b>directConnectId</b> <i>required</i>	string

Name	Schema
<b>fullName</b> <i>required</i>	string
<b>grantedDriverTypes</b> <i>required</i>	< string > array
<b>id</b> <i>required</i>	integer (int64)
<b>licenseNumber</b> <i>required</i>	string
<b>licenseState</b> <i>required</i>	string
<b>onlineStatus</b> <i>required</i>	enum (ONLINE, OFFLINE, RIDING, STUCK)
<b>payoneerStatus</b> <i>required</i>	enum (Not registered, Inactive, Active)
<b>rating</b> <i>required</i>	number (double)
<b>specialFlags</b> <i>required</i>	< enum (DEAF) > array
<b>ssn</b> <i>required</i>	string
<b>user</b> <i>required</i>	UserDto

## ConsoleRideDto

Name	Schema
<b>activeDriver</b> <i>optional</i>	ConsoleActiveDriverDto
<b>airportFee</b> <i>optional</i>	number

Name	Schema
<b>baseFare</b> <i>optional</i>	number
<b>bookingFee</b> <i>optional</i>	number
<b>campaign</b> <i>optional</i>	string
<b>campaignCoverage</b> <i>optional</i>	number
<b>cancellationFee</b> <i>optional</i>	number
<b>cancelledOn</b> <i>optional</i>	string (date-time)
<b>cityFee</b> <i>optional</i>	number
<b>completedOn</b> <i>optional</i>	string (date-time)
<b>distanceFare</b> <i>optional</i>	number
<b>distanceTravelled</b> <i>optional</i>	number
<b>driverAcceptedOn</b> <i>optional</i>	string (date-time)
<b>driverPayment</b> <i>optional</i>	number
<b>driverReachedOn</b> <i>optional</i>	string (date-time)
<b>endAddress</b> <i>optional</i>	string

Name	Schema
<b>endLocationLat</b> <i>optional</i>	number (double)
<b>endLocationLong</b> <i>optional</i>	number (double)
<b>estimatedFare</b> <i>optional</i>	number
<b>fareTotal</b> <i>optional</i>	number
<b>freeCreditCharged</b> <i>optional</i>	number
<b>id</b> <i>required</i>	integer (int64)
<b>minimumFare</b> <i>optional</i>	number
<b>normalFare</b> <i>optional</i>	number
<b>processingFee</b> <i>optional</i>	number
<b>raPayment</b> <i>optional</i>	number
<b>ratePerMile</b> <i>optional</i>	number
<b>ratePerMinute</b> <i>optional</i>	number
<b>requestedCarType</b> <i>required</i>	<a href="#">RequestedCarType</a>
<b>requestedOn</b> <i>required</i>	string (date-time)

Name	Schema
<b>rideCost</b> <i>optional</i>	number
<b>rider</b> <i>required</i>	<a href="#">ConsoleRiderDto</a>
<b>riderCard</b> <i>required</i>	<a href="#">ConsoleRiderCard</a>
<b>roundUpAmount</b> <i>optional</i>	number
<b>startAddress</b> <i>required</i>	string
<b>startLocationLat</b> <i>required</i>	number (double)
<b>startLocationLong</b> <i>required</i>	number (double)
<b>startedOn</b> <i>optional</i>	string (date-time)
<b>status</b> <i>required</i>	enum (REQUEST_QUEUED, REQUEST_QUEUE_EXPIRED, REQUESTED, RIDER_CANCELLED, DRIVER_ASSIGNED, DRIVER_CANCELLED, DRIVER_REACHED, ACTIVE, NO_AVAILABLE_DRIVER, COMPLETED, ADMIN_CANCELLED)
<b>stripeCreditCharge</b> <i>optional</i>	number
<b>subTotal</b> <i>optional</i>	number
<b>surgeFactor</b> <i>required</i>	number
<b>surgeFare</b> <i>optional</i>	number

Name	Schema
<b>timeFare</b> <i>optional</i>	number
<b>tip</b> <i>optional</i>	number
<b>tippedOn</b> <i>optional</i>	string (date-time)
<b>totalCharge</b> <i>optional</i>	number
<b>totalFare</b> <i>optional</i>	number

## ConsoleRiderCard

Name	Schema
<b>cardBrand</b> <i>required</i>	enum (VISA, MASTERCARD, AMERICAN_EXPRESS, DISCOVER, JCB, DINERS_CLUB, UNKNOWN)
<b>cardNumber</b> <i>required</i>	string

## ConsoleRiderDto

Name	Schema
<b>fullName</b> <i>required</i>	string
<b>id</b> <i>required</i>	integer (int64)
<b>user</b> <i>required</i>	<a href="#">ConsoleAvatarDto</a>

## ContactSuccessResponse

Name	Schema
<b>message</b> <i>required</i>	string
<b>status</b> <i>required</i>	string

## CreateRequest

Name	Description	Schema
<b>cityId</b> <i>required</i>	<b>Example :</b> 1	integer (int64)
<b>clientType</b> <i>required</i>		string
<b>configuration</b> <b>Key</b> <i>required</i>		string
<b>configuration</b> <b>Value</b> <i>required</i>		object
<b>default</b> <i>optional</i>		boolean
<b>isDefault</b> <i>required</i>		boolean

## CumulativeRidesReportEntry

Name	Schema
<b>driversRidesReport</b> <i>required</i>	Page«DriverRidesReportEntry»
<b>ridesReport</b> <i>required</i>	< RideReportEntry > array

## CurrentActiveDriverDto

Name	Schema
<b>carCategories</b> <i>required</i>	< string > array
<b>course</b> <i>optional</i>	number (double)
<b>driver</b> <i>required</i>	<a href="#">DriverDto</a>
<b>drivingTimeToRider</b> <i>optional</i>	integer (int64)
<b>latitude</b> <i>required</i>	number (double)
<b>longitude</b> <i>required</i>	number (double)
<b>ride</b> <i>optional</i>	<a href="#">MobileDriverRideDto</a>
<b>speed</b> <i>optional</i>	number (double)
<b>status</b> <i>required</i>	enum (AVAILABLE, REQUESTED, RIDING, INACTIVE, AWAY)

## CurrentRiderDto

Name	Schema
<b>cards</b> <i>required</i>	< <a href="#">RiderCardDto</a> > array
<b>ride</b> <i>required</i>	<a href="#">MobileRiderRideDto</a>
<b>rider</b> <i>required</i>	<a href="#">RiderDto</a>

Name	Schema
<b>unpaid</b> <i>required</i>	< PendingPaymentDto > array

## CustomPaymentDto

Name	Description	Schema
<b>category</b> <i>required</i>		enum (INCENTIVE, BONUS, COMPENSATION, REIMBURSEMENT, OTHER)
<b>creatorEmail</b> <i>required</i>		string
<b>creatorFirstName</b> <i>required</i>		string
<b>creatorId</b> <i>required</i>	<b>Example : 1</b>	integer (int64)
<b>creatorLastName</b> <i>required</i>		string
<b>description</b> <i>optional</i>		string
<b>driverEmail</b> <i>required</i>		string
<b>driverFirstName</b> <i>required</i>		string
<b>driverId</b> <i>required</i>	<b>Example : 1</b>	integer (int64)
<b>driverLastName</b> <i>required</i>		string

Name	Description	Schema
<b>id</b> <i>required</i>	<b>Example : 1</b>	integer (int64)
<b>paymentDate</b> <i>required</i>		string
<b>value</b> <i>required</i>		number

## DeferredResult«List«EventDto»»

Name	Schema
<b>result</b> <i>optional</i>	object
<b>setOrExpired</b> <i>optional</i>	boolean

## DeferredResult«MobileDriverRideDto»

Name	Schema
<b>result</b> <i>optional</i>	object
<b>setOrExpired</b> <i>optional</i>	boolean

## DeferredResult«ResponseEntity«object»»

Name	Schema
<b>result</b> <i>optional</i>	object
<b>setOrExpired</b> <i>optional</i>	boolean

## DeferredResult«ResponseEntity»

Name	Schema
<b>result</b> <i>optional</i>	object
<b>setOrExpired</b> <i>optional</i>	boolean

## DirectConnectDriverDto

Name	Description	Schema
<b>categories</b> <i>optional</i>	Available car categories	< string > array
<b>factors</b> <i>optional</i>	Surge factors per car category	< string, number > map
<b>firstName</b> <i>required</i>		string
<b>lastName</b> <i>required</i>		string
<b>photoUrl</b> <i>optional</i>		string
<b>rating</b> <i>required</i>		number (double)

## DirectConnectDto

Name	Schema
<b>directConnectId</b> <i>required</i>	string

## DirectConnectHistoryDto

Name	Schema
<b>directConnectId</b> <i>required</i>	string
<b>driverFirstName</b> <i>required</i>	string
<b>driverLastName</b> <i>required</i>	string
<b>photoURL</b> <i>required</i>	string
<b>requestedAt</b> <i>required</i>	string

## DispatcherAccountRideDto

Name	Schema
<b>activeDriver</b> <i>optional</i>	<a href="#">ActiveDriverDto</a>
<b>comment</b> <i>optional</i>	string
<b>completedOn</b> <i>optional</i>	string (date-time)
<b>driverAcceptedOn</b> <i>optional</i>	string (date-time)
<b>driverPayment</b> <i>optional</i>	number
<b>driverRating</b> <i>optional</i>	number (double)
<b>end</b> <i>optional</i>	<a href="#">Address</a>
<b>endAddress</b> <i>optional</i>	string

