School of Information Technology

Nanyang polytechnic

LIonPortal API GAtewaY

Table of Contents

[Functions in Backend (AWS API Gateway) 4](#_Toc49100629)

[ANNOUNCEMENT 6](#_Toc49100630)

[Get List of Announcements 6](#_Toc49100631)

[Add new Announcement 6](#_Toc49100632)

[Delete Announcement 7](#_Toc49100633)

[AVERAGE RATING 7](#_Toc49100634)

[Get average rating of the service provider 7](#_Toc49100635)

[BOOKING 8](#_Toc49100636)

[Get booking by booking code 8](#_Toc49100637)

[Cancel Booking (This booking Only) 9](#_Toc49100638)

[Cancel Booking (For Future Bookings) 9](#_Toc49100639)

[Check Booking 10](#_Toc49100640)

[Check Schedule 10](#_Toc49100641)

[Confirm Booking (One-Time) 11](#_Toc49100642)

[Confirm Booking (Recurring) 12](#_Toc49100643)

[Approve/Reject Booking 13](#_Toc49100644)

[Approve/Reject Booking (Recurring) 14](#_Toc49100645)

[Verify Booking 15](#_Toc49100646)

[Get bookings with issue 15](#_Toc49100647)

[Report booking with issue 16](#_Toc49100648)

[Get bookings with non-compliance 17](#_Toc49100649)

[Update booking with non-compliance 18](#_Toc49100650)

[Get bookings by service provider name and date 18](#_Toc49100651)

[Get bookings by service provider type and date 19](#_Toc49100652)

[Get booking by status 20](#_Toc49100653)

[Get Todays’ booking by booking unit 21](#_Toc49100654)

[Get current bookings by booking unit 22](#_Toc49100655)

[Get past & current bookings by booking unit 23](#_Toc49100656)

[Generate Booking Report 24](#_Toc49100657)

[FEEDBACK 26](#_Toc49100658)

[Provide Rating for Booking 26](#_Toc49100659)

[Provide Feedback to the Booking Unit (General/Non-Compliance) 26](#_Toc49100660)

[FERRY GUIDELINE 27](#_Toc49100661)

[Get List of ferry guidelines 27](#_Toc49100662)

[Add new Ferry Guideline 28](#_Toc49100663)

[Update Ferry Guideline 28](#_Toc49100664)

[NOTIFICATIONS 29](#_Toc49100665)

[Get List of notifications based on username for past 7 days 29](#_Toc49100666)

[PURPOSE 29](#_Toc49100667)

[Get List of Purposes 30](#_Toc49100668)

[Add new Purpose 30](#_Toc49100669)

[Update Purpose 31](#_Toc49100670)

[ROLE 31](#_Toc49100671)

[Get List of roles 31](#_Toc49100672)

[ROUTE 32](#_Toc49100673)

[Get List of routes 32](#_Toc49100674)

[Add new Route 33](#_Toc49100675)

[Update Route 34](#_Toc49100676)

[SCHEDULE 34](#_Toc49100677)

[Get List of schedules by service provider type and date 34](#_Toc49100678)

[Update Schedule 35](#_Toc49100679)

[Clear Schedule 36](#_Toc49100680)

[Block Schedule 36](#_Toc49100681)

[Add new Schedule (One-Time) 37](#_Toc49100682)

[Add new Schedule (Recurring) 38](#_Toc49100683)

[Get Schedule Template 38](#_Toc49100684)

[Set Schedule Template 39](#_Toc49100685)

[Populate Schedule 39](#_Toc49100686)

[SERVICE PROVIDER 40](#_Toc49100687)

[Get List of Service Providers 41](#_Toc49100688)

[Add new Service Provider 41](#_Toc49100689)

[Update Service Provider 42](#_Toc49100690)

[USER 42](#_Toc49100691)

[Get List of Users 42](#_Toc49100692)

[Add new User 44](#_Toc49100693)

[Update User 44](#_Toc49100694)

[Forget Password 45](#_Toc49100695)

[Confirm Forget Password 45](#_Toc49100696)

[Login 46](#_Toc49100697)

[Login with 2FA 47](#_Toc49100698)

[Complete Registration 48](#_Toc49100699)

[First Time Login (Complete Registration with Encryption) 48](#_Toc49100700)

[Change Password 49](#_Toc49100701)

[Verify OTP 49](#_Toc49100702)

[Send OTP 50](#_Toc49100703)

[Get Saf Unit 50](#_Toc49100704)

[Get Performance Rating 51](#_Toc49100705)

[VEHICLE 51](#_Toc49100706)

[Get List of Vehicles 52](#_Toc49100707)

[Add new Vehicle 52](#_Toc49100708)

[Update Vehicle 53](#_Toc49100709)

# Functions in Backend (AWS API Gateway)

The URL to AWS API Gateway:

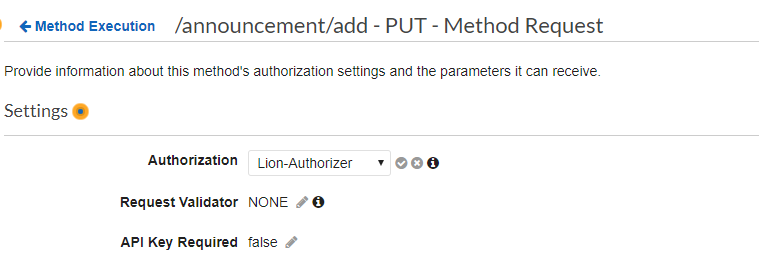
<https://9szv1s3oz6.execute-api.ap-southeast-1.amazonaws.com/LionDev>

**Note:**

1. **Add Authorization to API Gateway**

For deployment for production, please ensure that the method request authorizer is selected to “Lion-Authorizer”. It will verify if it exists in the AWS Cognito User Pool.

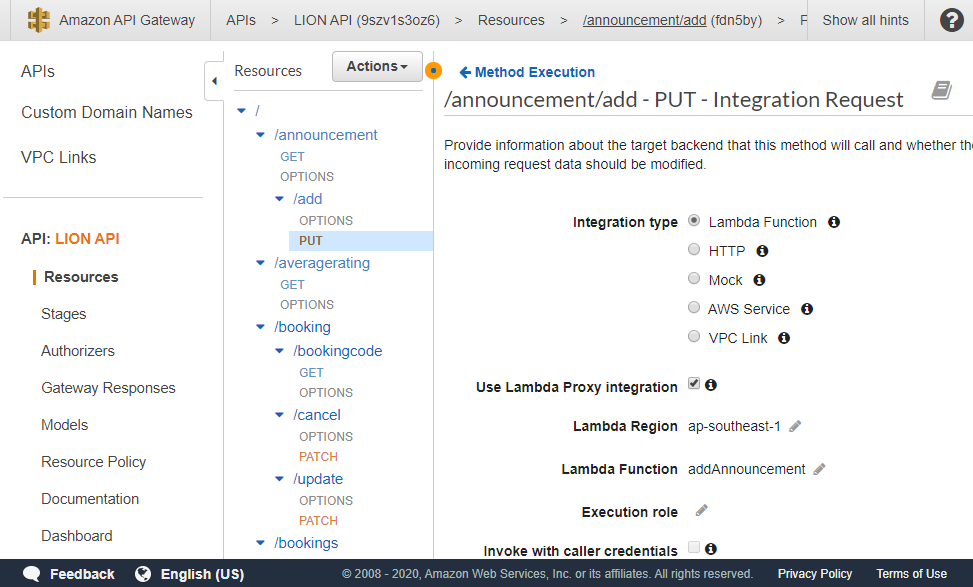
E.g. Announcement -> Method Request - > Authorization



* Select “Lion-Authorizer”
* Actions -> Deploy API

1. **Testing at AWS API Gateway**

For testing at AWS API Gateway, please ensure that the “Use Lambda Proxy Integration” is enabled. You can find it at the Integration Request.

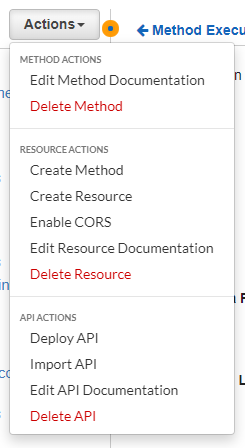


* 1. **Enable CORS**

After testing the Rest API at AWS API Gateway, please ensure that the “Use Lambda Proxy Integration” is enabled or disabled based on the method. You can find it at the Integration Request.

|  |  |
| --- | --- |
| GET Method | Enable “Use Lambda Proxy Integration “ |
| PUT or PATCH Method | Disable “Use Lambda Proxy Integration” |

After you have enabled or disabled the Proxy Integration, for that API gateway, please ensure you go to **Actions -> Enable CORS -> Enable CORS and replace existing CORS headers**



After you have enabled CORS, please click on **Actions -> Deploy API** to redeploy the API.

