Safar724 API Usage Guide v1.8

Overview

This document describes the resources that make up the Safar724 API v1-rc1. If you have any problems or requests, please contact api@safar724.com.

Schema

All API access is over HTTPS, and accessed from the `https://api.safar724.com/api/v1.0-rc`. All data is sent and received as JSON.

All timestamps returned in **Georgian date and UTC time** with ISO 8601 format (YYYY-MM-DDTHH:MM:SSZ) unless it mentions another type explicitly in the document.

Parameters

For GET requests, any parameters are not specified as a segment in the path can be passed as an HTTP query string parameter.

For POST, PATCH, PUT, and DELETE requests, parameters are not included in the URL should be encoded as JSON with a Content-Type of 'application/json'

Errors

Authentication and Authorization Errors

If server receives data form a client with invalid authentication data, it responds with 401 (Unauthorized) or 403 (forbidden) status code. See authentication section for more info.

Client Errors

If server receives invalid data from the request, it responds with 400 (bad request) status code and puts addition info to describe the error in the body in `Error` object. See possible errors section for more info.

Gateway Timeout Error

If the server did not receive a timely response from the underling company's upstream server, it responds with 504 (gateway timeout) status code. This error is not permanent and most of the time it will be resolved automatically in few minutes.

Internal Server Errors

If the server encountered an unexpected condition that prevent it from fulfilling the request, it responds with 500 (internal server error) status code.

Authentication

Authentication is based on OAuth2.0 protocol. First a POST request must be sent to the token endpoint with following form-url-encoded data:

- grant_type: “password”
- username: <username>
- password: <password>

```
Token Endpoint https://api.safar724.com/token
```

If the request succeeds, the server returns an access token in the response body. To authorize, every request must contain the received token in Authorization header.

```
Authorization: Bearer <token>
```

Test Services

In order to test the API, mocked buses are provided. Developers can easily book tickets from them and test the API. Fake buses can be found in Mashhad-Abadan route at 30th of every Persian month with the company name Bina-Gasht.

Methods

List all cities

```
GET /cities
```

This method lists all of the cities with a specific ID. Provided ID is required for further methods. Since list of cities rarely changes, caching the list for feature use is highly recommended.

Input

```
None
```

Output

```
Array of the City object.
```

List available routes

```
GET /routes/available
```

This method lists available routes in the current Persian month. Since list of available routes is not change frequently, caching the list for feature use is highly recommended.

Input

```
None
```

Output

```
Array of the Route object.
```

List available services

```
GET /buses/{from}/{to}/{date}
```

This method lists available services from the specific route.

Input

From	Required	Origin city ID
To	Required	Destination city ID
Date	Required	Departure date with “YYYY-MM-DD” format

Output

```
Array of the BusSummary object
```

Get a specific service

```
GET /buses/{id}
```

This method returns detailed information of the specific bus

Input

Id	Required	Specific bus ID

Output

```
The Bus object
```

Book a ticket

```
POST /tickets
```

This method books a ticket from the specific service and returns a ticket ID. If online payment method is selected, this method also will return a payment endpoint. To finalize the booking process for the online payment method, user must be redirected to the provided endpoint and should complete payment process within 10 minutes, then ticket is successfully booked and ticket number is issued.

Input

```
TicketToBook object
```

Output

```
TicketSummary object
```

Get a specific ticket

```
GET /tickets/{id}
```

This method returns detailed information of the specific ticket.

Input

id	Required	Specific ticket id

Output

```
Ticket object
```

Get refund overview of a ticket

```
GET /tickets/{id}/refund
```

This method returns detailed information about a ticket which is subject to refund.

Input

id	Required	Specific ticket id

Output

```
RefundOverview object
```

Refund a ticket

```
POST /tickets/{id}/refund
```

This method actually refunds the ticket.

