****

This document is licensed under the Creative Commons **Attribution-NonCommercial-ShareAlike 3.0 Unported (CC BY-NC-SA 3.0)** License. To view a copy of this license, visit http://creativecommons.org/licenses/by-nc-sa/3.0/; or, (b) send a letter to Creative Commons, 171 2nd Street, Suite 300, San Francisco, California, 94105, USA.”

Copyright (C) 2010 Fraunhofer Institute for Open Communication Systems (FOKUS)  
Fraunhofer FOKUS  
Kaiserin-Augusta-Allee 31  
10589 Berlin  
Tel: +49 30 3463-7000  
[*info@fokus.fraunhofer.de*](mailto:info@fokus.fraunhofer.de)

# OpenRide REST API Documentation

Every action can return a 404 Not Found (username does not exist) Status Code.

Actions which are not marked “public” return 403 Forbidden when invoked by unauthorized users (i.e. by anyone who isn’t the account owner)

(Offset parameters are not implemented yet, but may be added at a later time for list pagination.)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Action ‚shortname‘ | Description | | Operation  (request method + URI[[1]](#footnote-1)) | Request Contents  (JSON) | Response Contents  (JSON / XML) | Response Status Codes | Public? |
| User Account / Profile | | | | | | | |
| logout | This indicates that the user wants to end up his session. | | PUT /configuration/signoff | None | None | 200 OK | No |
| (register) | This actionname can be used to identify a register request.  **- not implemented yet -** | | POST /users | Name: String,  Username: String,  Password: String(MD5),  … | None | ... | n/a |
| status | On first page load, client request some general user and state infos (like open searches, ratings etc..) | | GET /configuration/init | None | InitResponse(nickname: string, openratings: int, openoffers: int, opensearches: int, updatedoffers: int, updatedsearches: int, profilpic: string[[2]](#footnote-2)) | 200 OK | No |
| profile | Client requests profile data to display on the gui. | | GET /users/{[other\_]username}/profile | None | ProfileResponse(firstName: string, lastName: string, gender: char[[3]](#footnote-3), dateOfBirth: long, email: string, fixedPhoneNumber: string, mobilePhoneNumber: string, streetAddress: string, zipCode: int, city: string, isSmoker: char[[4]](#footnote-4), licenseDate: long) | 200 OK | No |
| profile | Client sends complete profile data back to server after editing.  TODO: it could be improved by only sending changed data | | PUT /users/{username}/profile | ProfileRequest (dateOfBirth: long, email: string, fixedPhoneNumber: string, mobilePhoneNumber: string, streeAddress: string, zipCode: int, city: string, isSmoker: char, licenseDate: long) | None | 204 No Content,  400 Bad Request (invalid data) | No |
| preferences | Client requests current preferences to display on the gui. | | GET /users/{[other\_]username}/profile/preferences | None | PreferenceResponse(isSmoker: char, gender: char) | 200 OK | Yes? |
| preferences | Client sends updated preferences to server after editing. | | PUT /users/{username}/profile/preferences | PreferenceRequest(isSmoker: char, gender: char) | None | 204 No Content,  400 Bad Request (invalid data) | No |
| profilepicture | Client sends new profile picture to be stored on the server. | | POST /users/{username}/ profile/picture | image\_data (multipart/form-data) | None | None | No |
| password | This actionname indicates the wish to change the existing password of the logged-in customers account. | | PUT /users/{username}/profile/password | PasswordRequest(passwordOld: string,  password: string)  (not as MD5 – server will convert) | None | 200 OK,  400 Bad Request (wrong old password) | No |
| Routes | | | | | | | |
| getroute | Client provides start and dest lat/lon pairs, server generates and responds with an xml-based route | | GET /users/{username}/routes/new/{startlat},{startlon},{destlat},{destlon} | None | Xml-doc (as in old client version)  containing price or distance?? | 200 OK,  400 Bad Request (invalid coords) | No |