## ANNOUNCEMENT

|  |  |  |  |
| --- | --- | --- | --- |
| Get List of Announcements | | | |
| **Method** | /announcement | **Type** | GET |
| **Parameters** | role | role=all, u, pu, sp, a | |
| **Content Type** | application/json |  |  |
| **Headers** | {  Authorization: <token>  } | | |
| **Response Body** | {  "statusCode": 200,  "header": {  "Content-Type": "application/json",  "Access-Control-Allow-Origin": "\*"  },  "body": [  {  "announcementId": 3,  "message": "Gate 2 is closed.",  "targetAudience": "DP",  "createdBy": "system",  "createdDate": "2020-04-17T13:00:08.000Z"  }  ]  } | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Add new Announcement | | | |
| **Method** | /announcement/add | **Type** | PUT |
| **Parameters** | - | **Content Type** | application/json |
| **Headers** | {  Authorization: <token>  } | | |
| **Request Body (F**or testing at postman or AWS API Gateway) | {  "message": "Gate 5 is closed.",  "targetAudience": "DP",  "createdBy": "system"  } | | |
| **Response Body** | {  "statusCode": 200,  "header": {  "Content-Type": "application/json",  "Access-Control-Allow-Origin": "\*"  },  "body": {  "fieldCount": 0,  "affectedRows": 1,  "insertId": 17,  "serverStatus": 34,  "warningCount": 0,  "message": "&Records: 1 Duplicates: 0 Warnings: 0",  "protocol41": true,  "changedRows": 0  }  } | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Delete Announcement | | | |
| **Method** | /announcement/delete | **Type** | DELETE |
| **Parameters** | - | **Content Type** | application/json |
| **Headers** | {  Authorization: <token>  } | | |
| **Request Body (F**or testing at postman or AWS API Gateway) | {  "announcementId": "1"  } | | |
| **Response Body** | {  "statusCode": 200,  "body": {  "statusCode": 200,  "header": {  "Content-Type": "application/json",  "Access-Control-Allow-Origin": "\*"  },  "body": {  "fieldCount": 0,  "affectedRows": 1,  "insertId": 0,  "serverStatus": 2,  "warningCount": 0,  "message": "",  "protocol41": true,  "changedRows": 0  }  }  } | | |

## AVERAGE RATING

|  |  |  |  |
| --- | --- | --- | --- |
| Get average rating of the service provider | | | |
| **Method** | /averagerating | **Type** | GET |
| **Parameters** | - | **Content Type** | application/json |
| **Headers** | {  Authorization: <token>  } | | |
| **Response Body** | {  "statusCode": 200,  "header": {  "Content-Type": "application/json",  "Access-Control-Allow-Origin": "\*"  },  "body": [  {  "serviceProviderType": "P",  "avgRating": 3.25  },  {  "serviceProviderType": "RPL",  "avgRating": 4  }  ]  } | | |