Name	Schema
<b>endLocationLat</b> <i>optional</i>	number (double)
<b>endLocationLong</b> <i>optional</i>	number (double)
<b>estimatedTimeArrive</b> <i>optional</i>	integer (int64)
<b>estimatedTimeCompletion</b> <i>optional</i>	string (date-time)
<b>freeCancellationExpiresOn</b> <i>optional</i>	string (date-time)
<b>freeCreditCharged</b> <i>optional</i>	number
<b>id</b> <i>required</i>	integer (int64)
<b>mapUrl</b> <i>optional</i>	string
<b>passenger</b> <i>required</i>	<a href="#">DispatcherRiderDto</a>
<b>paymentProvider</b> <i>optional</i>	enum (CREDIT_CARD, APPLE_PAY)
<b>precedingRide</b> <i>optional</i>	<a href="#">PrecedingRide</a>
<b>requestedCarType</b> <i>required</i>	<a href="#">CityCarTypeDto</a>
<b>start</b> <i>optional</i>	<a href="#">Address</a>
<b>startAddress</b> <i>optional</i>	string

Name	Schema
<b>startLocationLat</b> <i>required</i>	number (double)
<b>startLocationLong</b> <i>required</i>	number (double)
<b>status</b> <i>required</i>	enum (REQUEST_QUEUED, REQUEST_QUEUE_EXPIRED, REQUESTED, RIDER_CANCELLED, DRIVER_ASSIGNED, DRIVER_CANCELLED, DRIVER_REACHED, ACTIVE, NO_AVAILABLE_DRIVER, COMPLETED, ADMIN_CANCELLED)
<b>tip</b> <i>optional</i>	number
<b>tipUntil</b> <i>optional</i>	string (date-time)
<b>tippingAllowed</b> <i>required</i>	boolean
<b>totalCharge</b> <i>optional</i>	number
<b>totalFare</b> <i>optional</i>	number
<b>upfrontCharge</b> <i>optional</i>	number
<b>upgradeRequest</b> <i>optional</i>	<a href="#">RideUpgradeRequestDto</a>

## DispatcherRiderDto

Name	Schema
<b>firstName</b> <i>required</i>	string
<b>lastName</b> <i>required</i>	string

Name	Schema
<b>phoneNumber</b> <i>required</i>	string

## Document

Name	Schema
<b>cityId</b> <i>optional</i>	integer (int64)
<b>documentStatus</b> <i>optional</i>	enum (APPROVED, PENDING, EXPIRED, REJECTED, NOT_PROVIDED)
<b>documentType</b> <i>optional</i>	enum (LICENSE, INSURANCE, PHOTO, FRONT, BACK, INSIDE, TRUNK, TNC_CARD, CHAUFFEUR_LICENSE, CAR_STICKER)
<b>documentUrl</b> <i>optional</i>	string
<b>id</b> <i>optional</i>	integer (int64)
<b>name</b> <i>optional</i>	string
<b>notes</b> <i>optional</i>	string
<b>removed</b> <i>optional</i>	boolean
<b>validityDate</b> <i>optional</i>	string (date-time)

## DocumentDto

Name	Description	Schema
<b>cityId</b> <i>optional</i>		integer (int64)

Name	Description	Schema
<b>documentStatus</b> <i>optional</i>		enum (APPROVED, PENDING, EXPIRED, REJECTED, NOT_PROVIDED)
<b>documentType</b> <i>optional</i>		enum (LICENSE, INSURANCE, PHOTO, FRONT, BACK, INSIDE, TRUNK, TNC_CARD, CHAUFFEUR_LICENSE, CAR_STICKER)
<b>documentUrl</b> <i>optional</i>		string
<b>id</b> <i>optional</i>		integer (int64)
<b>name</b> <i>optional</i>		string
<b>notes</b> <i>optional</i>		string
<b>validityDate</b> <i>optional</i>	Example : "yyyy-MM-dd"	string

## Driver

Name	Schema
<b>active</b> <i>optional</i>	boolean
<b>agreementDate</b> <i>optional</i>	string
<b>cityId</b> <i>optional</i>	integer (int64)
<b>id</b> <i>optional</i>	integer (int64)

Name	Schema
<b>insuranceExpiryDate</b> <i>required</i>	string (date-time)
<b>lastLoginDate</b> <i>optional</i>	string (date-time)
<b>licenseExpiryDate</b> <i>required</i>	string (date-time)
<b>licenseNumber</b> <i>required</i>	string
<b>licenseState</b> <i>required</i>	string
<b>rating</b> <i>required</i>	number (double)
<b>ssn</b> <i>required</i>	string
<b>user</b> <i>optional</i>	User

## DriverDto

Name	Schema
<b>fullName</b> <i>optional</i>	string
<b>id</b> <i>optional</i>	integer (int64)
<b>phoneNumber</b> <i>optional</i>	string

## DriverEmailReminderDto

Name	Schema
<b>cityId</b> <i>required</i>	integer (int64)
<b>extraFields</b> <i>required</i>	< <a href="#">ExtraField</a> > array
<b>id</b> <i>required</i>	integer (int64)
<b>name</b> <i>required</i>	string

## DriverInfo

Name	Schema
<b>fullName</b> <i>required</i>	string
<b>id</b> <i>required</i>	integer (int64)
<b>phoneNumber</b> <i>required</i>	string

## DriverOnline

Name	Schema
<b>seconds</b> <i>required</i>	integer (int64)

## DriverRide

Name	Schema
<b>baseFare</b> <i>required</i>	number
<b>bookingFee</b> <i>required</i>	number

Name	Schema
<b>cancelledOn</b> <i>required</i>	string (date-time)
<b>car</b> <i>required</i>	<a href="#">RideCar</a>
<b>cityFee</b> <i>required</i>	number
<b>completedOn</b> <i>required</i>	string (date-time)
<b>distanceFare</b> <i>required</i>	number
<b>driverPayment</b> <i>required</i>	number
<b>driverRating</b> <i>required</i>	number (double)
<b>end</b> <i>required</i>	<a href="#">Address</a>
<b>endLocationLat</b> <i>required</i>	number (double)
<b>endLocationLng</b> <i>required</i>	number (double)
<b>id</b> <i>required</i>	integer (int64)
<b>minimumFare</b> <i>required</i>	number
<b>raFee</b> <i>required</i>	number
<b>ratePerMile</b> <i>required</i>	number

Name	Schema
<b>ratePerMinute</b> <i>required</i>	number
<b>requestedCarType</b> <i>required</i>	CarType
<b>rideMap</b> <i>required</i>	string
<b>roundUpAmount</b> <i>required</i>	number
<b>start</b> <i>required</i>	Address
<b>startLocationLat</b> <i>required</i>	number (double)
<b>startLocationLng</b> <i>required</i>	number (double)
<b>startedOn</b> <i>required</i>	string (date-time)
<b>status</b> <i>required</i>	enum (REQUEST_QUEUED, REQUEST_QUEUE_EXPIRED, REQUESTED, RIDER_CANCELLED, DRIVER_ASSIGNED, DRIVER_CANCELLED, DRIVER_REACHED, ACTIVE, NO_AVAILABLE_DRIVER, COMPLETED, ADMIN_CANCELLED)
<b>subTotal</b> <i>required</i>	number
<b>surgeFactor</b> <i>required</i>	number
<b>surgeFare</b> <i>required</i>	number
<b>timeFare</b> <i>required</i>	number

Name	Schema
<b>tip</b> <i>required</i>	number
<b>totalFare</b> <i>required</i>	number

## DriverRidesReportEntry

Name	Schema
<b>cancellationFee</b> <i>required</i>	number
<b>completedRides</b> <i>required</i>	integer (int64)
<b>distanceTraveledInMiles</b> <i>required</i>	number
<b>distanceTravelled</b> <i>required</i>	number
<b>driverBasePayment</b> <i>required</i>	number
<b>driverId</b> <i>required</i>	integer (int64)
<b>driverPayment</b> <i>required</i>	number
<b>firstName</b> <i>required</i>	string
<b>lastName</b> <i>required</i>	string
<b>priorityFare</b> <i>required</i>	number
<b>priorityFareRides</b> <i>required</i>	integer (int64)

Name	Schema
<b>raGrossMargin</b> <i>required</i>	number
<b>tips</b> <i>required</i>	number
<b>totalFare</b> <i>required</i>	number
<b>userId</b> <i>required</i>	integer (int64)

## DriverStatisticDto

Name	Schema
<b>description</b> <i>required</i>	string
<b>name</b> <i>required</i>	string
<b>outOfTotal</b> <i>required</i>	integer (int32)
<b>value</b> <i>required</i>	integer (int32)

## DriverStatusDto

Name	Schema
<b>active</b> <i>optional</i>	integer (int64)
<b>finalReview</b> <i>optional</i>	integer (int64)
<b>pending</b> <i>optional</i>	integer (int64)

Name	Schema
<b>rejected</b> <i>optional</i>	integer (int64)
<b>suspended</b> <i>optional</i>	integer (int64)