Input

id	Required	Specific ticket id

Output

```
None
```

Get Credit Amount

This methods returns available credit in the user account.

```
GET /profile/credit
```

Output

```
Credit object
```

Hooking

Every booking request has to have a hooking URL which is called by the server when the booking process is finished. The endpoint has to accept POST HTTP method with following form-url-encoded data:

Id	long	ID of the ticket
Status	string	“Issued”, “Failed”, “PaymentUnsuccessful” “Canceled”, or “Rejected”

Objects

City

ID	string	City ID
Name	string	City name

Route

From	string	Origin City ID
To	string	Destination City ID

BusSummary

ID	string	Service ID	

Company	string	Company name	
BoardingPoint	Station	Service origin	
DroppingPoints	array of Station object	Service Destinations	
DepartureDate	DateTime	Service departure time	
Price	int	Each seat’s price	Obsolete. Don’t use this field it will be removed in the next version. Use Financial field instead.
Type	string	Bus type	
AvailableSeats	int	Number of available seats	
Financial	Financial	Financial information of the service	

Station

City	string	City name
Terminal	string	Terminal name

Financial

Price	int	Each seat’s price
MaxApplicableDiscountPercentage	float	The maximum amount of discounts the company allowed to apply
AvailablePaymentMethods	array of string	Lists available payment methods for the service. (“Online”, or “Credit”)
CommissionPercentage	float	Agency’s commission (in percentage)

Bus

ID	string	Service ID	
Company	string	Company name	
BoardingPoint	Station	Service origin	
DroppingPoints	array of Station	Service destination	
DepartureDate	DateTime	Service departure time	
Price	int	Each seat’s price	Obsolete. Don’t use this field it will be removed in the next version. Use Financial field instead.
Type	string	Bus type	

AvailableSeats	int	Number of available seats	
CommissionPercentage	float	Agency’s commission (in percentage)	Obsolete. Don’t use this field it will be removed in the next version. Use Financial field instead.
Seates	array of Seat object	Seats specification	
Financial	Financial	Financial information of the service	

Seat

Number	int	Seat number
Column	int	Related column of the seat
Row	int	Related row of the seat
Status	string	“Available”, “BookedForMale”, or “BookedForFemale”

TicketToBook

BusID	string	Service ID
PaymentMethod	string	“Online”, or “Credit”
DesiredDiscountPercentage	float	Desired discount to be applied in percentage
Passengers	array of Passenger object	Passengers information
Contact	Contact	Owner’s contact information
HookUrl	string	Ticket information will be returned to this url

Passenger

FirstName	string	First name of the passenger
LastName	string	Last name of the passenger
SeatNumber	int	The seat passenger will seat
NationalCode	string	Passengers national code
Gender	string	“Female”, or “Male”

Contact

Name	string	Owner’s first and last name
MobilePhone	string	Owner’s mobile phone number
Email	string (optional)	Owner’s email address

TicketSummary

ID	long	Ticket ID
PaymentEndpoint	string	The URL which user must be redirected to complete booking process.

Ticket

ID	long	Ticket ID
BusID	string	Bus ID
Contact	Contact	Owner’s contact information
TicketNumber	string	Reference number on the end user’s ticket
Company	string	Company name
BoardingPoint	Station	Service origin
DroppingPoint	Station	Service destination
DepartureDate	DateTime	Service departure time
BusType	string	Bus type
Price	int	Final price of the ticket
Discount	int	Final discount of the ticket
Passengers	Array of the Passenger object	Passengers information
Status	string	“Pending”, “Issued”, “Refunded”, “Failed”, “PaymentUnsuccessful” “Canceled”, or “Rejected”

RefundOverview

ID	long	Ticket ID
TicketPrice	int	Final price of the ticket
RefundAmount	int	Refundable amount after deductions
Status	string	“Refundable”, “Nonrefundable”, or “Refunded”

Credit

Amount	int	Credit amount in Iranian Rial

Error

Code	string	An unique code for each error type
Message	string	Additional info to describe the error

Possible Error Codes

Code	Description
`BadArgument`	The request data is invalid. Check `ModelState` field for more info.
`InvalidTicketStatus`	Ticket status is invalid for this operation.
`RefundRequestIsRejected`	The underlying company rejected the refund request. Possibly the ticket was printed.
`RefundDateExpired`	Refund date expired. The ticket is no longer refundable through the system.
`SeatUnavailable`	Selected seat is unavailable.
`InvalidBankAccountNumber`	Bank account is not registered or invalid.
`InvalidBusServiceStatus`	Service status is invalid for this operation.
`InsufficientCredit`	Credit amount is not sufficient for this payment.
`InvalidDiscount`	Requested discount is not applicable for this service.