## BOOKING

|  |  |  |  |
| --- | --- | --- | --- |
| Get booking by booking code | | | |
| **Method** | /booking/bookingcode | **Type** | GET |
| **Parameters** | status | E.g. status=pending, all, approved, cancelled | |
| **Content Type** | application/json | | |
| **Response Body** | {  "statusCode": 200,  "header": {  "Content-Type": "application/json",  "Access-Control-Allow-Origin": "\*"  },  "body": [  {  "bookingCode": "AS804D",  "departureDate": "2020-05-01T00:00:00.000Z",  "departureTime": "2230",  "bookingUnit": "blueFrog",  "displayUserName": "blueFrog",  "scheduleId": "51",  "purposeId": 3,  "purposeShort": "DE",  "routeId": 3,  "routeName": "SFT-TFT",  "advancedNotice": "at least 48 hours",  "serviceProviderType": "P",  "bookingType": "fixed",  "status": "Pending Review",  "remarks": null,  "numPassenger": 30,  "vehicles": null,  "cancellationReason": null,  "rejectedReason": null,  "onBoardTime": null,  "bookingGroup": null,  "createdBy": "system",  "createdDate": "2020-05-05T15:34:09.000Z",  "updatedBy": "tempaccount",  "updatedDate": "2020-05-08T17:53:57.000Z",  "reportIssue": "Issue with booking",  "findings": "test"  }]  } | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Cancel Booking (This booking Only) | | | |
| **Method** | /booking/cancel | **Type** | PATCH |
| **Content Type** | application/json | | |
| **Headers** | {  Authorization: <token>  } | | |
| **Request Body** | {  "bookingCode":"THTQ4Z ",  "bookingUnit":"blueFrog",  "cancellationReason":"Cancel Booking from 2020-06-11 onwards",  "bookingGroup": "",  "departureDate": "2020-06-11",  "updatedBy": "system"  } | | |
| **Response Body** | {  "statusCode": 200,  "header": {  "Content-Type": "application/json",  "Access-Control-Allow-Origin": "\*"  },  "body": {  "fieldCount": 0,  "affectedRows": 1,  "insertId": 17,  "serverStatus": 34,  "warningCount": 0,  "message": "&Records: 1 Duplicates: 0 Warnings: 0",  "protocol41": true,  "changedRows": 0  }  } | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Cancel Booking (For Future Bookings) | | | |
| **Method** | /booking/cancel | **Type** | PATCH |
| **Content Type** | application/json | | |
| **Headers** | {  Authorization: <token>  } | | |
| **Request Body** | {  "bookingCode":"",  "bookingUnit":"blueFrog",  "cancellationReason":"Cancel Booking from 2020-06-11 onwards",  "bookingGroup": "THTQ4Z",  "departureDate": "2020-06-11",  "updatedBy": "system"  } | | |
| **Response Body** | {  "statusCode": 200,  "header": {  "Content-Type": "application/json",  "Access-Control-Allow-Origin": "\*"  },  "body": {  "fieldCount": 0,  "affectedRows": 1,  "insertId": 17,  "serverStatus": 34,  "warningCount": 0,  "message": "&Records: 1 Duplicates: 0 Warnings: 0",  "protocol41": true,  "changedRows": 0  }  } | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Check Booking | | | |
| This API function is to check if the booking is advanced notice and the type of booking. E.g. OOS or Fixed or Adhoc | | | |
| **Method** | /booking/checkbooking | **Type** | GET |
| **Parameters** | serviceProviderType=P&departureDate=YYYY-MM-dd&departureTime=HHmm | serviceProviderType=P&departureDate=2020-05-30&departureTime=0900 | |
| **Headers** | {  Authorization: <token>  } | | |
| **Content Type** | application/json | | |
| **Response Body** | {  "statusCode": 200,  "header": {  "Content-Type": "application/json",  "Access-Control-Allow-Origin": "\*"  },  "body": {  "advancedNotice": "at least 48 hours",  "bookingType": "OOS"  }  } | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Check Schedule | | | |
| This API function is to get the list of schedules and its availability based on the booking date. | | | |
| **Method** | /booking/checkschedule | **Type** | GET |
| **Parameters** | serviceProviderType=P&departureDate=YYYY-MM-dd&routeId=3&capacity=10 | serviceProviderType=P&departureDate=2020-05-30&routeId=3&capacity=10 | |
| **Headers** | {  Authorization: <token>  } | | |
| **Content Type** | application/json | | |
| **Response Body** | {  "statusCode": 200,  "header": {  "Content-Type": "application/json",  "Access-Control-Allow-Origin": "\*"  },  "body": [  {  "scheduleId": "606",  "departureTime": "0830",  "maxLimit": 50,  "available": "true"  },  {  "scheduleId": "611",  "departureTime": "0830",  "maxLimit": 50,  "available": "true"  },  {  "scheduleId": "616",  "departureTime": "0930",  "maxLimit": 50,  "available": "true"  },  {  "scheduleId": "671",  "departureTime": "2130",  "maxLimit": 50,  "available": "true"  },  {  "scheduleId": "676",  "departureTime": "2230",  "maxLimit": 50,  "available": "true"  },  {  "scheduleId": "681",  "departureTime": "2330",  "maxLimit": 50,  "available": "true"  }  ]  } | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Confirm Booking (One-Time) | | | |
| You can call checkBooking first to determine the booking type and advanced notice before you call confirmBooking. Once you got the booking type and advanced notice, you will be able to call confirm booking. | | | |
| **Method** | /booking/confirmbooking | **Type** | PUT |
| **Parameters** | - | **Content Type** | application/json |
| **Headers** | {  Authorization: <token>  } | | |
| **Request Body (F**or testing at postman or AWS API Gateway)  **Passenger** | {  "roleId": "U",  "departureDate": "2020-05-30",  "departureTime": "0230",  "bookingUnit": "blueFrog",  "displayUserName": "blueFrog",  "scheduleId": "",  "purposeId": "3",  "routeId": 3,  "advancedNotice": "at least 48 hours",  "serviceProviderType": "P",  "bookingType": "OOS",  "status": "",  "remarks": "",  "numPassenger": "30",  "vehicles": "",  "cancellationReason": "",  "rejectedReason": null,  "onBoardTime": "",  "bookingGroup": "",  "createdBy": "tempAccount",  "performanceRating": "80"  } | | |
| **Request Body (F**or testing at postman or AWS API Gateway)  **RPL** | {  "roleId": "A",  "departureDate": "2020-05-30",  "departureTime": "0230",  "bookingUnit": "blueFrog",  "displayUserName": "ADMIN - blueFrog",  "scheduleId": "",  "purposeId": "3",  "routeId": 3,  "advancedNotice": "at least 48 hours",  "serviceProviderType": "RPL",  "bookingType": "OOS",  "status": "",  "remarks": "",  "numPassenger": "",  "vehicles": "1x2;2x1",  "cancellationReason": "",  "onBoardTime": "",  "bookingGroup": "",  "createdBy": "tempAccount",  "performanceRating": "80"  } | | |
| **Response Body** | {  "statusCode": 200,  "header": {  "Content-Type": "application/json",  "Access-Control-Allow-Origin": "\*"  },  "body": {  "status": "Approved"  }  } | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Confirm Booking (Recurring) | | | |
| You can call checkBooking first to determine the booking type and advanced notice before you call confirmBooking. Once you got the booking type and advanced notice, you will be able to call confirm booking.  This confirm booking method allows recurring booking. | | | |
| **Method** | /booking/confirmrecurringbooking | **Type** | PUT |
| **Parameters** | - | **Content Type** | application/json |
| **Headers** | {  Authorization: <token>  } | | |
| **Request Body (F**or testing at postman or AWS API Gateway)  **Passenger** | {  "roleId": "A",  "departureDate": "2020-06-19",  "departureTime": "0730",  "bookingUnit": "blueFrog",  "displayUserName": "blueFrog",  "scheduleId": "",  "purposeId": "3",  "routeId": "3",  "serviceProviderType": "P",  "remarks": "",  "totalLoad": "",  "vehicles": "",  "numPassenger": "50",  "createdBy": "system",  "performanceRating": "80",  "recurring": "2;3",  "endRepeatDate": "2020-06-24"  } | | |
| **Request Body (F**or testing at postman or AWS API Gateway)  **RPL** | {  "roleId": "A",  "departureDate": "2020-06-08",  "departureTime": "0730",  "bookingUnit": "blueFrog",  "displayUserName": "blueFrog",  "scheduleId": "",  "purposeId": "12",  "routeId": "1",  "serviceProviderType": "RPL",  "remarks": "SFV1234R, SPL9998E",  "totalLoad": "2",  "vehicles": "1x1;2x1",  "numPassenger": "",  "createdBy": "system",  "performanceRating": "80",  "recurring": "2;3",  "endRepeatDate": "2020-06-19"  } | | |
| **Response Body** | {  "statusCode": 200,  "header": {  "Content-Type": "application/json",  "Access-Control-Allow-Origin": "\*"  },  "body": {  "fieldCount": 0,  "affectedRows": 1,  "insertId": 17,  "serverStatus": 34,  "warningCount": 0,  "message": "&Records: 1 Duplicates: 0 Warnings: 0",  "protocol41": true,  "changedRows": 0  }  } | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Approve/Reject Booking | | | |
| This function is used when user approve or reject a booking. | | | |
| **Method** | /booking/update | **Type** | PATCH |
| **Content Type** | application/json | | |
| **Headers** | {  Authorization: <token>  } | | |
| **Request Body** | {  "bookingCode": "MI2OP9",  "departureDate": "2020-07-11",  "departureTime": "1000",  "bookingUnit": "redleopard",  "status": "Rejected",  "rejectedReason":"Performance Rating fall below benchmark",  "updatedBy": "bluefrog"  } | | |
| **Response Body** | {  "statusCode": 200,  "header": {  "Content-Type": "application/json",  "Access-Control-Allow-Origin": "\*"  },  "body": {  "fieldCount": 0,  "affectedRows": 1,  "insertId": 17,  "serverStatus": 34,  "warningCount": 0,  "message": "&Records: 1 Duplicates: 0 Warnings: 0",  "protocol41": true,  "changedRows": 0  }  } | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Approve/Reject Booking (Recurring) | | | |
| This function is used when user approve or reject recurring booking. | | | |
| **Method** | /booking/update | **Type** | PATCH |
| **Content Type** | application/json | | |
| **Headers** | {  Authorization: <token>  } | | |
| **Request Body** | {  "bookingCode": "MI2OP9",  "departureDate": "2020-07-11",  "departureTime": "1000",  "bookingUnit": "redleopard",  "status": "Rejected",  "rejectedReason":"Performance Rating fall below benchmark",  "updatedBy": "bluefrog",  "bookingGroup”: "MI2OP9"  } | | |
| **Response Body** | {  "statusCode": 200,  "header": {  "Content-Type": "application/json",  "Access-Control-Allow-Origin": "\*"  },  "body": {  "fieldCount": 0,  "affectedRows": 1,  "insertId": 17,  "serverStatus": 34,  "warningCount": 0,  "message": "&Records: 1 Duplicates: 0 Warnings: 0",  "protocol41": true,  "changedRows": 0  }  } | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Verify Booking | | | |
| This function is used by service provider to verify if the booking is valid, if valid, it will record the boarding time. | | | |
| **Method** | /booking/verifybooking | **Type** | PATCH |
| **Content Type** | application/json | | |
| **Headers** | {  Authorization: <token>  } | | |
| **Request Body** | {  "bookingCode": “XZS6EE",  “updatedBy”: “tempAccount”  } | | |
| **Response Body** | {  "statusCode": 200,  "header": {  "Content-Type": "application/json",  "Access-Control-Allow-Origin": "\*"  },  "body": {  "message": "Boarding time recorded. It is marked as late. Your performance rating is deducted by 20%."  }  } | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Get bookings with issue | | | |
| **Method** | /booking/issue | | |
| **Type** | GET | **Content Type** | application/json |
| **Headers** | {  Authorization: <token>  } | | |
| **Response Body** | {  "statusCode": 200,  "header": {  "Content-Type": "application/json",  "Access-Control-Allow-Origin": "\*"  },  "body": [  {  "bookingCode": "AS804D",  "departureDate": "2020-05-01",  "departureTime": "2230",  "bookingUnit": "blueFrog",  "displayUserName": "blueFrog",  "scheduleId": "51",  "purposeId": 3,  "purposeShort": "DE",  "routeId": 3,  "routeName": "SFT-TFT",  "advancedNotice": "at least 48 hours",  "serviceProviderType": "P",  "bookingType": "fixed",  "bookingStatus": "pending",  "remarks": null,  "numPassenger": 30,  "cancellationReason": null,  "rejectedReason": null,  "onBoardTime": null,  "bookingGroup": null,  "createdBy": "system",  "createdDate": "2020-04-30T15:21:30.000Z",  "updatedBy": "system",  "updatedDate": "2020-04-30T15:21:30.000Z",  "safFeedbackId": 1,  "feedbackType": null,  "rating": 4,  "comments": null,  "spFeedbackId": null,  "SPFStatus": null,  "findings": null,  "reportIssue": "Issue with booking",  "bookingIssueStatus": "Pending Review",  "bookingIssueFindings": "test"  }  ]  } | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Report booking with issue | | | |
| **Method** | /bookingissue/add | **Type** | PUT |
| **Content Type** | application/json | | |
| **Headers** | {  Authorization: <token>  } | | |