## DriverStatusPendingDto

Name	Description	Schema
<b>carInspection</b> <i>optional</i>	Count of drivers with car inspection status PENDING, NOT_INSPECTED	integer (int64)
<b>carPhotos</b> <i>optional</i>	Count of drivers with car photos status PENDING	integer (int64)
<b>cityApproval</b> <i>optional</i>	Count of drivers with city approval status PENDING, NOT_PROVIDED	integer (int64)
<b>driverLicense</b> <i>optional</i>	Count of drivers with driver license status PENDING	integer (int64)
<b>inspectionSticker</b> <i>optional</i>	Count of drivers with inspection sticker status PENDING	integer (int64)
<b>insurance</b> <i>optional</i>	Count of drivers with insurance status PENDING	integer (int64)
<b>payoneer</b> <i>optional</i>	Count of drivers with payoneer status PENDING, INACTIVE	integer (int64)
<b>profilePhotos</b> <i>optional</i>	Count of drivers with profile photos status PENDING	integer (int64)

## EndAddress

Name	Schema
<b>address</b> <i>optional</i>	string

Name	Schema
<b>latitude</b> <i>required</i>	number (double)
<b>longitude</b> <i>required</i>	number (double)
<b>zipCode</b> <i>optional</i>	string

## EstimatedFareDTO

Name	Description	Schema
<b>campaignInfo</b> <i>optional</i>	Information on available ride campaign	<a href="#">CampaignBannerDto</a>
<b>duration</b> <i>optional</i>		integer (int64)
<b>estimates</b> <i>optional</i>	Map of CarCategoryEstimate per car category	< string, <a href="#">CarCategoryEstimate</a> > map
<b>totalFare</b> <i>optional</i>	Total estimated fare	number

## EventDto

Name	Schema
<b>eventType</b> <i>required</i>	enum (REQUESTED, HANDSHAKE, RIDER_CANCELLED, DRIVER_ASSIGNED, DRIVER_CANCELLED, DRIVER_REACHED, ACTIVE, NO_AVAILABLE_DRIVER, COMPLETED, ADMIN_CANCELLED, END_LOCATION_UPDATED, GO_OFFLINE, CUSTOM_MESSAGE, QUEUED_AREA_ENTERING, QUEUED_AREA_LEAVEING, QUEUED_AREA_LEAVEING_INACTIVE, QUEUED_AREA_LEAVEING_PENALTY, QUEUED_AREA_LEAVEING_RIDE, QUEUED_AREA_UPDATE, SURGE_AREA_UPDATES, RATING_UPDATED, CAR_CATEGORY_CHANGE, DRIVER_TYPE_UPDATE, CONFIG_CREATED, CONFIG_CHANGED, RIDER_LOCATION_UPDATED, RIDER_COMMENT_UPDATED, RIDE_UPGRADE_ACCEPTED, RIDE_UPGRADE_DECLINED, RIDE_STACKED_REASSIGNED)
<b>id</b> <i>required</i>	integer (int64)
<b>message</b> <i>optional</i>	string
<b>nextRide</b> <i>optional</i>	<a href="#">MobileDriverRideDto</a>
<b>parameters</b> <i>optional</i>	string
<b>ride</b> <i>optional</i>	<a href="#">MobileDriverRideDto</a>

## ExtendedRideDto

Name	Schema
<b>activeDriverId</b> <i>optional</i>	integer (int64)
<b>carType</b> <i>required</i>	string

Name	Schema
<b>completed</b> <i>optional</i>	string (date-time)
<b>distance</b> <i>optional</i>	number
<b>distanceInMiles</b> <i>optional</i>	number
<b>driverAppVersion</b> <i>required</i>	string
<b>driverFirstName</b> <i>required</i>	string
<b>driverId</b> <i>required</i>	integer (int64)
<b>driverLastName</b> <i>required</i>	string
<b>driverLatitude</b> <i>optional</i>	number (double)
<b>driverLongitude</b> <i>optional</i>	number (double)
<b>driverPhoneNumber</b> <i>required</i>	string
<b>endAddress</b> <i>optional</i>	<a href="#">AddressDto</a>
<b>rideId</b> <i>required</i>	integer (int64)
<b>riderAppVersion</b> <i>required</i>	string
<b>riderFirstName</b> <i>required</i>	string

Name	Schema
<b>riderId</b> <i>required</i>	integer (int64)
<b>riderLastName</b> <i>required</i>	string
<b>startAddress</b> <i>required</i>	<a href="#">AddressDto</a>
<b>started</b> <i>optional</i>	string (date-time)
<b>status</b> <i>required</i>	enum (REQUEST_QUEUED, REQUEST_QUEUE_EXPIRED, REQUESTED, RIDER_CANCELLED, DRIVER_ASSIGNED, DRIVER_CANCELLED, DRIVER_REACHED, ACTIVE, NO_AVAILABLE_DRIVER, COMPLETED, ADMIN_CANCELLED)

## ExtraField

Name	Schema
<b>id</b> <i>required</i>	string
<b>name</b> <i>required</i>	string
<b>order</b> <i>required</i>	integer (int32)
<b>type</b> <i>required</i>	enum (TEXT, TEXTAREA)

## FarePaymentDto

Name	Schema
<b>chargeId</b> <i>optional</i>	string

Name	Schema
<b>createdDate</b> <i>required</i>	string
<b>freeCreditCharged</b> <i>required</i>	number
<b>id</b> <i>required</i>	integer (int64)
<b>mainRider</b> <i>required</i>	boolean
<b>paymentProvider</b> <i>required</i>	enum (CREDIT_CARD, APPLE_PAY)
<b>paymentStatus</b> <i>required</i>	enum (PAID, UNPAID, BLOCKED, PREPAID_UPFRONT)
<b>rideId</b> <i>required</i>	integer (int64)
<b>riderFullName</b> <i>required</i>	string
<b>riderId</b> <i>required</i>	integer (int64)
<b>riderPhoto</b> <i>optional</i>	string
<b>status</b> <i>required</i>	enum (REQUESTED, ACCEPTED, DECLINED)
<b>stripeCreditCharge</b> <i>required</i>	number
<b>updatedDate</b> <i>required</i>	string
<b>usedCard</b> <i>required</i>	<a href="#">RiderCardDto</a>

# Field

Name	Schema
<b>fieldPlaceholder</b> <i>optional</i>	string
<b>fieldTitle</b> <i>required</i>	string
<b>fieldType</b> <i>required</i>	string
<b>isMandatory</b> <i>required</i>	boolean
<b>mandatory</b> <i>optional</i>	boolean
<b>variable</b> <i>required</i>	string

# FoundItemDto

Name	Description	Schema
<b>details</b> <i>required</i>	Description of the found item	string
<b>foundOn</b> <i>required</i>	Date the item is found on	string (date-time)
<b>rideDescription</b> <i>required</i>	Ride description	string
<b>rideId</b> <i>optional</i>	Ride ID	integer (int64)
<b>sharingContactsAllowed</b> <i>required</i>	Does driver allow to share phone number with the rider	boolean

## GeolocationLogDto

Name	Schema
<b>createdDate</b> <i>required</i>	string (date-time)
<b>event</b> <i>required</i>	enum (RIDER_APP_OPEN, GET_ACTIVE_DRIVERS_BY RIDER)
<b>id</b> <i>required</i>	integer (int64)
<b>locationLat</b> <i>required</i>	number (double)
<b>locationLng</b> <i>required</i>	number (double)
<b>riderEmail</b> <i>required</i>	string
<b>riderFirstName</b> <i>required</i>	string
<b>riderId</b> <i>optional</i>	integer (int64)
<b>riderLastName</b> <i>required</i>	string
<b>updatedDate</b> <i>required</i>	string (date-time)

## GlobalPriorityFareDto

Name	Description	Schema
<b>cityId</b> <i>required</i>	City ID <b>Example :</b> 1	integer (int64)
<b>surgeMode</b> <i>required</i>	Surge mode	enum (FULL_AUTO, LIMITED_AUTO, MANUAL)

# Info

Name	Schema
<b>active</b> <i>optional</i>	boolean
<b>id</b> <i>optional</i>	integer (int64)
<b>type</b> <i>optional</i>	enum (RIDER, DRIVER, ADMIN, API_CLIENT, DISPATCHER)

## Iterable«AppInfo»

Type : object

## Iterable«ExtendedRideDto»

Type : object

## Iterable«ZipCodeReportEntry»

Type : object

## LatLng

Name	Description	Schema
<b>lat</b> <i>required</i>	<b>Example :</b> 30.286804	number (double)
<b>lng</b> <i>required</i>	<b>Example :</b> -97.707425	number (double)

## Line

Name	Schema
<b>a</b> <i>optional</i>	number (double)

Name	Schema
<b>b</b> <i>optional</i>	number (double)
<b>end</b> <i>optional</i>	Point
<b>start</b> <i>optional</i>	Point
<b>vertical</b> <i>optional</i>	boolean

## ListPromocodeDto

Name	Schema
<b>applicableToFees</b> <i>required</i>	boolean
<b>cappedAmountPerUse</b> <i>required</i>	number
<b>carTypes</b> <i>required</i>	< string > array
<b>cities</b> <i>required</i>	< integer (int64) > array
<b>codeLiteral</b> <i>required</i>	string
<b>codeValue</b> <i>required</i>	number
<b>currentRedemption</b> <i>required</i>	integer (int64)
<b>endsOn</b> <i>required</i>	string (date-time)
<b>id</b> <i>required</i>	integer (int64)