| currentroute | This actionname indicates that the currently active route is of interest.  (Client requests the users currentroute = there's a running ride (maybe also for active, non-running rides??) and the server generates the xml for the route coords, and the driver/riders positions) | | GET /users/{username}/routes/current | None | Xml-doc (as in old client version) | 200 OK | No |
| Rides | | | | | | | |
| inactivesearches | | Client requests list of rides the user has accomplished with openride. The client uses the data for list display and for detailed ride info | GET /users/{username}/rides/searches/inactive | None | List(  Search(riderRouteId: int, riderId: int, ridestartPtLat: double, ridestartPtLon: double, rideendPtLat: double, rideendPtLon: double, ridestartTimeLatest: long, rideComment: string, maxwaitingtime: int, searchedSeatsNo: int, savetemplate: bool, ridestartTimeEarliest: long, price: double, startptAddress: sting, endptAddress: string, updated: bool)  ) | 200 OK | No |
| inactiveoffers | |  | GET /users/{username}/rides/offers/inactive | None | List(  Offer(riderId: int, ridestartPtLat: double, ridestartPtLon: double, rideendPtLat: double, rideendPtLon: double, ridestartTime: long,  rideprice: double, rideComment: string,  acceptableDetourInMin: int, acceptableDetourInKm: int, acceptableDetourInPercent: int, offeredSeatsNo: int, startptAddress: sting, endptAddress: string, updated: bool, intermediatePoints: List(Point))  ) | 200 OK | No |
| activesearches | | This actionname can be used to get the currently active searches of the logged-in customer. (Analogous to getactiveoffers) | GET /users/{username}/rides/searches | None | List(  Search(riderRouteId: int, riderId: int, ridestartPtLat: double, ridestartPtLon: double, rideendPtLat: double, rideendPtLon: double, ridestartTimeLatest: long, rideComment: string, maxwaitingtime: int, searchedSeatsNo: int, savetemplate: bool, ridestartTimeEarliest: long, price: double, startptAddress: sting, endptAddress: string, updated: bool)  ) | 200 OK | No |
| search | | This actionname can be used to generat a new search request.  (Client sends full new search data to server) | POST /users/{username}/rides/searches | Search(riderRouteId: int, riderId: int, ridestartPtLat: double, ridestartPtLon: double, rideendPtLat: double, rideendPtLon: double, ridestartTimeLatest: long, rideComment: string, maxwaitingtime: int, searchedSeatsNo: int, savetemplate: bool, ridestartTimeEarliest: long, price: double, startptAddress: sting, endptAddress: string, updated: bool) | PostSearchResponse(rideId: int) | 200 OK,  400 Bad Request (invalid data) | No |
| (searchtemplates) | | **- not implemented yet -** | GET /users/{username}/rides/searches/templates | None | List | 200 OK | No |
| searchdetails | |  | GET /users/{username}/rides/searches/{ride\_id} | None | Search(riderRouteId: int, riderId: int, ridestartPtLat: double, ridestartPtLon: double, rideendPtLat: double, rideendPtLon: double, ridestartTimeLatest: long, rideComment: string, maxwaitingtime: int, searchedSeatsNo: int, savetemplate: bool, ridestartTimeEarliest: long, price: double, startptAddress: sting, endptAddress: string, updated: bool) | 200 OK,  400 Bad Request (invalid rideId) | No |
| editsearch | |  | PUT /users/{username}/rides/searches/{ride\_id} | Search(riderRouteId: int, riderId: int, ridestartPtLat: double, ridestartPtLon: double, rideendPtLat: double, rideendPtLon: double, ridestartTimeLatest: long, rideComment: string, maxwaitingtime: int, searchedSeatsNo: int, savetemplate: bool, ridestartTimeEarliest: long, price: double, startptAddress: sting, endptAddress: string, updated: bool) | PostSearchResponse(rideId: int) | 200 OK,  400 Bad Request (invalid rideId or data) | No |
| removesearch | |  | DELETE /users/{username}/rides/searches/{ride\_id} | None | None | 200 OK,  400 Bad Request (invalid rideId) | No |