| **Request Body** | {  "bookingCode": “XRT34A",  "reportIssue": "Bad Attitude by the guard",  “createdBy”: “bluefrog”  } | | |
| **Response Body** | {  "statusCode": 200,  "header": {  "Content-Type": "application/json",  "Access-Control-Allow-Origin": "\*"  },  "body": {  "fieldCount": 0,  "affectedRows": 1,  "insertId": 17,  "serverStatus": 34,  "warningCount": 0,  "message": "&Records: 1 Duplicates: 0 Warnings: 0",  "protocol41": true,  "changedRows": 0  }  } | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Get bookings with non-compliance | | | |
| **Method** | /bookings/noncompliance | | |
| **Type** | GET | **Content Type** | application/json |
| **Headers** | {  Authorization: <token>  } | | |
| **Response Body** | {  "statusCode": 200,  "header": {  "Content-Type": "application/json",  "Access-Control-Allow-Origin": "\*"  },  "body": [  {  "bookingCode": "BE788W",  "departureDate": "2020-05-01",  "departureTime": "2230",  "bookingUnit": "blackpanther",  "displayUserName": "blackpanther",  "scheduleId": "51",  "purposeId": 3,  "purposeShort": "DE",  "routeId": 3,  "routeName": "SFT-TFT",  "advancedNotice": "at least 48 hours",  "serviceProviderType": "P",  "bookingType": "fixed",  "bookingStatus": "pending",  "remarks": null,  "numPassenger": 20,  "cancellationReason": null,  "rejectedReason": null,  "onBoardTime": null,  "bookingGroup": null,  "createdBy": "system",  "createdDate": "2020-04-30T15:21:30.000Z",  "updatedBy": "system",  "updatedDate": "2020-04-30T15:21:30.000Z",  "safFeedbackId": 4,  "feedbackType": "Non-Compliance",  "rating": 0,  "comments": "Rowdy Behaviour",  "spFeedbackId": 2,  "SPFStatus": "Investigating",  "findings": "Update finding",  "reportIssue": null,  "bookingIssueStatus": null,  "bookingIssueFindings": null,  "Vehicle": [  {  "vehicleId": 1,  "name": "Lorry (12 FT)"  }  ]  }  ]  } | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Update booking with non-compliance | | | |
| **Method** | /bookings/noncompliance/update | **Type** | PATCH |
| **Content Type** | application/json | | |
| **Headers** | {  Authorization: <token>  } | | |
| **Request Body** | {  "bookingCode": “XRT34A",  "feedbackType": "Non-Compliance",  “comments”: “Rowdy Behaviour”,  "status": “Completed",  “findings”: “Case Closed.”  “updatedBy”: “system”  } | | |
| **Response Body** | {  "statusCode": 200,  "header": {  "Content-Type": "application/json",  "Access-Control-Allow-Origin": "\*"  },  "body": {  "fieldCount": 0,  "affectedRows": 1,  "insertId": 17,  "serverStatus": 34,  "warningCount": 0,  "message": "&Records: 1 Duplicates: 0 Warnings: 0",  "protocol41": true,  "changedRows": 0  }  } | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Get bookings by service provider name and date | | | |
| **Method** | /bookings/serviceprovidername/startdateenddate | | |
| **Parameter** | ?serviceProviderType=P&sp=<SERVICEPROVIDERNAME>&startDate=YYYY-MM-dd&endDate=YYYY-MM-dd | | |
| **E.g.** | /bookings/ serviceprovidername/startdateenddate?serviceProviderType=RPL&sp=ENG-HUP%20SHIPPING&startDate=2020-06-01&endDate=2020-06-01 | | |
| **Type** | GET | **Content Type** | application/json |
| **Headers** | {  Authorization: <token>  } | | |
| **Response Body** | {  "statusCode": 200,  "header": {  "Content-Type": "application/json",  "Access-Control-Allow-Origin": "\*"  },  "body": [  {  "bookingCode": "BE788W",  "departureDate": "2020-05-01",  "departureTime": "2230",  "bookingUnit": "blackpanther",  "displayUserName": "blackpanther",  "scheduleId": "51",  "purposeId": 3,  "purposeShort": "DE",  "routeId": 3,  "routeName": "SFT-TFT",  "advancedNotice": "at least 48 hours",  "serviceProviderType": "P",  "bookingType": "fixed",  "bookingStatus": "pending",  "remarks": null,  "numPassenger": 20,  "cancellationReason": null,  "rejectedReason": null,  "onBoardTime": null,  "bookingGroup": null,  "createdBy": "system",  "createdDate": "2020-04-30T15:21:30.000Z",  "updatedBy": "system",  "updatedDate": "2020-04-30T15:21:30.000Z",  "safFeedbackId": 4,  "feedbackType": "Non-Compliance",  "rating": 0,  "comments": "Rowdy Behaviour",  "spFeedbackId": 2,  "SPFStatus": "Investigating",  "findings": "Update finding",  "reportIssue": null,  "bookingIssueStatus": null,  "bookingIssueFindings": null,  "Vehicle": [  {  "vehicleId": 1,  "name": "Lorry (12 FT)"  }  ]  }  ]  } | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Get bookings by service provider type and date | | | |
| **Method** | /bookings/startdateenddate | | |
| **Parameter** | ?serviceProviderType=all&startDate=YYYY-MM-DD&endDate=YYYY-MM-DD | | |
| **Type** | GET | **Content Type** | application/json |
| **Headers** | {  Authorization: <token>  } | | |
| **Response Body** | {  "statusCode": 200,  "header": {  "Content-Type": "application/json",  "Access-Control-Allow-Origin": "\*"  },  "body": [  {  "bookingCode": "BE788W",  "departureDate": "2020-05-01",  "departureTime": "2230",  "bookingUnit": "blackpanther",  "displayUserName": "blackpanther",  "scheduleId": "51",  "purposeId": 3,  "purposeShort": "DE",  "routeId": 3,  "routeName": "SFT-TFT",  "advancedNotice": "at least 48 hours",  "serviceProviderType": "P",  "bookingType": "fixed",  "bookingStatus": "pending",  "remarks": null,  "numPassenger": 20,  "cancellationReason": null,  "rejectedReason": null,  "onBoardTime": null,  "bookingGroup": null,  "createdBy": "system",  "createdDate": "2020-04-30T15:21:30.000Z",  "updatedBy": "system",  "updatedDate": "2020-04-30T15:21:30.000Z",  "safFeedbackId": 4,  "feedbackType": "Non-Compliance",  "rating": 0,  "comments": "Rowdy Behaviour",  "spFeedbackId": 2,  "SPFStatus": "Investigating",  "findings": "Update finding",  "reportIssue": null,  "bookingIssueStatus": null,  "bookingIssueFindings": null,  "Vehicle": [  {  "vehicleId": 1,  "name": "Lorry (12 FT)"  }  ]  }  ]  } | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Get booking by status | | | |
| **Method** | /bookings/status | | |
| **Parameters** | status | E.g. status=pending, all, approved, cancelled | |
| **Type** | GET | **Content Type** | application/json |
| **Response Body** | {  "statusCode": 200,  "header": {  "Content-Type": "application/json",  "Access-Control-Allow-Origin": "\*"  },  "body": [  {  "bookingCode": "BE788W",  "departureDate": "2020-05-01",  "departureTime": "2230",  "bookingUnit": "blackpanther",  "displayUserName": "blackpanther",  "scheduleId": "51",  "purposeId": 3,  "purposeShort": "DE",  "routeId": 3,  "routeName": "SFT-TFT",  "advancedNotice": "at least 48 hours",  "serviceProviderType": "P",  "bookingType": "fixed",  "bookingStatus": "pending",  "remarks": null,  "numPassenger": 20,  "cancellationReason": null,  "rejectedReason": null,  "onBoardTime": null,  "bookingGroup": null,  "createdBy": "system",  "createdDate": "2020-04-30T15:21:30.000Z",  "updatedBy": "system",  "updatedDate": "2020-04-30T15:21:30.000Z",  "safFeedbackId": 4,  "feedbackType": "Non-Compliance",  "rating": 0,  "comments": "Rowdy Behaviour",  "spFeedbackId": 2,  "SPFStatus": "Investigating",  "findings": "Update finding",  "reportIssue": null,  "bookingIssueStatus": null,  "bookingIssueFindings": null,  "Vehicle": [  {  "vehicleId": 1,  "name": "Lorry (12 FT)"  }  ]  }  ]  } | | |
| Get Todays’ booking by booking unit | | | |
| **Method** | /bookings/todaybookings | | |
| **Parameters** | status  bookingUnit | E.g. status=pending, all, approved, cancelled | |
| **Type** | GET | **Content Type** | application/json |
| **Response Body** | {  "statusCode": 200,  "header": {  "Content-Type": "application/json",  "Access-Control-Allow-Origin": "\*"  },  "body": [  {  "bookingCode": "BE788W",  "departureDate": "2020-05-01",  "departureTime": "2230",  "bookingUnit": "blackpanther",  "displayUserName": "blackpanther",  "scheduleId": "51",  "purposeId": 3,  "purposeShort": "DE",  "routeId": 3,  "routeName": "SFT-TFT",  "advancedNotice": "at least 48 hours",  "serviceProviderType": "P",  "bookingType": "fixed",  "bookingStatus": "pending",  "remarks": null,  "numPassenger": 20,  "cancellationReason": null,  "rejectedReason": null,  "onBoardTime": null,  "bookingGroup": null,  "createdBy": "system",  "createdDate": "2020-04-30T15:21:30.000Z",  "updatedBy": "system",  "updatedDate": "2020-04-30T15:21:30.000Z",  "safFeedbackId": 4,  "feedbackType": "Non-Compliance",  "rating": 0,  "comments": "Rowdy Behaviour",  "spFeedbackId": 2,  "SPFStatus": "Investigating",  "findings": "Update finding",  "reportIssue": null,  "bookingIssueStatus": null,  "bookingIssueFindings": null,  "Vehicle": [  {  "vehicleId": 1,  "name": "Lorry (12 FT)"  }  ]  }  ]  } | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Get current bookings by booking unit | | | |
| This function will retrieve the bookings from today to future based on booking Unit | | | |
| **Method** | /bookings/current | | |
| **Parameters** | bookingUnit | e.g. bookingUnit=blackpanther | |
| **Type** | GET | **Content Type** | application/json |
| **Response Body** | {  "statusCode": 200,  "header": {  "Content-Type": "application/json",  "Access-Control-Allow-Origin": "\*"  },  "body": [  {  "bookingCode": "BE788W",  "departureDate": "2020-05-01",  "departureTime": "2230",  "bookingUnit": "blackpanther",  "displayUserName": "blackpanther",  "scheduleId": "51",  "purposeId": 3,  "purposeShort": "DE",  "routeId": 3,  "routeName": "SFT-TFT",  "advancedNotice": "at least 48 hours",  "serviceProviderType": "P",  "bookingType": "fixed",  "bookingStatus": "pending",  "remarks": null,  "numPassenger": 20,  "cancellationReason": null,  "rejectedReason": null,  "onBoardTime": null,  "bookingGroup": null,  "createdBy": "system",  "createdDate": "2020-04-30T15:21:30.000Z",  "updatedBy": "system",  "updatedDate": "2020-04-30T15:21:30.000Z",  "safFeedbackId": 4,  "feedbackType": "Non-Compliance",  "rating": 0,  "comments": "Rowdy Behaviour",  "spFeedbackId": 2,  "SPFStatus": "Investigating",  "findings": "Update finding",  "reportIssue": null,  "bookingIssueStatus": null,  "bookingIssueFindings": null,  "Vehicle": [  {  "vehicleId": 1,  "name": "Lorry (12 FT)"  }  ]  }  ]  } | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Get past & current bookings by booking unit | | | |
| This function will retrieve the bookings from today to future and past 1 week bookings based on booking Unit | | | |
| **Method** | /bookings/pastcurrent | | |
| **Parameters** | bookingUnit | e.g. bookingUnit=blackpanther | |
| **Type** | GET | **Content Type** | application/json |
| **Response Body** | {  "statusCode": 200,  "header": {  "Content-Type": "application/json",  "Access-Control-Allow-Origin": "\*"  },  "body": [  {  "bookingCode": "BE788W",  "departureDate": "2020-05-01",  "departureTime": "2230",  "bookingUnit": "blackpanther",  "displayUserName": "blackpanther",  "scheduleId": "51",  "purposeId": 3,  "purposeShort": "DE",  "routeId": 3,  "routeName": "SFT-TFT",  "advancedNotice": "at least 48 hours",  "serviceProviderType": "P",  "bookingType": "fixed",  "bookingStatus": "pending",  "remarks": null,  "numPassenger": 20,  "cancellationReason": null,  "rejectedReason": null,  "onBoardTime": null,  "bookingGroup": null,  "createdBy": "system",  "createdDate": "2020-04-30T15:21:30.000Z",  "updatedBy": "system",  "updatedDate": "2020-04-30T15:21:30.000Z",  "safFeedbackId": 4,  "feedbackType": "Non-Compliance",  "rating": 0,  "comments": "Rowdy Behaviour",  "spFeedbackId": 2,  "SPFStatus": "Investigating",  "findings": "Update finding",  "reportIssue": null,  "bookingIssueStatus": null,  "bookingIssueFindings": null,  "Vehicle": [  {  "vehicleId": 1,  "name": "Lorry (12 FT)"  }  ]  }  ]  } | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Generate Booking Report | | | |
| This function will retrieve the bookings from the selected date and send the report to the service provider. | | | |
| **Method** | /bookings/generatereport | | |
| **Type** | GET | **Content Type** | application/json |
| **Request Body** | {  "serviceProviderType": "RPL",  "startDate": "2020-06-01",  "endDate": "2020-06-10",  "purposeId": "10",  "bookingType": "Ad-Hoc,OOS",  "email": "XXXXX@gmail.com"  } | | |
| **Response Body** | Response:  {  "statusCode": 200,  "headers": {  "Access-Control-Allow-Headers": "Authorization",  "Access-Control-Allow-Origin": "\*"  },  "body": "{\"statusCode\":200,\"header\":{\"Content-Type\":\"application/json\",\"Access-Control-Allow-Origin\":\"\*\"},\"body\":{\"message\":\"Email sent to the service provider's email.\"}}"  } | | |