Name	Schema
<b>maxPromotionValue</b> <i>optional</i>	number
<b>maximumRedemption</b> <i>required</i>	integer (int64)
<b>maximumUsesPerAccount</b> <i>required</i>	integer (int32)
<b>newRidersOnly</b> <i>required</i>	boolean
<b>nextTripOnly</b> <i>required</i>	boolean
<b>startsOn</b> <i>required</i>	string (date-time)
<b>title</b> <i>required</i>	string
<b>usageCount</b> <i>required</i>	integer (int64)
<b>useEndDate</b> <i>required</i>	string (date-time)
<b>validForNumberOfDays</b> <i>required</i>	integer (int32)
<b>validForNumberOfRides</b> <i>required</i>	integer (int32)

## Location

Name	Description	Schema
<b>lat</b> <i>required</i>	Current GPS latitude <b>Example :</b> <code>30.286804</code>	number (double)
<b>lng</b> <i>required</i>	Current GPS longitude <b>Example :</b> <code>-97.707425</code>	number (double)

## LostAndFoundRequestDto

Name	Schema
<b>content</b> <i>required</i>	string
<b>createdOn</b> <i>required</i>	string (date-time)
<b>type</b> <i>required</i>	enum (CALL, EMAIL)

## LostSuccessMessage

Name	Schema
<b>message</b> <i>required</i>	string

## MapDisplayConfig

Name	Schema
<b>enabled</b> <i>required</i>	boolean
<b>exclusionAreas</b> <i>required</i>	< <a href="#">AreaDto</a> > array
<b>iconCoordinate</b> <i>required</i>	<a href="#">LatLng</a>
<b>iconUrl</b> <i>required</i>	string
<b>waitingAreas</b> <i>required</i>	< <a href="#">AreaDto</a> > array

## MapInfoDto

Name	Schema
<b>activeDriver</b> <i>optional</i>	<a href="#">ActiveDriverInfo</a>
<b>id</b> <i>required</i>	integer (int64)
<b>rider</b> <i>required</i>	<a href="#">RiderInfo</a>
<b>startLocationLat</b> <i>required</i>	number (double)
<b>startLocationLong</b> <i>required</i>	number (double)
<b>status</b> <i>required</i>	enum (REQUEST_QUEUED, REQUEST_QUEUE_EXPIRED, REQUESTED, RIDER_CANCELLED, DRIVER_ASSIGNED, DRIVER_CANCELLED, DRIVER_REACHED, ACTIVE, NO_AVAILABLE_DRIVER, COMPLETED, ADMIN_CANCELLED)

## Map«string,string»

Type : < string, string > map

## MobileCar

Name	Schema
<b>color</b> <i>required</i>	string
<b>id</b> <i>required</i>	integer (int64)
<b>license</b> <i>required</i>	string
<b>make</b> <i>required</i>	string

Name	Schema
<b>model</b> <i>required</i>	string
<b>photoUrl</b> <i>required</i>	string
<b>year</b> <i>required</i>	string

## MobileDriverDriverDto

Name	Description	Schema
<b>active</b> <i>required</i>		boolean
<b>agreedToLega</b> lTerms <i>required</i>		boolean
<b>cars</b> <i>required</i>		< <a href="#">Car</a> > array
<b>chauffeurPer</b> mit <i>optional</i>		boolean
<b>cityId</b> <i>required</i>	<b>Example :</b> 1	integer (int64)
<b>deaf</b> <i>optional</i>		boolean
<b>directConnect</b> <b>Id</b> <i>required</i>		string
<b>email</b> <i>required</i>		string
<b>firstname</b> <i>optional</i>		string

Name	Description	Schema
<b>fullname</b> <i>optional</i>		string
<b>grantedDriver</b> <b>Types</b> <i>required</i>		< string > array
<b>id</b> <i>required</i>		integer (int64)
<b>lastname</b> <i>optional</i>		string
<b>nickName</b> <i>optional</i>		string
<b>phoneNumber</b> <i>required</i>		string
<b>photoUrl</b> <i>required</i>		string
<b>rating</b> <i>required</i>		number (double)
<b>type</b> <i>required</i>		enum (RIDER, DRIVER, ADMIN, API_CLIENT, DISPATCHER)
<b>user</b> <i>required</i>		User

## MobileDriverRideDto

Name	Schema
<b>comment</b> <i>optional</i>	string
<b>driverPayment</b> <i>required</i>	number

Name	Schema
<b>end</b> <i>optional</i>	<a href="#">Address</a>
<b>endAddress</b> <i>optional</i>	string
<b>endLocationLat</b> <i>required</i>	number (double)
<b>endLocationLong</b> <i>required</i>	number (double)
<b>estimatedTimeArrive</b> <i>optional</i>	integer (int64)
<b>freeCreditCharged</b> <i>optional</i>	number
<b>id</b> <i>required</i>	integer (int64)
<b>mapUrl</b> <i>optional</i>	string
<b>nextRide</b> <i>optional</i>	<a href="#">MobileDriverRideDto</a>
<b>requestedCarType</b> <i>required</i>	<a href="#">RequestedCarType</a>
<b>requestedDispatchType</b> <i>required</i>	enum (REGULAR, DIRECT_CONNECT)
<b>requestedDriverType</b> <i>required</i>	<a href="#">RequestedDriverType</a>
<b>requestedDriverTypes</b> <i>required</i>	< <a href="#">RequestedDriverType</a> > array
<b>rider</b> <i>required</i>	<a href="#">Rider</a>

Name	Schema
<b>start</b> <i>required</i>	<a href="#">Address</a>
<b>startAddress</b> <i>required</i>	string
<b>startLocationLat</b> <i>required</i>	number (double)
<b>startLocationLong</b> <i>required</i>	number (double)
<b>status</b> <i>required</i>	enum (REQUEST_QUEUED, REQUEST_QUEUE_EXPIRED, REQUESTED, RIDER_CANCELLED, DRIVER_ASSIGNED, DRIVER_CANCELLED, DRIVER_REACHED, ACTIVE, NO_AVAILABLE_DRIVER, COMPLETED, ADMIN_CANCELLED)
<b>surgeFactor</b> <i>required</i>	number
<b>upgradeRequest</b> <i>optional</i>	<a href="#">RideUpgradeRequestDto</a>

## MobileRiderCarDto

Name	Schema
<b>carPhotos</b> <i>required</i>	< string, string > map
<b>color</b> <i>required</i>	string
<b>id</b> <i>required</i>	integer (int64)
<b>license</b> <i>required</i>	string
<b>make</b> <i>required</i>	string

Name	Schema
<b>model</b> <i>required</i>	string
<b>photoUrl</b> <i>required</i>	string
<b>year</b> <i>required</i>	string

## MobileRiderDriverDto

Name	Schema
<b>active</b> <i>required</i>	boolean
<b>cars</b> <i>required</i>	< <a href="#">MobileCar</a> > array
<b>email</b> <i>required</i>	string
<b>firstname</b> <i>optional</i>	string
<b>fullname</b> <i>optional</i>	string
<b>grantedDriverTypes</b> <i>required</i>	< string > array
<b>grantedDriverTypesBitmask</b> <i>optional</i>	integer (int32)
<b>id</b> <i>required</i>	integer (int64)
<b>lastname</b> <i>optional</i>	string
<b>nickName</b> <i>optional</i>	string

Name	Schema
<b>phoneNumber</b> <i>required</i>	string
<b>photoUrl</b> <i>required</i>	string
<b>rating</b> <i>required</i>	number (double)
<b>type</b> <i>required</i>	enum (RIDER, DRIVER, ADMIN, API_CLIENT, DISPATCHER)
<b>user</b> <i>required</i>	User

## MobileRiderRideDto

Name	Schema
<b>activeDriver</b> <i>optional</i>	ActiveDriverDto
<b>comment</b> <i>optional</i>	string
<b>completedOn</b> <i>optional</i>	string (date-time)
<b>driverAcceptedOn</b> <i>optional</i>	string (date-time)
<b>driverPayment</b> <i>optional</i>	number
<b>driverRating</b> <i>optional</i>	number (double)
<b>end</b> <i>optional</i>	Address
<b>endAddress</b> <i>optional</i>	string

Name	Schema
<b>endLocationLat</b> <i>optional</i>	number (double)
<b>endLocationLong</b> <i>optional</i>	number (double)
<b>estimatedTimeArrive</b> <i>optional</i>	integer (int64)
<b>estimatedTimeCompletion</b> <i>optional</i>	string (date-time)
<b>freeCancellationExpiresOn</b> <i>optional</i>	string (date-time)
<b>freeCreditCharged</b> <i>optional</i>	number
<b>id</b> <i>required</i>	integer (int64)
<b>mapUrl</b> <i>optional</i>	string
<b>paymentProvider</b> <i>optional</i>	enum (CREDIT_CARD, APPLE_PAY)
<b>precedingRide</b> <i>optional</i>	<a href="#">PrecedingRide</a>
<b>requestedCarType</b> <i>required</i>	<a href="#">CityCarTypeDto</a>
<b>start</b> <i>optional</i>	<a href="#">Address</a>
<b>startAddress</b> <i>optional</i>	string
<b>startLocationLat</b> <i>required</i>	number (double)