| activeoffers | | This actionname can be used to get a list of the currently active offers of the logged-in customer.  (Client request list of all active offers with all attributes, packed as an array of jason objects. client can then extract attributes for list display or for display of a single complete active offer  TODO:  if intelligent list-caching will be implemented, the list entries could be loaded step by step, as the user scrolls down the list) | GET /users/{username}/rides/offers | None | List(  Offer(riderId: int, ridestartPtLat: double, ridestartPtLon: double, rideendPtLat: double, rideendPtLon: double, ridestartTime: long,  rideprice: double, rideComment: string,  acceptableDetourInMin: int, acceptableDetourInKm: int, acceptableDetourInPercent: int, offeredSeatsNo: int, startptAddress: sting, endptAddress: string, updated: bool, intermediatePoints: List(Point))  ) | 200 OK | No |
| (updatetrack) | | This actionname can be used to reference the tracker Unit.  **- not implemented yet -** | PUT /users/{username}/rides/offers/{ride\_id}/tracker | Point currentPoint | None |  | No |
| (track) | | This actionname can be used to reference the tracker Unit.  **- not implemented yet -** | GET /users/{username}/rides/offers/{ride\_id}/tracker | None | Point currentPoint |  | Yes |
| offer | | This actionname can be used to generate a new offer | POST /users/{username}/rides/offers | Offer(riderId: int, ridestartPtLat: double, ridestartPtLon: double, rideendPtLat: double, rideendPtLon: double, ridestartTime: long,  rideprice: double, rideComment: string,  acceptableDetourInMin: int, acceptableDetourInKm: int, acceptableDetourInPercent: int, offeredSeatsNo: int, startptAddress: sting, endptAddress: string, updated: bool, intermediatePoints: List(Point)) | PostOfferResponse(rideId: int) | 200 OK,  400 Bad Request (invalid data) | No |
| (offertemplates) | | This actionname can be used to get a list of templates for ride-offers  **- not implemented yet -** | GET /users/{username}/rides/offers/templates | None | List | 200 OK | No |
| offerdetails | |  | GET /users/{username}/rides/offers/{ride\_id} | None | Offer(riderId: int, ridestartPtLat: double, ridestartPtLon: double, rideendPtLat: double, rideendPtLon: double, ridestartTime: long,  rideprice: double, rideComment: string,  acceptableDetourInMin: int, acceptableDetourInKm: int, acceptableDetourInPercent: int, offeredSeatsNo: int, startptAddress: sting, endptAddress: string, updated: bool, intermediatePoints: List(Point)) | 200 OK,  400 Bad Request (invalid rideId) | No |
| editoffer | |  | PUT /users/{username}/rides/offers/{ride\_id} | Offer(riderId: int, ridestartPtLat: double, ridestartPtLon: double, rideendPtLat: double, rideendPtLon: double, ridestartTime: long,  rideprice: double, rideComment: string,  acceptableDetourInMin: int, acceptableDetourInKm: int, acceptableDetourInPercent: int, offeredSeatsNo: int, startptAddress: sting, endptAddress: string, updated: bool, intermediatePoints: List(Point)) | PostOfferResponse(rideId: int) | 200 OK,  400 Bad Request (invalid rideId or data) | No |
| removeoffer | |  | DELETE /users/{username}/rides/offers/{ride\_id} | None | None | 200 OK,  400 Bad Request (invalid rideId) | No |
| Booking | | | | | | | |
| acceptdrivermatch | | This actionname indicates that the logged-in user accepts the currently matched drivers‘ offer. | PUT /users/{username}/rides/searches/{riderrouteId}/matches/{rideId}/accept | None | None | 200 OK,  400 Bad Request (invalid rideId) | Yes |
| rejectdrivermatch | | This actionname indicates that the logged-in user rejects the currently matched drivers‘ offer. | PUT /users/{username}/rides/searches/{riderrouteId}/matches/{rideId}/reject | None | None | 200 OK,  400 Bad Request (invalid rideId) | Yes |