## FEEDBACK

|  |  |  |  |
| --- | --- | --- | --- |
| Provide Rating for Booking | | | |
| This function is to provide rating for the booking. | | | |
| **Method** | /feedback/saffeedback/add | **Type** | PUT |
| **Parameters** | - | **Content Type** | application/json |
| **Headers** | {  Authorization: <token>  } | | |
| **Request Body (F**or testing at postman or AWS API Gateway) | {  "bookingCode": "XZS6EE",  "feedbackType": "General",  "rating": "4",  "comments": "Good attitude",  "createdBy": "system"  } | | |
| **Response Body** | {  "statusCode": 200,  "header": {  "Content-Type": "application/json",  "Access-Control-Allow-Origin": "\*"  },  "body": {  "fieldCount": 0,  "affectedRows": 1,  "insertId": 8,  "serverStatus": 2,  "warningCount": 0,  "message": "",  "protocol41": true,  "changedRows": 0  }  } | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Provide Feedback to the Booking Unit (General/Non-Compliance) | | | |
| This function is to provide feedback to the bookingUnit who board the ferry. Reported by Service Providers | | | |
| **Method** | /feedback/spfeedback/add | **Type** | PUT |
| **Parameters** | - | **Content Type** | application/json |
| **Headers** | {  Authorization: <token>  } | | |
| **Request Body (F**or testing at postman or AWS API Gateway) | {  "bookingCode": "XZS6EE",  "feedbackType": "Non-Compliance",  "comments": "Rowdy behaviour",  "findings": "",  "createdBy": "system"  } | | |
| **Response Body** | {  "statusCode": 200,  "header": {  "Content-Type": "application/json",  "Access-Control-Allow-Origin": "\*"  },  "body": {  "fieldCount": 0,  "affectedRows": 1,  "insertId": 8,  "serverStatus": 2,  "warningCount": 0,  "message": "",  "protocol41": true,  "changedRows": 0  }  } | | |

## FERRY GUIDELINE

|  |  |  |  |
| --- | --- | --- | --- |
| Get List of ferry guidelines | | | |
| **Method** | /ferryguideline | **Type** | GET |
| **Parameter** | status | E.g. status=all, A, I | |
| **Content Type** | application/json | | |
| **Headers** | {  Authorization: <token>  } | | |
| **Response Body** | [  {  "ferryGuidelineId": 2,  "serviceProviderType": "RPL",  "guidelines": "1. All vehicles are to enter the RPL through Gate 3 at SFT.`~2. Please note that RPL gate will only be opened 30 minutes prior to stated RPL timing.`~3. Any vehicles attempting to enter SFT earlier than the stated timings will be rejected and asked t",  "status": "A",  "createdBy": "system",  "createdDate": "2020-03-17T16:23:00.000Z",  "updatedBy": "tempaccount",  "updatedDate": "2020-04-30T11:58:26.000Z"  },  {  "ferryGuidelineId": 3,  "serviceProviderType": "P",  "guidelines": "1. Indented strength should be as accurate as possible. Over indenting of strength to a large degree will be followed up on as a waste of funds.`~2.Sea Transport Section is to be informed should the unit be unable to make it for their timing. Failure to do so could result in heavy penalty.`~3. NRU who comes in with norminal roll must approach the Ferry Service Provider to identify their unit as well as to verify the booking code.`~4. NRU without nominal roll will have to exchange their NRIC/11B ",  "status": "A",  "createdBy": "system",  "createdDate": "2020-04-17T11:31:09.000Z",  "updatedBy": "system",  "updatedDate": "2020-04-29T23:27:19.000Z"  }  ] | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Add new Ferry Guideline | | | |
| **Method** | /ferryguideline/add | **Type** | PUT |
| **Parameters** | - | **Content Type** | application/json |
| **Headers** | {  Authorization: <token>  } | | |
| **Request Body (F**or testing at postman or AWS API Gateway) | {  "serviceProviderType": "P",  "guidelines": "Ensure that there is 1-meter social distancing",  "createdBy": "system"  } | | |
| **Response Body** | {  "statusCode": 200,  "header": {  "Content-Type": "application/json",  "Access-Control-Allow-Origin": "\*"  },  "body": {  "fieldCount": 0,  "affectedRows": 1,  "insertId": 17,  "serverStatus": 34,  "warningCount": 0,  "message": "&Records: 1 Duplicates: 0 Warnings: 0",  "protocol41": true,  "changedRows": 0  }  } | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Update Ferry Guideline | | | |
| **Method** | /ferryguideline/update | **Type** | PATCH | |
| **Content Type** | application/json | | |
| **Headers** | {  Authorization: <token>  } | | |
| **Request Body** | {  "ferryGuidelineId": "1",  "serviceProviderType": "P",  "guidelines": "Ensure that there is 1-meter social distancing",  "status": "A",  "updatedBy": "system"  } | | |
| **Response Body** | {  "statusCode": 200,  "header": {  "Content-Type": "application/json",  "Access-Control-Allow-Origin": "\*"  },  "body": {  "fieldCount": 0,  "affectedRows": 1,  "insertId": 17,  "serverStatus": 34,  "warningCount": 0,  "message": "&Records: 1 Duplicates: 0 Warnings: 0",  "protocol41": true,  "changedRows": 0  }  } | | |

## NOTIFICATIONS

|  |  |  |  |
| --- | --- | --- | --- |
| Get List of notifications based on username for past 7 days | | | |
| **Method** | /notification | E.g. /notification?username=blackpanther | |
| **Parameters** | username | username=blackpanther | |
| **Type** | GET | **Content Type** | application/json |
| **Headers** | {  Authorization: <token>  } | | |
| **Response Body** | {  "statusCode": 200,  "header": {  "Content-Type": "application/json",  "Access-Control-Allow-Origin": "\*"  },  "body": [  {  "notificationId": 2,  "title": "Booking Approved: QHC0EG",  "message": "2020-07-26, 1200",  "bookingCode": "QHC0EG",  "username": "blackpanther",  "createdDate": "2020-07-02T14:03:52.000Z"  },  {  "notificationId": 3,  "title": "Booking Approved: LO11QQ",  "message": "2020-11-07, 0700",  "bookingCode": "LO11QQ",  "username": "blackpanther",  "createdDate": "2020-07-01T14:03:52.000Z"  }  ]  } | | |

## PURPOSE

|  |  |  |  |
| --- | --- | --- | --- |
| Get List of Purposes | | | |
| **Method** | /purpose | E.g. /purpose?status=I | |
| **Parameters** | status | status=all, A, I | |
| **Type** | GET | **Content Type** | application/json |
| **Headers** | {  Authorization: <token>  } | | |
| **Response Body** | {  "statusCode": 200,  "header": {  "Content-Type": "application/json",  "Access-Control-Allow-Origin": "\*"  },  "body": [  {  "purposeId": 2,  "purposeShort": "BIBO",  "purposeDesc": "Recruits Book in/out",  "serviceProviderType": "P",  "status": "A",  "createdBy": "system",  "createdDate": "2020-03-17T16:11:16.000Z",  "updatedBy": "system",  "updatedDate": "2020-03-17T16:11:16.000Z"  }  ]  } | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Add new Purpose | | | |
| **Method** | /purpose/add | E.g. /purpose/add | |
| **Type** | PUT | **Content Type** | application/json |
| **Headers** | {  Authorization: <token>  } | | |
| **Request Body** | {  "purposeId":"15",  "purposeShort": "BIBO",  "purposeDesc": "Recruits Book in/out",  "serviceProviderType": "P",  "status":"A",  "updatedBy": "system"  } | | |
| **Response Body** | {  "statusCode": 200,  "header": {  "Content-Type": "application/json",  "Access-Control-Allow-Origin": "\*"  },  "body": {  "fieldCount": 0,  "affectedRows": 1,  "insertId": 17,  "serverStatus": 34,  "warningCount": 0,  "message": "&Records: 1 Duplicates: 0 Warnings: 0",  "protocol41": true,  "changedRows": 0  }  } | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Update Purpose | | | |
| **Method** | /purpose/update | E.g. /purpose/update | |
| **Type** | PATCH | **Content Type** | application/json |
| **Headers** | {  Authorization: <token>  } | | |
| **Request Body** | {  "purposeId":"15",  "purposeShort": "BIBO",  "purposeDesc": "Recruits Book in/out",  "serviceProviderType": "P",  "status":"A",  "updatedBy": "system"  } | | |
| **Response Body** | {  "statusCode": 200,  "header": {  "Content-Type": "application/json",  "Access-Control-Allow-Origin": "\*"  },  "body": {  "fieldCount": 0,  "affectedRows": 1,  "insertId": 0,  "serverStatus": 34,  "warningCount": 0,  "message": "(Rows matched: 1 Changed: 1 Warnings: 0",  "protocol41": true,  "changedRows": 1  }  } | | |

## ROLE

|  |  |  |  |
| --- | --- | --- | --- |
| Get List of roles | | | |
| **Method** | /role | | |
| **Type** | GET | **Content Type** | application/json |
| **Headers** | {  Authorization: <token>  } | | |
| **Response Body** | {  "statusCode": 200,  "header": {  "Content-Type": "application/json",  "Access-Control-Allow-Origin": "\*"  },  "body": [  {  "roleId": "A",  "roleName": "Administrator"  },  {  "roleId": "DP",  "roleName": "Duty Personnel"  },  {  "roleId": "PU",  "roleName": "Privilege User"  },  {  "roleId": "SP",  "roleName": "Service Provider"  },  {  "roleId": "U",  "roleName": "User"  }  ]  } | | |