Name	Schema
<b>startLocationLong</b> <i>required</i>	number (double)
<b>status</b> <i>required</i>	enum (REQUEST_QUEUED, REQUEST_QUEUE_EXPIRED, REQUESTED, RIDER_CANCELLED, DRIVER_ASSIGNED, DRIVER_CANCELLED, DRIVER_REACHED, ACTIVE, NO_AVAILABLE_DRIVER, COMPLETED, ADMIN_CANCELLED)
<b>tip</b> <i>optional</i>	number
<b>tipUntil</b> <i>optional</i>	string (date-time)
<b>tippingAllowed</b> <i>required</i>	boolean
<b>totalCharge</b> <i>optional</i>	number
<b>totalFare</b> <i>optional</i>	number
<b>upfrontCharge</b> <i>optional</i>	number
<b>upgradeRequest</b> <i>optional</i>	<a href="#">RideUpgradeRequestDto</a>

## Page«ActiveDriverDto»

Name	Schema
<b>content</b> <i>optional</i>	< <a href="#">ActiveDriverDto</a> > array
<b>first</b> <i>optional</i>	boolean
<b>last</b> <i>optional</i>	boolean

Name	Schema
<b>number</b> <i>optional</i>	integer (int32)
<b>numberOfElements</b> <i>optional</i>	integer (int32)
<b>size</b> <i>optional</i>	integer (int32)
<b>totalElements</b> <i>optional</i>	integer (int64)
<b>totalPages</b> <i>optional</i>	integer (int32)

## Page«CompactRideDto»

Name	Schema
<b>content</b> <i>optional</i>	< <a href="#">CompactRideDto</a> > array
<b>first</b> <i>optional</i>	boolean
<b>last</b> <i>optional</i>	boolean
<b>number</b> <i>optional</i>	integer (int32)
<b>numberOfElements</b> <i>optional</i>	integer (int32)
<b>size</b> <i>optional</i>	integer (int32)
<b>totalElements</b> <i>optional</i>	integer (int64)
<b>totalPages</b> <i>optional</i>	integer (int32)

## Page«CustomPaymentDto»

Name	Schema
<b>content</b> <i>optional</i>	< <a href="#">CustomPaymentDto</a> > array
<b>first</b> <i>optional</i>	boolean
<b>last</b> <i>optional</i>	boolean
<b>number</b> <i>optional</i>	integer (int32)
<b>numberOfElements</b> <i>optional</i>	integer (int32)
<b>size</b> <i>optional</i>	integer (int32)
<b>totalElements</b> <i>optional</i>	integer (int64)
<b>totalPages</b> <i>optional</i>	integer (int32)

## Page«DriverRidesReportEntry»

Name	Schema
<b>content</b> <i>optional</i>	< <a href="#">DriverRidesReportEntry</a> > array
<b>first</b> <i>optional</i>	boolean
<b>last</b> <i>optional</i>	boolean
<b>number</b> <i>optional</i>	integer (int32)

Name	Schema
<b>numberOfElements</b> <i>optional</i>	integer (int32)
<b>size</b> <i>optional</i>	integer (int32)
<b>totalElements</b> <i>optional</i>	integer (int64)
<b>totalPages</b> <i>optional</i>	integer (int32)

## Page«DriverRide»

Name	Schema
<b>content</b> <i>optional</i>	< DriverRide > array
<b>first</b> <i>optional</i>	boolean
<b>last</b> <i>optional</i>	boolean
<b>number</b> <i>optional</i>	integer (int32)
<b>numberOfElements</b> <i>optional</i>	integer (int32)
<b>size</b> <i>optional</i>	integer (int32)
<b>totalElements</b> <i>optional</i>	integer (int64)
<b>totalPages</b> <i>optional</i>	integer (int32)

## Page«ListPromocodeDto»

Name	Schema
<b>content</b> <i>optional</i>	< <a href="#">ListPromocodeDto</a> > array
<b>first</b> <i>optional</i>	boolean
<b>last</b> <i>optional</i>	boolean
<b>number</b> <i>optional</i>	integer (int32)
<b>numberOfElements</b> <i>optional</i>	integer (int32)
<b>size</b> <i>optional</i>	integer (int32)
<b>totalElements</b> <i>optional</i>	integer (int64)
<b>totalPages</b> <i>optional</i>	integer (int32)

## Page«PaymentHistoryDto»

Name	Schema
<b>content</b> <i>optional</i>	< <a href="#">PaymentHistoryDto</a> > array
<b>first</b> <i>optional</i>	boolean
<b>last</b> <i>optional</i>	boolean
<b>number</b> <i>optional</i>	integer (int32)

Name	Schema
<b>numberOfElements</b> <i>optional</i>	integer (int32)
<b>size</b> <i>optional</i>	integer (int32)
<b>totalElements</b> <i>optional</i>	integer (int64)
<b>totalPages</b> <i>optional</i>	integer (int32)

## Page«RedisSurgeArea»

Name	Schema
<b>content</b> <i>optional</i>	< <a href="#">RedisSurgeArea</a> > array
<b>first</b> <i>optional</i>	boolean
<b>last</b> <i>optional</i>	boolean
<b>number</b> <i>optional</i>	integer (int32)
<b>numberOfElements</b> <i>optional</i>	integer (int32)
<b>size</b> <i>optional</i>	integer (int32)
<b>totalElements</b> <i>optional</i>	integer (int64)
<b>totalPages</b> <i>optional</i>	integer (int32)

## Page«ReportMetadata»

Name	Schema
<b>content</b> <i>optional</i>	< <a href="#">ReportMetadata</a> > array
<b>first</b> <i>optional</i>	boolean
<b>last</b> <i>optional</i>	boolean
<b>number</b> <i>optional</i>	integer (int32)
<b>numberOfElements</b> <i>optional</i>	integer (int32)
<b>size</b> <i>optional</i>	integer (int32)
<b>totalElements</b> <i>optional</i>	integer (int64)
<b>totalPages</b> <i>optional</i>	integer (int32)

## Page«RideHistoryDto»

Name	Schema
<b>content</b> <i>optional</i>	< <a href="#">RideHistoryDto</a> > array
<b>first</b> <i>optional</i>	boolean
<b>last</b> <i>optional</i>	boolean
<b>number</b> <i>optional</i>	integer (int32)

Name	Schema
<b>numberOfElements</b> <i>optional</i>	integer (int32)
<b>size</b> <i>optional</i>	integer (int32)
<b>totalElements</b> <i>optional</i>	integer (int64)
<b>totalPages</b> <i>optional</i>	integer (int32)

## Page«SimpleDriverDto»

Name	Schema
<b>content</b> <i>optional</i>	< SimpleDriverDto > array
<b>first</b> <i>optional</i>	boolean
<b>last</b> <i>optional</i>	boolean
<b>number</b> <i>optional</i>	integer (int32)
<b>numberOfElements</b> <i>optional</i>	integer (int32)
<b>size</b> <i>optional</i>	integer (int32)
<b>totalElements</b> <i>optional</i>	integer (int64)
<b>totalPages</b> <i>optional</i>	integer (int32)

## Page«SimpleRiderDto»

Name	Schema
<b>content</b> <i>optional</i>	< SimpleRiderDto > array
<b>first</b> <i>optional</i>	boolean
<b>last</b> <i>optional</i>	boolean
<b>number</b> <i>optional</i>	integer (int32)
<b>numberOfElements</b> <i>optional</i>	integer (int32)
<b>size</b> <i>optional</i>	integer (int32)
<b>totalElements</b> <i>optional</i>	integer (int64)
<b>totalPages</b> <i>optional</i>	integer (int32)

## Page«UserTrackStatsDto»

Name	Schema
<b>content</b> <i>optional</i>	< UserTrackStatsDto > array
<b>first</b> <i>optional</i>	boolean
<b>last</b> <i>optional</i>	boolean
<b>number</b> <i>optional</i>	integer (int32)

Name	Schema
<b>numberOfElements</b> <i>optional</i>	integer (int32)
<b>size</b> <i>optional</i>	integer (int32)
<b>totalElements</b> <i>optional</i>	integer (int64)
<b>totalPages</b> <i>optional</i>	integer (int32)

## PaymentHistoryDto

Name	Schema
<b>campaignDescription</b> <i>optional</i>	string
<b>campaignDescriptionHistory</b> <i>optional</i>	string
<b>campaignDiscount</b> <i>optional</i>	number
<b>campaignProvider</b> <i>optional</i>	string
<b>cancelledOn</b> <i>optional</i>	string
<b>cancelledOnUTC</b> <i>optional</i>	string (date-time)
<b>carBrand</b> <i>required</i>	string
<b>carModel</b> <i>required</i>	string
<b>cardNumber</b> <i>optional</i>	string

Name	Schema
<b>completedOn</b> <i>optional</i>	string
<b>completedOnUTC</b> <i>optional</i>	string (date-time)
<b>driverFirstName</b> <i>required</i>	string
<b>driverId</b> <i>required</i>	integer (int64)
<b>driverLastName</b> <i>required</i>	string
<b>driverNickName</b> <i>optional</i>	string
<b>driverPicture</b> <i>required</i>	string
<b>driverRating</b> <i>required</i>	number (double)
<b>farePaymentId</b> <i>required</i>	integer (int64)
<b>freeCreditCharged</b> <i>required</i>	number
<b>mainRider</b> <i>optional</i>	boolean
<b>mainRiderFistName</b> <i>required</i>	string
<b>mainRiderId</b> <i>required</i>	integer (int64)
<b>mainRiderLastName</b> <i>required</i>	string