| offermatches | |  | GET /users/{username}/rides/offers/{rideId}/matches | None | List(  MatchResponse(driverState: bool[[5]](#footnote-5), riderState: bool, matchSharedDistanceMeters: double, matchDetourMeters: double, matchExpectedStartTime: long, rideId: int, riderRouteId: int, riderCustId: int, riderNickname: string, riderRatingsRatioPercent: int, riderMobilePhoneNumber: string, matchPriceCents: int, startPtAddress: string, endPtAddress: string)  ) | 200 OK,  400 Bad Request (invalid rideId) | No |
| searchmatches | |  | GET /users/{username}/rides/searches/{riderrouteId}/matches | None | List(  MatchResponse(driverState: bool, riderState: bool, matchExpectedStartTime: long, rideId: int, riderRouteId: int, driverCustId: int, driverNickname: string, driverRatingsRatioPercent: int, driverMobilePhoneNumber: string, driverCarDescription: string, matchPriceCents: int, startPtAddress: string, endPtAddress: string)  ) | 200 OK,  400 Bad Request (invalid riderrouteId) |  |
| acceptridermatch | | This actionname indicates that the logged-in user accepts a currently matched rider. | PUT /users/{username}/rides/offers/{rideId}/matches/{riderrouteId}/accept | None | None | 200 OK,  400 Bad Request (invalid rideId) | Yes |
| Rejectridermatch | | This actionname indicates that the logged-in user rejects a currently matched rider. | PUT /users/{username}/rides/offers/{rideId}/matches/{riderrouteId}/reject | None | None | 200 OK,  400 Bad Request (invalid rideId) | Yes |
| Ratings | | | | | | | |
| ratings | This actionname can be used to get the current ratings of a customer. | | GET /users/{[other\_]username}/ratings | offset? | List (  ReceivedRatingResponse(custId: int, custNickname: string, custGender: char, custRole[[6]](#footnote-6): char, timestamprealized: long, receivedrating: int, receivedratingComment: string)  ) | 200 OK | No? |
| ratingsSummary | This actionname can be used to get the ratings summary of a customer. | | GET /users/{[other\_]username}/ratings/summary | None | RatingsSummary(ratingsTotal: int, ratingsRatioPercent: int, ratingsLatestPositive: int, ratingsLatestNeutral: int, ratingsLatestNegative: int) | 200 OK | Yes? |
| rating | Client sends rating for a specific ride. | | POST /users/{other\_username}/ratings | GivenRatingRequest(riderRouteId: int, rating: int, ratingComment: string) | None | 200 OK,  400 Bad Request (invalid riderRouteId / rating) | Yes |
| openRatings | Client requests list of rides that are missing his/her rating. | | GET /users/{username}/ratings/open | offset? | List ( OpenRatingResponse(riderRouteId: int, custId: int, custNickname: string, custGender: char, custRole: char, timestamprealized: long)  ) | 200 OK | No |
| Favorite Points | | | | | | | |
| favpoints | Client requests list of currently stored favorite points. | | GET /users/{username}/favoritepoints | offset? | List ( FavoritePointResponse(favptId: int, favptAddress: string, favptGeoCoords: string, favptDisplayName: string) ) | 200 OK | No |
| favpoint | Client adds favorite point. | | POST /users/{username}/favoritepoints | FavoritePointRequest(favptAddress: string, favptGeoCoords: string, favptDisplayName: string) | None | 201 Created,  400 Bad Request (displayname already exists) | No |
| favpoint | Client removes favorite point. | | DELETE /users/{username}/favoritepoints/{displayName} | None | None | 200 OK,  404 Not Found | No |

1. The method URIs should also include a „versioning prefix“, i.e. all URIs should begin with „/v1/...“ (backwards compatibility for future API changes).   
   **(not implemented yet)** [↑](#footnote-ref-1)
2. Containing its URL. [↑](#footnote-ref-2)
3. Gender char may be one of: „m“, „f“ [↑](#footnote-ref-3)
4. isSmoker char may be one of: „y“, „n“, „-“ (yes/no/won’t tell) [↑](#footnote-ref-4)
5. driverState and riderState may be one of: true, false, null (see „Buttons\_Stati\_Abstimmungsprozess.docx“) [↑](#footnote-ref-5)
6. custRole may be of: “d”, “r” (i.e., was the customer who gave this rating driver or rider?) [↑](#footnote-ref-6)