## ROUTE

|  |  |  |  |
| --- | --- | --- | --- |
| Get List of routes | | | |
| **Method** | /route | E.g. /route/status=A&serviceprovidertype=all | |
| **Parameters** | status serviceprovidertype | status=all, A, I serviceprovidertype = all, P, RPL | |
| **Type** | GET | **Content Type** | application/json |
| **Headers** | {  Authorization: <token>  } | | |
| **Response Body** | {  "statusCode": 200,  "header": {  "Content-Type": "application/json",  "Access-Control-Allow-Origin": "\*"  },  "body": [  {  "routeId": 2,  "routeName": "LDP-MLR",  "from": "Ladang Ramp",  "destination": "Mainland Ramp",  "serviceProviderType": "RPL",  "status": "A",  "createdBy": "system",  "createdDate": "2020-03-17T16:17:07.000Z",  "updatedBy": "system",  "updatedDate": "2020-04-30T10:35:15.000Z"  },  {  "routeId": 1,  "routeName": "MLR-LDP",  "from": "Mainland Ramp",  "destination": "Ladang Ramp",  "serviceProviderType": "RPL",  "status": "A",  "createdBy": "system",  "createdDate": "2020-03-17T16:16:26.000Z",  "updatedBy": "system",  "updatedDate": "2020-04-30T10:39:36.000Z"  },  {  "routeId": 3,  "routeName": "SFT-TFT",  "from": "SAF Ferry Terminal",  "destination": "Tekong Ferry Terminal",  "serviceProviderType": "P",  "status": "A",  "createdBy": "system",  "createdDate": "2020-03-17T16:19:00.000Z",  "updatedBy": "system",  "updatedDate": "2020-03-17T16:19:00.000Z"  },  {  "routeId": 4,  "routeName": "TFT-SFT",  "from": "Tekong Ferry Terminal",  "destination": "SAF Ferry Terminal",  "serviceProviderType": "P",  "status": "A",  "createdBy": "system",  "createdDate": "2020-03-17T16:19:00.000Z",  "updatedBy": "system",  "updatedDate": "2020-04-30T10:39:45.000Z"  }  ]  } | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Add new Route | | | |
| **Method** | /route/add | E.g. /route/add | |
| **Type** | PUT | **Content Type** | application/json |
| **Headers** | {  Authorization: <token>  } | | |
| **Request Body** | {  "routeName":"NFT-SFT",  "from": "North Ferry Terminal",  "destination": "SAF Ferry Terminal",  "serviceProviderType": "P",  "status":"A",  "createdBy": "system"  } | | |
| **Response Body** | {  "statusCode": 200,  "header": {  "Content-Type": "application/json",  "Access-Control-Allow-Origin": "\*"  },  "body": {  "fieldCount": 0,  "affectedRows": 1,  "insertId": 17,  "serverStatus": 34,  "warningCount": 0,  "message": "&Records: 1 Duplicates: 0 Warnings: 0",  "protocol41": true,  "changedRows": 0  }  } | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Update Route | | | |
| **Method** | /route/update | E.g. /route/update | |
| **Type** | PATCH | **Content Type** | application/json |
| **Headers** | {  Authorization: <token>  } | | |
| **Request Body** | {  "routeId":"2",  "routeName":"NFT-SFT",  "from": "North Ferry Terminal",  "destination": "SAF Ferry Terminal",  "serviceProviderType": "P",  "status":"A",  "updatedBy": "system"  } | | |
| **Response Body** | {  "statusCode": 200,  "header": {  "Content-Type": "application/json",  "Access-Control-Allow-Origin": "\*"  },  "body": {  "fieldCount": 0,  "affectedRows": 1,  "insertId": 0,  "serverStatus": 34,  "warningCount": 0,  "message": "(Rows matched: 1 Changed: 1 Warnings: 0",  "protocol41": true,  "changedRows": 1  }  } | | |

## SCHEDULE

|  |  |  |  |
| --- | --- | --- | --- |
| Get List of schedules by service provider type and date | | | |
| **Method** | /schedule | E.g. /schedule/serviceProviderType=all&startDate=YYYY-MM-DD&endDate=YYYY-MM-DD | |
| **Parameters** | serviceProviderType  startDate  endDate | serviceProviderType = all, P, RPL  YYYY-MM-DD YYYY-MM-DD | |
| **Type** | GET | **Content Type** | application/json |
| **Headers** | {  Authorization: <token>  } | | |
| **Response Body** | {  "statusCode": 200,  "header": {  "Content-Type": "application/json",  "Access-Control-Allow-Origin": "\*"  },  "body": [  {  "scheduleId": 86,  "departureDate": "2020-05-05",  "departureTime": "0830",  "maxLimit": 50,  "serviceProviderType": "P",  "serviceProviderName": "ENG-HUP SHIPPING",  "routeId": 3,  "routeName": "SFT-TFT",  "status": "A",  "createdBy": "system",  "createdDate": "2020-05-04T17:09:36.000Z",  "updatedBy": "system",  "updatedDate": "2020-05-04T17:09:36.000Z"  }  ]  } | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Update Schedule | | | |
| **Method** | /schedule/update | **Type** | POST | |
| **Content Type** | application/json | | |
| **Headers** | {  Authorization: <token>  } | | |
| **Request Body** | {  "scheduleId" : "205",  "departureDate": "2020-05-25",  "departureTime": "1100",  "maxLimit": "5",  "serviceProviderName": "ABC SHIPPING",  "routeId": "1",  "status": "A",  "serviceProviderType": "RPL",  "endRepeatDate": "",  "recurring": "",  "updatedBy": "system"  } | | |
| **Response Body** | {  "statusCode": 200,  "header": {  "Content-Type": "application/json",  "Access-Control-Allow-Origin": "\*"  },  "body": {  "fieldCount": 0,  "affectedRows": 1,  "insertId": 17,  "serverStatus": 34,  "warningCount": 0,  "message": "&Records: 1 Duplicates: 0 Warnings: 0",  "protocol41": true,  "changedRows": 0  }  } | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Clear Schedule | | | |
| The function allows the user to clear the schedule from a specified date range and cancel any bookings that are within this date range. | | | |
| **Method** | /schedule/clear | **Type** | DELETE |
| **Content Type** | application/json | | |
| **Headers** | {  Authorization: <token>  } | | |
| **Request Body** | {  "serviceProviderType": "P",  "startDate": "2020-12-02",  "endDate": "2020-12-04",  "cancellationReason" : "Ferry Service not available",  "updatedBy": "system"  } | | |
| **Response Body** | {  "statusCode": 200,  "header": {  "Content-Type": "application/json",  "Access-Control-Allow-Origin": "\*"  },  "body": {  "fieldCount": 0,  "affectedRows": 1,  "insertId": 17,  "serverStatus": 34,  "warningCount": 0,  "message": "&Records: 1 Duplicates: 0 Warnings: 0",  "protocol41": true,  "changedRows": 0  }  } | | |
| Block Schedule | | | |
| The function allows the user to block the schedule from a specified date range and cancel any bookings that are within this date range. | | | |
| **Method** | /schedule/block | **Type** | PATCH |
| **Content Type** | application/json | | |
| **Headers** | {  Authorization: <token>  } | | |
| **Request Body** | {  "serviceProviderType": "P",  "startDate": "2020-12-02",  "endDate": "2020-12-04",  "cancellationReason" : "Ferry Service not available",  "updatedBy": "system"  } | | |
| **Response Body** | {  "statusCode": 200,  "header": {  "Content-Type": "application/json",  "Access-Control-Allow-Origin": "\*"  },  "body": {  "fieldCount": 0,  "affectedRows": 1,  "insertId": 17,  "serverStatus": 34,  "warningCount": 0,  "message": "&Records: 1 Duplicates: 0 Warnings: 0",  "protocol41": true,  "changedRows": 0  }  } | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Add new Schedule (One-Time) | | | |
| **Method** | /schedule/add | E.g. /schedule /add | |
| **Type** | PUT | **Content Type** | application/json |
| **Headers** | {  Authorization: <token>  } | | |
| **Request Body** | {  "departureDate": "2020-05-18",  "departureTime": "0830",  "maxLimit": "50",  "serviceProviderName": "ENG-HUP SHIPPING",  "routeId": "3",  "status": "A",  "serviceProviderType": "P",  "endRepeatDate": "",  "recurring": "",  "createdBy": "system"  } | | |
| **Response Body** | {  "statusCode": 200,  "header": {  "Content-Type": "application/json",  "Access-Control-Allow-Origin": "\*"  },  "body": {  "fieldCount": 0,  "affectedRows": 1,  "insertId": 602,  "serverStatus": 2,  "warningCount": 0,  "message": "",  "protocol41": true,  "changedRows": 0  }  } | | |
| Add new Schedule (Recurring) | | | |
| **Method** | /schedule/add | E.g. /schedule /add | |
| **Type** | PUT | **Content Type** | application/json |
| **Headers** | {  Authorization: <token>  } | | |
| **Request Body** | {  "departureDate": "2020-05-25",  "departureTime": "0830",  "maxLimit": "50",  "serviceProviderName": "ENG-HUP SHIPPING",  "routeId": "3",  "status": "A",  "serviceProviderType": "P",  "endRepeatDate": "2020-05-29",  "recurring": "1;2;3;4;5",  "createdBy": "system"  } | | |
| **Response Body** | {  "statusCode": 200,  "header": {  "Content-Type": "application/json",  "Access-Control-Allow-Origin": "\*"  },  "body": [  {  "selected\_date": "2020-05-25",  "dayOfWeek": 1  },  {  "selected\_date": "2020-05-26",  "dayOfWeek": 2  },  {  "selected\_date": "2020-05-27",  "dayOfWeek": 3  },  {  "selected\_date": "2020-05-28",  "dayOfWeek": 4  },  {  "selected\_date": "2020-05-29",  "dayOfWeek": 5  }  ]  } | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Get Schedule Template | | | |
| **Method** | /scheduletemplate/get | E.g. /scheduletemplate/set | |
| **Type** | GET | **Content Type** | application/json |
| **Headers** | {  Authorization: <token>  } | | |
| **Response Body** | {  "statusCode": 200,  "header": {  "Content-Type": "application/json",  "Access-Control-Allow-Origin": "\*"  },  "body": {  "fieldCount": 0,  "affectedRows": 12,  "insertId": 39,  "serverStatus": 2,  "warningCount": 0,  "message": "'Records: 12 Duplicates: 0 Warnings: 0",  "protocol41": true,  "changedRows": 0  }  } | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Set Schedule Template | | | |
| **Method** | /scheduletemplate/set | E.g. /scheduletemplate/set | |
| **Type** | PUT | **Content Type** | application/json |
| **Headers** | {  Authorization: <token>  } | | |
| **Request Body** | {  "serviceProviderType": "P",  "routeId": "3",  "schedule": "0830x1;2;3;4;5;6,0930x1;2;3;4;5,1030x1;2;3;4;5",  } | | |
| **Response Body** | {  "statusCode": 200,  "header": {  "Content-Type": "application/json",  "Access-Control-Allow-Origin": "\*"  },  "body": {  "fieldCount": 0,  "affectedRows": 12,  "insertId": 39,  "serverStatus": 2,  "warningCount": 0,  "message": "'Records: 12 Duplicates: 0 Warnings: 0",  "protocol41": true,  "changedRows": 0  }  } | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Populate Schedule | | | |
| This method will populate the next three months of schedule based on the schedule template in the database. | | | |
| **Method** | /schedule/populate | E.g. /schedule/populate | |
| **Type** | PUT | **Content Type** | application/json |
| **Headers** | {  Authorization: <token>  } | | |
| **Request Body** | {  "startDate": "2020-11-15",  "serviceProviderType": "RPL",  "maxLimit": "15",  "createdBy": "system"  } | | |
| **Response Body** | {  "statusCode": 200,  "header": {  "Content-Type": "application/json",  "Access-Control-Allow-Origin": "\*"  },  "body": [  [  "2020-08-17",  "0730",  "15",  "TIAN-SAN SHIPPING",  1,  "A",  "system",  "system"  ],  [  "2020-08-17",  "0930",  "15",  "TIAN-SAN SHIPPING",  1,  "A",  "system",  "system"  ],  [  "2020-08-17",  "0830",  "15",  "TIAN-SAN SHIPPING",  2,  "A",  "system",  "system"  ],  [  "2020-08-17",  "1230",  "15",  "TIAN-SAN SHIPPING",  2,  "A",  "system",  "system"  ]  ]  } | | |