Name	Schema
<b>mainRiderPicture</b> <i>required</i>	string
<b>mapUrl</b> <i>optional</i>	string
<b>otherPaymentMethodUrl</b> <i>optional</i>	string
<b>rideEndAddress</b> <i>optional</i>	string
<b>rideId</b> <i>required</i>	integer (int64)
<b>rideStartAddress</b> <i>required</i>	string
<b>rideStatus</b> <i>required</i>	string
<b>rideTotalFare</b> <i>required</i>	number
<b>startedOn</b> <i>optional</i>	string
<b>stripeCreditCharge</b> <i>required</i>	number
<b>usedCardBrand</b> <i>optional</i>	string
<b>usedCardId</b> <i>optional</i>	integer (int64)

## PendingPaymentDto

Name	Schema
<b>amount</b> <i>required</i>	number

Name	Schema
<b>rideId</b> <i>required</i>	integer (int64)
<b>willChargeOn</b> <i>optional</i>	string (date-time)

## Point

Name	Schema
<b>x</b> <i>optional</i>	number (double)
<b>y</b> <i>optional</i>	number (double)

## Polygon

Name	Schema
<b>sides</b> <i>optional</i>	< <a href="#">Line</a> > array

## PrecedingRide

Name	Schema
<b>end</b> <i>required</i>	<a href="#">EndAddress</a>
<b>id</b> <i>required</i>	integer (int64)
<b>startAddress</b> <i>optional</i>	string
<b>startLocationLat</b> <i>required</i>	number (double)
<b>startLocationLong</b> <i>required</i>	number (double)

Name	Schema
<b>status</b> <i>required</i>	enum (REQUEST_QUEUED, REQUEST_QUEUE_EXPIRED, REQUESTED, RIDER_CANCELLED, DRIVER_ASSIGNED, DRIVER_CANCELLED, DRIVER_REACHED, ACTIVE, NO_AVAILABLE_DRIVER, COMPLETED, ADMIN_CANCELLED)

## PromocodeDto

Name	Schema
<b>applicableToFees</b> <i>required</i>	boolean
<b>cappedAmountPerUse</b> <i>required</i>	number
<b>carTypes</b> <i>required</i>	< string > array
<b>cities</b> <i>required</i>	< integer (int64) > array
<b>codeLiteral</b> <i>optional</i>	string
<b>codeValue</b> <i>required</i>	number
<b>currentRedemption</b> <i>required</i>	integer (int64)
<b>driverId</b> <i>required</i>	integer (int64)
<b>endsOn</b> <i>required</i>	string (date-time)
<b>id</b> <i>required</i>	integer (int64)
<b>maxPromotionValue</b> <i>required</i>	number

Name	Schema
<b>maximumRedemption</b> <i>required</i>	integer (int64)
<b>maximumUsesPerAccount</b> <i>required</i>	integer (int32)
<b>newRidersOnly</b> <i>required</i>	boolean
<b>nextTripOnly</b> <i>required</i>	boolean
<b>promocodeType</b> <i>required</i>	enum (USER, PUBLIC)
<b>startsOn</b> <i>required</i>	string (date-time)
<b>title</b> <i>required</i>	string
<b>usageCount</b> <i>required</i>	integer (int64)
<b>useEndDate</b> <i>required</i>	string (date-time)
<b>validForNumberOfDays</b> <i>required</i>	integer (int32)
<b>validForNumberOfRides</b> <i>required</i>	integer (int32)

## PromocodeRedemptionDTO

Name	Schema
<b>codeLiteral</b> <i>required</i>	string
<b>codeValue</b> <i>required</i>	number

Name	Schema
<b>createdDate</b> <i>required</i>	string (date-time)
<b>expiresOn</b> <i>required</i>	string (date-time)
<b>maximumUses</b> <i>required</i>	integer (int32)
<b>remainingValue</b> <i>required</i>	number
<b>timesUsed</b> <i>required</i>	integer (int32)

## RatingUpdateDto

Name	Schema
<b>comment</b> <i>optional</i>	string
<b>createdDate</b> <i>required</i>	string (date-time)
<b>id</b> <i>required</i>	integer (int64)
<b>ratedByFullName</b> <i>required</i>	string
<b>ratedById</b> <i>required</i>	integer (int64)
<b>ratedFullName</b> <i>required</i>	string
<b>ratedId</b> <i>required</i>	integer (int64)
<b>rating</b> <i>required</i>	number (double)

Name	Schema
<b>rideId</b> <i>required</i>	integer (int64)
<b>updatedDate</b> <i>required</i>	string (date-time)

## RedeemPromocodeDto

Name	Schema
<b>codeLiteral</b> <i>required</i>	string

## RedisSurgeArea

Name	Description	Schema
<b>automated</b> <i>required</i>		boolean
<b>bottomRightCornerLat</b> <i>optional</i>		number (double)
<b>bottomRightCornerLng</b> <i>optional</i>		number (double)
<b>carCategories</b> <i>optional</i>		< string > array
<b>carCategoriesAcceptedRides</b> <i>required</i>		< string, integer (int32) > map
<b>carCategoriesAvailableCars</b> <i>required</i>		< string, integer (int32) > map

Name	Description	Schema
<b>carCategories</b> <b>Bitmask</b> <i>required</i>	<b>Example : 1</b>	integer (int32)
<b>carCategories</b> <b>Cars</b> <i>required</i>		< string, integer (int32) > map
<b>carCategories</b> <b>Factors</b> <i>required</i>		< string, number > map
<b>carCategories</b> <b>NumberOfEye</b> <b>balls</b> <i>required</i>		< string, integer (int32) > map
<b>carCategories</b> <b>RequestedRides</b> <i>required</i>		< string, integer (int32) > map
<b>centerPointLat</b> <i>optional</i>		number (double)
<b>centerPointLng</b> <i>optional</i>		number (double)
<b>cityId</b> <i>required</i>	<b>Example : 1</b>	integer (int64)
<b>csvGeometry</b> <i>optional</i>		string
<b>geometry_id</b> <i>optional</i>		integer (int64)
<b>id</b> <i>required</i>	<b>Example : 1</b>	integer (int64)

Name	Description	Schema
<b>labelLat</b> <i>optional</i>		number (double)
<b>labelLng</b> <i>optional</i>		number (double)
<b>name</b> <i>required</i>		string
<b>recommendedSurgeMapping</b> <i>required</i>		< string, number > map
<b>topLeftCornerLat</b> <i>optional</i>		number (double)
<b>topLeftCornerLng</b> <i>optional</i>		number (double)

## RemainderResponse

Name	Schema
<b>remainder</b> <i>required</i>	number

## ReportMetadata

Name	Schema
<b>id</b> <i>required</i>	integer (int64)
<b>reportDescription</b> <i>required</i>	string
<b>reportFormat</b> <i>required</i>	enum (CSV, XLSX)

Name	Schema
<b>reportName</b> <i>required</i>	string

## ReportParameter

Name	Schema
<b>availableValues</b> <i>optional</i>	< object > array
<b>defaultValue</b> <i>required</i>	string
<b>order</b> <i>required</i>	integer (int32)
<b>parameterDescription</b> <i>required</i>	string
<b>parameterLabel</b> <i>required</i>	string
<b>parameterName</b> <i>required</i>	string
<b>parameterType</b> <i>required</i>	enum (STRING, INTEGER, DECIMAL, DATE, DATETIME, BOOLEAN, ENUM)
<b>required</b> <i>required</i>	boolean

## RequestedCarType

Name	Schema
<b>carCategory</b> <i>required</i>	string
<b>minimumFare</b> <i>required</i>	number

Name	Schema
<b>ratePerMile</b> <i>required</i>	number
<b>ratePerMinute</b> <i>required</i>	number
<b>title</b> <i>required</i>	string

## RequestedDriverType

Name	Schema
<b>name</b> <i>required</i>	string

## ResponseEntity

Name	Schema
<b>body</b> <i>optional</i>	object
<b>statusCode</b> <i>optional</i>	enum (100, 101, 102, 103, 200, 201, 202, 203, 204, 205, 206, 207, 208, 226, 300, 301, 302, 303, 304, 305, 307, 308, 400, 401, 402, 403, 404, 405, 406, 407, 408, 409, 410, 411, 412, 413, 414, 415, 416, 417, 418, 419, 420, 421, 422, 423, 424, 426, 428, 429, 431, 451, 500, 501, 502, 503, 504, 505, 506, 507, 508, 509, 510, 511)
<b>statusCodeValue</b> <i>optional</i>	integer (int32)

## RideCar

Name	Schema
<b>make</b> <i>required</i>	string

Name	Schema
<b>model</b> <i>required</i>	string

## RideEvents

Name	Schema
<b>events</b> <i>optional</i>	< Map«string,string» > array

## RideHistoryDto

Name	Schema
<b>cancelledOn</b> <i>optional</i>	string (date-time)
<b>completedOn</b> <i>optional</i>	string (date-time)
<b>driver</b> <i>required</i>	string
<b>endAddress</b> <i>required</i>	string
<b>estimatedFare</b> <i>optional</i>	number
<b>id</b> <i>required</i>	integer (int64)
<b>rider</b> <i>required</i>	string
<b>startAddress</b> <i>required</i>	string
<b>startedOn</b> <i>optional</i>	string (date-time)