## SERVICE PROVIDER

|  |  |  |  |
| --- | --- | --- | --- |
| Get List of Service Providers | | | |
| **Method** | /serviceprovider | **Type** | GET |
| **Parameters** | - | **Content Type** | application/json |
| **Headers** | {  Authorization: <token>  } | | |
| **Response Body** | {  "statusCode": 200,  "header": {  "Content-Type": "application/json",  "Access-Control-Allow-Origin": "\*"  },  "body": [  {  "spId": 1,  "name": "ENG-HUP SHIPPING",  "startDate": "2020-01-01T00:00:00.000Z",  "endDate": "2025-12-31T00:00:00.000Z",  "serviceProviderType": "P",  "createdBy": "system",  "createdDate": "2020-01-01T12:00:00.000Z",  "updatedBy": "system",  "updatedDate": "2020-01-01T12:00:00.000Z"  },  {  "spId": 2,  "name": "TIAN-SAN SHIPPING",  "startDate": "2020-01-01T00:00:00.000Z",  "endDate": "2025-12-31T00:00:00.000Z",  "serviceProviderType": "RPL",  "createdBy": "system",  "createdDate": "2020-01-01T12:00:00.000Z",  "updatedBy": "system",  "updatedDate": "2020-01-01T12:00:00.000Z"  }  ]  } | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Add new Service Provider | | | |
| **Method** | /serviceprovider/add | E.g. /serviceprovider/add | |
| **Type** | PUT | **Content Type** | application/json |
| **Headers** | {  Authorization: <token>  } | | |
| **Request Body** | {  "name":"ENG ENG SHIPPING",  "startDate": "2020-01-01",  "endDate": "2020-12-01",  "serviceProviderType": "P",  "createdBy": "system"  } | | |
| **Response Body** | {  "statusCode": 200,  "header": {  "Content-Type": "application/json",  "Access-Control-Allow-Origin": "\*"  },  "body": {  "fieldCount": 0,  "affectedRows": 1,  "insertId": 1,  "serverStatus": 34,  "warningCount": 0,  "message": "&Records: 1 Duplicates: 0 Warnings: 0",  "protocol41": true,  "changedRows": 0  }  } | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Update Service Provider | | | |
| **Method** | /serviceprovider/update | E.g. /serviceprovider/update | |
| **Type** | PATCH | **Content Type** | application/json |
| **Headers** | {  Authorization: <token>  } | | |
| **Request Body** | {  "spId":"1",  "name":"ENG ENG SHIPPING",  "startDate": "2020-01-01",  "endDate": "2020-12-01",  "serviceProviderType": "P",  "updatedBy": "system"  } | | |
| **Response Body** | {  "statusCode": 200,  "header": {  "Content-Type": "application/json",  "Access-Control-Allow-Origin": "\*"  },  "body": {  "fieldCount": 0,  "affectedRows": 1,  "insertId": 0,  "serverStatus": 34,  "warningCount": 0,  "message": "(Rows matched: 1 Changed: 1 Warnings: 0",  "protocol41": true,  "changedRows": 1  }  } | | |

## USER

|  |  |  |  |
| --- | --- | --- | --- |
| Get List of Users | | | |
| **Method** | /user | **Type** | GET |
| **Query String** | /user?UserInfo | **Example** | /user?UserInfo=bluefrog (to just get data of bluefrog ) |
| **Parameters** | - | **Content Type** | application/json |
| **Headers** | {  Authorization: <token>  } | | |
| **Response Body** | {  "Users": [  {  "Username": "1ddeebdb-7b1c-4751-8660-221a1a3b8b21",  "Attributes": [  {  "Name": "sub",  "Value": "333e4f57-da98-44fe-9132-4edd906ceff0"  },  {  "Name": "email\_verified",  "Value": "true"  },  {  "Name": "custom:updatedBy",  "Value": "btpy285@yahoo.com"  },  {  "Name": "preferred\_username",  "Value": "redparrot"  },  {  "Name": "given\_name",  "Value": "Red"  },  {  "Name": "custom:performanceRating",  "Value": "100"  },  {  "Name": "custom:createdBy",  "Value": "btpy285@yahoo.com"  },  {  "Name": "family\_name",  "Value": "Parrot"  },  {  "Name": "custom:role",  "Value": "PU"  },  {  "Name": "email",  "Value": "cally\_ng@nyp.edu.sg"  },  {  "Name": "custom:safUnit",  "Value": "Red Parrot"  }  ],  "UserCreateDate": "2020-04-30T07:01:01.664Z",  "UserLastModifiedDate": "2020-04-30T07:40:29.261Z",  "Enabled": true,  "UserStatus": "CONFIRMED"  }  ]  } | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Add new User | | | |
| **Method** | /user | **Type** | PUT |
| **Parameters** | - | **Content Type** | application/json |
| **Headers** | {  Authorization: <token>  } | | |
| **Request Body** | {  "FirstName": "Iron",  "LastName": "Man",  "SAFUnit": "Ironman",  "Role": "PU",  "CreatorEmail": "system",  "Email": "email@gmail.com"  } | | |
| **Response Body** | If error {  "statusCode": 400,  "body": {  “messsage”:”An account with the email already exists.”  “code”:” UsernameExistsException”  }  }  If success {  "statusCode": 200,  "body": {  “User”:{  <Created user attributes>  }  }  } | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Update User | | | |
| **Method** | /user | **Type** | PATCH |
| **Content Type** | application/json | | |
| **Headers** | {  Authorization: <token>  } | | |
| **Request Body** | {  "UserInfo": "ironman",  "FirstName": "ironman",  "LastName": "ironman",  "SafUnit": "Ironman",  "PerformanceRating": "50",  "Role": "PU",  "UpdatedBy": "system",  "Status": "A"  } | | |
| **Response Body** | If error {  "statusCode": 400,  "body": {  "messsage”:” User does not exist. ",  "code”:” UserNotFoundException"  }  }  If success {  "statusCode": 200,  "body": {}  } | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Forget Password | | | |
| **Method** | /user/forget | **Type** | POST |
| **Content Type** | application/json | | |
| **Headers** | {  Authorization: <token>  } | | |
| **Request Body** | {  "UserInfo": "ironman"  } | | |
| **Response Body** | If error {  "statusCode": 400,  "body": {  "messsage": "User does not exist. "  "code”: "UserNotFoundException"  }  }  If success {  "statusCode": 200,  "body": {  "CodeDeliveryDetails": {  "Destination": "b\*\*\*@g\*\*\*.com",  "DeliveryMedium": "EMAIL",  "AttributeName": "email"  }  }  } | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Confirm Forget Password | | | |
| **Method** | /user/verifyforgetpassword | **Type** | POST |
| **Content Type** | application/json | | |
| **Headers** | {  Authorization: <token>  } | | |
| **Request Body** | The following request is encrypted before it is sent over.  {  "UserInfo": "[liona@gmail.com](mailto:liona@gmail.com)",  "UserPass": "xxxxx",  "Code": "xxxxx"  } | | |
| **Response Body** | If error  {  "statusCode":400,  "body": {  "message": "Invalid verification code provided, please try again.",  "code": "CodeMismatchException",  "time": "2020-07-15T06:53:45.440Z",  "requestId": "3003a875-299f-4c57-ae71-66b3a4ef749b",  "statusCode": 400,  "retryable": false,  "retryDelay": 0.4475507487005759  }  }  If success, data is encrypted  {  "statusCode":200,  "body":{}  } | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Login | | | |
| **Method** | /user/login | **Type** | POST |
| **Content Type** | application/json | | |
| **Headers** | {  Authorization: <token>  } | | |
| **Request Body** | The following request is encrypted before it is sent over.  {"body":"{\"UserInfo\": \"lionu\", \"UserPass\": \"XXXXXXXX\"}"} | | |
| **Response Body** | If error {  "statusCode": 400,  "body": {  “messsage”:”Invalid verification code provided, please try again.”  “code”:” CodeMismatchException”  }  }  If success {  "statusCode": 200,  "body": {  \”keys”\:{  \”AccessToken\”:\”qwerty\”,  \”RefreshToken\”:\”asdfgh\”,  \”IdToken\”:\”zxcvb\”,  },  \”userData\”:{  \”preferred\_username\”:\”\”,  …  }  }  }  If firstTimeLogin {  "statusCode": 200,  "body": {  "ChallengeName":”NewPasswordRequired”,  “Session”:”“  …  }  } | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Login with 2FA | | | |
| **Method** | /user/lionlogin | **Type** | POST |
| **Content Type** | application/json | | |
| **Headers** | {  Authorization: <token>  } | | |
| **Request Body** | The following request content is encrypted before it is sent over.  For Web portal  {"body":"{  "UserInfo": "lionu",  "UserPass": "XXXXXXXX",  "WebPortal": "true"  }"  }  For Mobile  {"body":"{  "UserInfo":"pinkpanther",  "UserPass":"P@55w0rd",  "Platform":"android",  "DeviceToken":"XXXXXXXXX"  } | | |
| **Response Body** | If error {  "statusCode": 400,  "body": {  “messsage”:”Invalid verification code provided, please try again.”  “code”:” CodeMismatchException”  }  }  If success //the userDate returned in encrypted, please decrypt first. {  "statusCode": 200,  "body": {  \”userData\”:{  \”preferred\_username\”:\”\”,  …  }  }  }  If firstTimeLogin {  "statusCode": 200,  "body": {  "ChallengeName":”NewPasswordRequired”,  “Session”:”“  …  }  } | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Complete Registration | | | |
| This function is called when user login for the first time and would require user to complete their registration. | | | |
| **Method** | /user/login | **Type** | PUT |
| **Content Type** | application/json | | |
| **Request Body** *(Note: Session can be retrieved from first time login)* | {  "UserInfo": "ironman@email.com",  "NewPass": "xxxxxx",  "NewUsername": "ironman",  "Session": "xxxxxx" } | | |
| **Response Body** | If error {  "statusCode": 400,  "body": {  “message”: “Invalid session for the user, session is expired.",  "code": "NotAuthorizedException"  }  }  If success {  "statusCode": 200,  "body": {}  } | | |