Name	Schema
<b>status</b> <i>required</i>	enum (REQUEST_QUEUED, REQUEST_QUEUE_EXPIRED, REQUESTED, RIDER_CANCELLED, DRIVER_ASSIGNED, DRIVER_CANCELLED, DRIVER_REACHED, ACTIVE, NO_AVAILABLE_DRIVER, COMPLETED, ADMIN_CANCELLED)
<b>tip</b> <i>optional</i>	number
<b>tippedOn</b> <i>optional</i>	string (date-time)
<b>totalFare</b> <i>optional</i>	number

## RideReportEntry

Name	Schema
<b>averageDistanceTraveled</b> <i>required</i>	number
<b>averageDistanceTraveledInMiles</b> <i>optional</i>	number
<b>averageTotalFares</b> <i>required</i>	number (double)
<b>cancelledRidesCount</b> <i>required</i>	integer (int64)
<b>date</b> <i>required</i>	string (date-time)
<b>distanceTraveled</b> <i>required</i>	number
<b>distanceTraveledInMiles</b> <i>optional</i>	number
<b>priorityFaresRidesCount</b> <i>required</i>	integer (int64)

Name	Schema
<b>ridesCount</b> <i>required</i>	integer (int64)
<b>totalFares</b> <i>required</i>	number

## RideTrackingShareDto

Name	Schema
<b>cityId</b> <i>required</i>	integer (int64)
<b>completedOn</b> <i>required</i>	string
<b>createdDate</b> <i>required</i>	string
<b>currentCourse</b> <i>required</i>	number (double)
<b>currentHeading</b> <i>required</i>	number (double)
<b>currentHeadingDirection</b> <i>required</i>	enum (N, NE, E, SE, S, SW, W, NW)
<b>currentSpeed</b> <i>required</i>	number (double)
<b>driverCar</b> <i>required</i>	Car
<b>driverCarTypes</b> <i>required</i>	< string > array
<b>driverFirstName</b> <i>required</i>	string
<b>driverId</b> <i>required</i>	integer (int64)

Name	Schema
<b>driverLastName</b> <i>required</i>	string
<b>driverLicensePlate</b> <i>required</i>	string
<b>driverLocation</b> <i>required</i>	<a href="#">Location</a>
<b>driverPhoto</b> <i>required</i>	string
<b>driverRating</b> <i>required</i>	number (double)
<b>endAddress</b> <i>required</i>	string
<b>endLocation</b> <i>required</i>	<a href="#">Location</a>
<b>locations</b> <i>required</i>	< <a href="#">Location</a> > array
<b>rideCarCategory</b> <i>required</i>	string
<b>rideId</b> <i>required</i>	integer (int64)
<b>riderFirstName</b> <i>required</i>	string
<b>riderId</b> <i>required</i>	integer (int64)
<b>riderLastName</b> <i>required</i>	string
<b>riderPhoto</b> <i>required</i>	string

Name	Schema
<b>startAddress</b> <i>required</i>	string
<b>startLocation</b> <i>required</i>	<a href="#">Location</a>
<b>startedOn</b> <i>required</i>	string
<b>status</b> <i>required</i>	enum (REQUEST_QUEUED, REQUEST_QUEUE_EXPIRED, REQUESTED, RIDER_CANCELLED, DRIVER_ASSIGNED, DRIVER_CANCELLED, DRIVER_REACHED, ACTIVE, NO_AVAILABLE_DRIVER, COMPLETED, ADMIN_CANCELLED)
<b>updatedDate</b> <i>required</i>	string

## RideUpgradeRequestDto

Name	Schema
<b>source</b> <i>required</i>	string
<b>status</b> <i>required</i>	enum (REQUESTED, ACCEPTED, DECLINED, CANCELLED, EXPIRED)
<b>surgeFactor</b> <i>required</i>	number
<b>target</b> <i>required</i>	string

## RideUpgradeResponse

Name	Schema
<b>message</b> <i>required</i>	string

# Rider

Name	Schema
<b>active</b> <i>optional</i>	boolean
<b>charity</b> <i>optional</i>	Charity
<b>cityId</b> <i>optional</i>	integer (int64)
<b>dispatcherAccount</b> <i>optional</i>	boolean
<b>email</b> <i>optional</i>	string
<b>facebookId</b> <i>optional</i>	string
<b>firstname</b> <i>optional</i>	string
<b>fullName</b> <i>optional</i>	string
<b>gender</b> <i>optional</i>	enum (MALE, FEMALE, OTHER, UNKNOWN)
<b>id</b> <i>optional</i>	integer (int64)
<b>lastLoginDate</b> <i>optional</i>	string (date-time)
<b>lastname</b> <i>optional</i>	string
<b>phoneNumber</b> <i>optional</i>	string
<b>rating</b> <i>optional</i>	number (double)

Name	Schema
<b>stripeId</b> <i>optional</i>	string
<b>type</b> <i>optional</i>	enum (RIDER, DRIVER, ADMIN, API_CLIENT, DISPATCHER)
<b>user</b> <i>optional</i>	User

## RiderCardDto

Name	Description	Schema
<b>cardBrand</b> <i>required</i>		enum (VISA, MASTERCARD, AMERICAN_EXPRES S, DISCOVER, JCB, DINERS_CLUB, UNKNOWN)
<b>cardExpired</b> <i>required</i>		boolean
<b>cardNumber</b> <i>required</i>		string
<b>expirationMo nth</b> <i>required</i>		string
<b>expirationYea r</b> <i>required</i>		string
<b>id</b> <i>required</i>	Example : 1	integer (int64)
<b>primary</b> <i>required</i>		boolean

## RiderDto

Name	Schema
<b>active</b> <i>required</i>	boolean
<b>cardExpired</b> <i>required</i>	boolean
<b>charity</b> <i>required</i>	<a href="#">CharityDto</a>
<b>email</b> <i>required</i>	string
<b>firstname</b> <i>required</i>	string
<b>fullName</b> <i>required</i>	string
<b>id</b> <i>required</i>	integer (int64)
<b>isDispatcher</b> <i>required</i>	boolean
<b>lastname</b> <i>required</i>	string
<b>phoneNumber</b> <i>required</i>	string
<b>rating</b> <i>required</i>	number (double)
<b>type</b> <i>required</i>	enum (RIDER, DRIVER, ADMIN, API_CLIENT, DISPATCHER)
<b>user</b> <i>required</i>	<a href="#">UserDto</a>

## RiderInfo

Name	Schema
<b>id</b> <i>required</i>	integer (int64)

## RiderPromoCodeDto

Name	Schema
<b>codeLiteral</b> <i>required</i>	string
<b>codeValue</b> <i>required</i>	number
<b>currentRedemption</b> <i>required</i>	integer (int64)
<b>detailText</b> <i>required</i>	string
<b>emailBody</b> <i>required</i>	string
<b>maximumRedemption</b> <i>required</i>	integer (int64)
<b>remainingCredit</b> <i>required</i>	number
<b>smsBody</b> <i>required</i>	string

## SimpleCarDto

Name	Description	Schema
<b>carPhotosStat</b> <b>us</b> <i>required</i>	Status of car photos	< string, string > map

Name	Description	Schema
<b>categories</b> <i>required</i>		< string > array
<b>color</b> <i>required</i>		string
<b>driverId</b> <i>required</i>		integer (int64)
<b>id</b> <i>required</i>		integer (int64)
<b>inspectionStatus</b> <i>required</i>		enum (APPROVED, REJECTED, PENDING, NOT_INSPECTED)
<b>inspectionStickerStatus</b> <i>required</i>		enum (APPROVED, PENDING, EXPIRED, REJECTED, NOT_PROVIDED)
<b>insurancePhotoUrl</b> <i>required</i>		string
<b>insuranceStatus</b> <i>required</i>		enum (APPROVED, PENDING, EXPIRED, REJECTED, NOT_PROVIDED)
<b>license</b> <i>required</i>		string
<b>make</b> <i>required</i>		string
<b>model</b> <i>required</i>		string
<b>year</b> <i>required</i>		string

## SimpleDriverDto

Name	Schema
<b>activationStatus</b> <i>required</i>	enum (ACTIVE, REJECTED, SUSPENDED, DEACTIVATED_OTHER, INACTIVE)
<b>active</b> <i>required</i>	boolean
<b>cars</b> <i>required</i>	< <a href="#">SimpleCarDto</a> > array
<b>cityApprovalStatus</b> <i>required</i>	enum (PENDING, NOT_PROVIDED, APPROVED, REJECTED_PHOTO, REJECTED_BY_CITY, EXPIRED)
<b>driverId</b> <i>required</i>	integer (int64)
<b>driverLicensePicture</b> <i>required</i>	string
<b>driverLicenseStatus</b> <i>required</i>	enum (APPROVED, PENDING, EXPIRED, REJECTED, NOT_PROVIDED)
<b>driverPicture</b> <i>optional</i>	string
<b>driverTypes</b> <i>required</i>	< string > array
<b>email</b> <i>required</i>	string
<b>enabled</b> <i>required</i>	boolean
<b>firstName</b> <i>required</i>	string
<b>gender</b> <i>required</i>	enum (MALE, FEMALE, OTHER, UNKNOWN)
<b>lastLoginDate</b> <i>optional</i>	string (date-time)

Name	Schema
<b>lastName</b> <i>required</i>	string
<b>onboardingStatus</b> <i>required</i>	enum (REJECTED, SUSPENDED, PENDING, FINAL_REVIEW, ACTIVE)
<b>payoneerStatus</b> <i>required</i>	enum (Not registered, Inactive, Active)
<b>phoneNumber</b> <i>required</i>	string
<b>profilePhotosStatus</b> <i>required</i>	enum (APPROVED, PENDING, EXPIRED, REJECTED, NOT_PROVIDED)
<b>rating</b> <i>required</i>	number (double)
<b>ratingAverage</b> <i>required</i>	number
<b>userId</b> <i>required</i>	integer (int64)
<b>userPicture</b> <i>optional</i>	string

## SimpleRiderDto

Name	Schema
<b>active</b> <i>required</i>	boolean
<b>email</b> <i>required</i>	string
<b>enabled</b> <i>required</i>	boolean
<b>firstName</b> <i>required</i>	string

Name	Schema
<b>lastName</b> <i>required</i>	string
<b>phoneNumber</b> <i>required</i>	string
<b>riderId</b> <i>required</i>	integer (int64)
<b>riderPictureUrl</b> <i>required</i>	string
<b>userId</b> <i>required</i>	integer (int64)

## SpecialFee

Name	Schema
<b>description</b> <i>optional</i>	string
<b>title</b> <i>optional</i>	string
<b>value</b> <i>optional</i>	number
<b>valueType</b> <i>optional</i>	enum (AMOUNT, RATE)

## SplitFareDto

Name	Schema
<b>createdDate</b> <i>required</i>	string
<b>id</b> <i>required</i>	integer (int64)

Name	Schema
<b>rideId</b> <i>required</i>	integer (int64)
<b>riderFullName</b> <i>required</i>	string
<b>riderId</b> <i>required</i>	integer (int64)
<b>riderPhoto</b> <i>required</i>	string
<b>sourceRiderFullName</b> <i>required</i>	string
<b>sourceRiderPhotoURL</b> <i>required</i>	string
<b>status</b> <i>required</i>	enum (REQUESTED, ACCEPTED, DECLINED)
<b>updatedDate</b> <i>required</i>	string

## SupportRequestDto

Name	Description	Schema
<b>comments</b> <i>optional</i>	Support request comments	string
<b>rideId</b> <i>optional</i>	Ride ID	integer (int64)
<b>topicId</b> <i>required</i>	Support topic ID	integer (int64)

## SupportTopicDto

Name	Schema
<b>avatarType</b> <i>required</i>	enum (RIDER, DRIVER, ADMIN, API_CLIENT, DISPATCHER)
<b>description</b> <i>required</i>	string
<b>followUpTypes</b> <i>required</i>	< enum (EMAIL, SMS, CALL) > array
<b>hasChildren</b> <i>required</i>	boolean
<b>hasForms</b> <i>required</i>	boolean
<b>id</b> <i>required</i>	integer (int64)
<b>parent</b> <i>optional</i>	SupportTopicDto

## SupportTopicFormDto

Name	Schema
<b>actionTitle</b> <i>required</i>	string
<b>actionType</b> <i>required</i>	string
<b>body</b> <i>required</i>	string
<b>headerText</b> <i>required</i>	string
<b>id</b> <i>required</i>	integer (int64)
<b>supportFields</b> <i>required</i>	< Field > array

Name	Schema
<b>title</b> <i>required</i>	string

## SurgeAreaDto

Name	Description	Schema
<b>automated</b> <i>optional</i>		boolean
<b>bottomRightCornerLat</b> <i>required</i>		number (double)
<b>bottomRightCornerLng</b> <i>required</i>		number (double)
<b>centerPointLat</b> <i>required</i>		number (double)
<b>centerPointLng</b> <i>required</i>		number (double)
<b>cityId</b> <i>required</i>	<b>Example :</b> 1	integer (int64)
<b>csvGeometry</b> <i>required</i>		string
<b>id</b> <i>required</i>		integer (int64)
<b>labelLat</b> <i>required</i>		number (double)
<b>labelLng</b> <i>required</i>		number (double)

Name	Description	Schema
<b>name</b> <i>required</i>		string
<b>surgeFactors</b> <i>required</i>	A String->BigDecimal map containing mapping between car category and surge factor	< string, number > map
<b>topLeftCorner</b> <b>Lat</b> <i>required</i>		number (double)
<b>topLeftCorner</b> <b>Lng</b> <i>required</i>		number (double)

## SurgeRecalculationConfig

Name	Schema
<b>defaultAreaMonitoringPeriod</b> <i>required</i>	integer (int32)
<b>maxAuthorizedLimitedAutoValue</b> <i>required</i>	number
<b>surgeEquationMapping</b> <i>required</i>	< string, number > map
<b>surgeMode</b> <i>required</i>	enum (FULL_AUTO, LIMITED_AUTO, MANUAL)
<b>surgeProvider</b> <i>required</i>	enum (STATS)
<b>utilizationThreshold</b> <i>required</i>	number

## Topic

Name	Schema
<b>arn</b> <i>required</i>	string

Name	Schema
<b>description</b> <i>optional</i>	string
<b>id</b> <i>required</i>	integer (int64)
<b>name</b> <i>required</i>	string
<b>subscriptionPolicyClassName</b> <i>required</i>	string

## TrackingShareToken

Name	Schema
<b>token</b> <i>optional</i>	string

## URI

Name	Schema
<b>absolute</b> <i>optional</i>	boolean
<b>authority</b> <i>optional</i>	string
<b>fragment</b> <i>optional</i>	string
<b>host</b> <i>optional</i>	string
<b>opaque</b> <i>optional</i>	boolean
<b>path</b> <i>optional</i>	string

Name	Schema
<b>port</b> <i>optional</i>	integer (int32)
<b>query</b> <i>optional</i>	string
<b>rawAuthority</b> <i>optional</i>	string
<b>rawFragment</b> <i>optional</i>	string
<b>rawPath</b> <i>optional</i>	string
<b>rawQuery</b> <i>optional</i>	string
<b>rawSchemeSpecificPart</b> <i>optional</i>	string
<b>rawUserInfo</b> <i>optional</i>	string
<b>scheme</b> <i>optional</i>	string
<b>schemeSpecificPart</b> <i>optional</i>	string
<b>userInfo</b> <i>optional</i>	string

## UpdateRequest

Name	Schema
<b>configurationValue</b> <i>required</i>	object

## UpdateUserDto

Name	Schema
<b>firstname</b> <i>optional</i>	string
<b>gender</b> <i>optional</i>	enum (MALE, FEMALE, OTHER, UNKNOWN)
<b>lastname</b> <i>optional</i>	string
<b>nickName</b> <i>optional</i>	string
<b>phoneNumber</b> <i>optional</i>	string

## UrlDto

Name	Schema
<b>url</b> <i>optional</i>	string

## User

Name	Schema
<b>address</b> <i>optional</i>	Address
<b>avatars</b> <i>optional</i>	< Info > array
<b>dateOfBirth</b> <i>optional</i>	string (date-time)
<b>email</b> <i>optional</i>	string
<b>emailVerified</b> <i>optional</i>	boolean

Name	Schema
<b>enabled</b> <i>optional</i>	boolean
<b>facebookId</b> <i>optional</i>	string
<b>firstname</b> <i>optional</i>	string
<b>fullName</b> <i>optional</i>	string
<b>gender</b> <i>optional</i>	enum (MALE, FEMALE, OTHER, UNKNOWN)
<b>id</b> <i>optional</i>	integer (int64)
<b>lastname</b> <i>optional</i>	string
<b>middleName</b> <i>optional</i>	string
<b>nickName</b> <i>optional</i>	string
<b>phoneNumber</b> <i>optional</i>	string
<b>phoneNumberVerified</b> <i>optional</i>	boolean
<b>photoUrl</b> <i>optional</i>	string
<b>rawPassword</b> <i>optional</i>	string

## UserDto

Name	Schema
<b>active</b> <i>required</i>	boolean
<b>address</b> <i>optional</i>	<a href="#">Address</a>
<b>avatars</b> <i>required</i>	< <a href="#">AvatarDto</a> > array
<b>dateOfBirth</b> <i>optional</i>	string (date-time)
<b>deviceBlocked</b> <i>required</i>	boolean
<b>email</b> <i>required</i>	string
<b>enabled</b> <i>required</i>	boolean
<b>facebookId</b> <i>optional</i>	string
<b>firstname</b> <i>required</i>	string
<b>fullName</b> <i>required</i>	string
<b>gender</b> <i>required</i>	enum (MALE, FEMALE, OTHER, UNKNOWN)
<b>id</b> <i>required</i>	integer (int64)
<b>lastname</b> <i>required</i>	string
<b>middleName</b> <i>optional</i>	string

Name	Schema
<b>nickName</b> <i>optional</i>	string
<b>nickname</b> <i>optional</i>	string
<b>phoneNumber</b> <i>required</i>	string
<b>photoUrl</b> <i>required</i>	string

## UserTrackStatsDto

Name	Schema
<b>campaign</b> <i>required</i>	string
<b>distance</b> <i>required</i>	number (double)
<b>driverPayment</b> <i>required</i>	number
<b>fare</b> <i>required</i>	number
<b>medium</b> <i>required</i>	string
<b>raPayment</b> <i>required</i>	number
<b>rides</b> <i>required</i>	integer (int64)
<b>source</b> <i>required</i>	string

## ZipCodeReportEntry

Name	Schema
<b>rideCount</b> <i>required</i>	integer (int64)
<b>zipCode</b> <i>required</i>	string

## Security

### basic

*Type* : basic