|  |  |  |  |
| --- | --- | --- | --- |
| First Time Login (Complete Registration with Encryption) | | | |
| This function is called when user login for the first time and would require user to complete their registration. | | | |
| **Method** | /user/firsttimelogin | **Type** | POST |
| **Content Type** | application/json | | |
| **Request Body** *(Note: Session can be retrieved from first time login)* | The following request is encrypted before it is sent over.  {  "UserInfo": "ironman@email.com",  "NewPass": "xxxxxx",  "NewUsername": "ironman",  "Session": "xxxxxx" } | | |
| **Response Body** | If error {  "statusCode": 400,  "body": {  “message”: “Invalid session for the user, session is expired.",  "code": "NotAuthorizedException"  }  }  If success {  "statusCode": 200,  "body": {}  } | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Change Password | | | |
| This function is called when the user requests for a change of password in the profile page. | | | |
| **Method** | /user/changepassword | **Type** | POST |
| **Content Type** | application/json | | |
| **Request Body** *(Note: Session can be retrieved from first time login)* | Each value is encrypted before it is sent over.  {  "CurrentPassword": "xxxxxx",  "NewPassword": "xxxxxxx",  "AccessToken": "xxxxxxx"  } | | |
| **Response Body** | If error  {  "statusCode": 400,  "body": {  "message": {  "message": "Incorrect username or password.",  "code": "NotAuthorizedException",  "time": "2020-07-15T04:02:52.133Z",  "requestId": "d0d2d006-ef9c-41cc-877c-51a91be4c7f2",  "statusCode": 400,  "retryable": false,  "retryDelay": 52.2978258893892  }  }  }  If success  {  "statusCode": 200,  "body": {  "message": {}  }  } | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Verify OTP | | | |
| This function is called when the user verifies their OTP at 2FA page. | | | |
| **Method** | /user/verifyotp | **Type** | POST |
| **Content Type** | application/json | | |
| **Request Body** *(Note: Session can be retrieved from first time login)* | The following request is encrypted before it is sent over.  {  "email": "[liona@gmail.com](mailto:liona@gmail.com)",  "otpDigit": "554569"  } | | |
| **Response Body** | If error  {  "message": "Invalid"  }  If success, data is encrypted  {  "message": "Valid",  "data": {  "idToken": "xxxxxx",  "accessToken": "xxxxxx"  }  } | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Send OTP | | | |
| This function is called when the user requests to resend their OTP at 2FA page. | | | |
| **Method** | /user/sendotp | **Type** | POST |
| **Content Type** | application/json | | |
| **Request Body** *(Note: Session can be retrieved from first time login)* | The following request is encrypted before it is sent over.  {  "email": "[liona@gmail.com](mailto:liona@gmail.com)",  } | | |
| **Response Body** | If error  {  "message": "Invalid"  }  If success, data is encrypted  {  "updatedRows":0,  "addedRows":1  } | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Get Saf Unit | | | |
| **Method** | /user/safunit | E.g. /user/safunit | |
| **Type** | GET | **Content Type** | application/json |
| **Headers** | {  Authorization: <token>  } | | |
| **Response Body** | If Error  {  "statusCode": 400,  "body": [    ]  } If success {  "statusCode": 200,  "headers": {  "Access-Control-Allow-Headers": "Authorization",  "Access-Control-Allow-Origin": "\*"  },  "body": [  "Red Parrot",  "Black Panther",  "DP",  "SP",  "U",  "PU"  ]  } | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Get Performance Rating | | | |
| **Method** | /user/getperformancerating | E.g. /user/getperformancerating | |
| **Parameters** | username | E.g username=blackpanther | |
| **Type** | GET | **Content Type** | application/json |
| **Headers** | {  Authorization: <token>  } | | |
| **Response Body** | If error  {  "message": "User does not exist.",  "code": "UserNotFoundException",  "time": "2020-07-24T11:00:59.028Z",  "requestId": "c2925d40-b259-4401-b03b-da5371723bfa",  "statusCode": 400,  "retryable": false,  "retryDelay": 64.14416554308542  }  If success  {  "username": "blackpanther",  "performanceRating": "16"  } | | |

## VEHICLE

|  |  |  |  |
| --- | --- | --- | --- |
| Get List of Vehicles | | | |
| **Method** | /vehicle | E.g. /vehicle?status=I | |
| **Parameters** | status | status=all, A, I | |
| **Type** | GET | **Content Type** | application/json |
| **Headers** | {  Authorization: <token>  } | | |
| **Response Body** | {  "statusCode": 200,  "header": {  "Content-Type": "application/json",  "Access-Control-Allow-Origin": "\*"  },  "body": [  {  "vehicleId": 5,  "name": "Bus (40 seater)",  "load": 4.1,  "status": "A",  "createdBy": "system",  "createdDate": "2020-04-30T09:56:02.000Z",  "updatedBy": "tempaccount",  "updatedDate": "2020-05-06T08:38:53.000Z"  }  ]  } | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Add new Vehicle | | | |
| **Method** | /vehicle/add | E.g. /vehicle/add | |
| **Type** | PUT | **Content Type** | application/json |
| **Headers** | {  Authorization: <token>  } | | |
| **Request Body** | {  "name":"Van",  "load": "0.3",  "createdBy": "system"  } | | |
| **Response Body** | {  "statusCode": 200,  "header": {  "Content-Type": "application/json",  "Access-Control-Allow-Origin": "\*"  },  "body": {  "fieldCount": 0,  "affectedRows": 1,  "insertId": 17,  "serverStatus": 34,  "warningCount": 0,  "message": "&Records: 1 Duplicates: 0 Warnings: 0",  "protocol41": true,  "changedRows": 0  }  } | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Update Vehicle | | | |
| **Method** | /vehicle/update | E.g. /vehicle/update | |
| **Type** | PATCH | **Content Type** | application/json |
| **Headers** | {  Authorization: <token>  } | | |
| **Request Body** | {  "vehicleId": 5,  "name": "Bus (40 seater)",  "load": 4.1,  "status": "A",  "updatedBy": "tempaccount"  } | | |
| **Response Body** | {  "statusCode": 200,  "header": {  "Content-Type": "application/json",  "Access-Control-Allow-Origin": "\*"  },  "body": {  "fieldCount": 0,  "affectedRows": 1,  "insertId": 0,  "serverStatus": 34,  "warningCount": 0,  "message": "(Rows matched: 1 Changed: 1 Warnings: 0",  "protocol41": true,  "changedRows": 1  }  } | | |