Alterra Travel chat engine API

This document describes the API to Alterra.ai, a virtual AI travel agent. It can recommend where to go on vacation and book flights and hotels there.

Using this API third-party developers (aka clients) may build their own virtual travel agents, embed the functionality in broader virtual assistants, pair it up with live travel agents, etc.

Alterra is a natural language processing engine. Clients may access it from a variety of i/o conduits, such as:
- Messenger bots (Facebook, Slack, Skype, etc.);
- Live chat on their websites and in apps (Intercom, Salesforce Desk, Zendesk, LiveChat, etc.);
- SMS;
- Email;
- Telephone;
- Voice assistants (with the help from third-party voice recognition engines);
- Free text NLP search on their websites and in apps, etc.

An example of Facebook Messenger bot implementation can be found here:
http://m.me/alterra.cc

The intended usage is as follows: client receives a free-text message from his user, passes it to this API, receives the bot's reply back and displays it to the user.

Alterra API takes free text as input and returns:
- user query intent;
- parsed user query;
- respective search results;
- finished rich text reply ready to show to the user.

First, Alterra intent classifier takes free-form text (user query) as input and attempts to determine user intent. Supported intents are: `book_hotel`, `book_flight`, `destination_recommendation` and `other`. They are returned in the `intent` field.

Next, Alterra query parser (aka slot filler) takes the free-form user query, extracts all necessary features and parameters such as cities, dates, trip duration, target price, class of service, additional amenities, etc. and returns them in a structured form, in the `parsed_query` field.

Alterra then passes these structured queries to its preferred search engines (below) and returns respective search results, in the `search_results` field.

Alterra uses the following search engines:
- `book_hotel` intent: Hotels Combined;
- `book_flight` intent: Google Flights;
- `destination_recommendation` intent: Alterra vacation recommender;

- `other` intent: Alterra misc. chat engine.
(Flight search by Kiwi.com is available upon request.)

The client who wishes to use different search engines may pass the structured queries to his search engine of choice.

Finally, the search results are processed to generate rich text replies ready to be displayed to the user. They are returned in in the `bot_replies` field.

Note that Alterra keeps context, indefinitely. For example, suppose the user's first request is "a 3 star hotel in new york from oct 22 till oct 28 with free wi-fi". Alterra would return the respective search results. Then, he types just "and gym". Alterra will add "gym" to the previous query. Accordingly, in the `parsed_query` field, it will return the parsed version of the previous query, plus "gym".

Likewise, Alterra allows users to edit their previous queries. E.g, if the user says just "check-out on oct 30", Alterra will return the full query, with oct 28 replaced with oct 30.

The bot tries to guess if the user keeps editing an old query, or has started anew. It acts accordingly. If it is in doubt, it may ask the user for clarification. Also, the user may always override the bot's logic and start a completely new query by typing "start over", "clear", or the like.

In any event, the bot always returns its state in the `parsed_query` field.

API v2 returns all four components: intent, parsed user query, raw search results and finished rich text bot replies.
API v1 returns only the last portion -- finished rich text bot replies.
Clients not interested in knowing what's going on under the hood shall use v1. It hides the internals, it's simpler.

To request an API access key please contact info@alterra.ai

API transport and endpoints

The API calls are done with JSON over HTTPS protocol using https://alterra.ai/api/travel endpoint. The client should issue a POST request with JSON-formatted body. The answer is also JSON-formatted.

JSON request format

The request data format is a JSON object with the following fields:

Field name	Type	Required?	Comment
`client_key`	string	Y	API access key assigned at sign-up
`version`	string	Y	Version of the protocol used (v1 or v2)
`session_id`	string	Y	ID of the conversation (see note below)
`timestamp`	int	Y	The time of the event, as UNIX timestamp
`message`	string	Y	The message sent by user

JSON reply format

Alterra.ai's replies are JSON objects with the fields described below.

API v1

API v1 returns finished rich text bot's replies, ready to be displayed to the user.

Field name	Type	Always present?	Comment
`status`	ENUM	Y	Processing status (`"OK"` or `"ERR"`)
`error`	string	N	Error message, in case `status` is `"ERR"`
`bot_replies`	list of objects	N	Rich text bot's replies
`intent`	ENUM	Y	User query intent, as recognized by the bot (`book_hotel`, `book_flight`, `destination_recommendation` or `other`)

Each reply is an object with fields:

Field name	Type	Always present?	Comment
`parts`	list of objects	Y	Parts of reply

Each part is either text, image, or a link. The image can also be a link (that user would follow if they click on the image).

Field name	Type	Always present?	Comment
`text`	string	N	Text of message to be sent to the user
`image_url`	string	N	URL of an image to be sent
`image_link`	string	N	URL of the image’s landing page
`link_url`	string	N	Link URL to be sent to the user

Field name	Type	Always present?	Comment
`link_text`	string	N	Title of the link

Exactly one of `text`, `image_url`, or `link_url` will be present in any part, other fields are optional (`image_link` can only be present if there is an `image_url`, and `link_text` can only be present if there is a `link_url` in the same part).

API v2

API v2 returns four components: intent, parsed user query, raw search results and finished rich text bot replies.

Field name	Type	Always present?	Comment
`status`	ENUM	Y	Processing status (currently `"OK"` or `"ERR"`)
`error`	string	N	Error message, in case `status` is `"ERR"`
`intent`	ENUM	Y	User query intent, as recognized by the bot (`book_hotel`, `book_flight`, `destination_recommendation` or `other`)
`parsed_query`	JSON object	N	Structured data extracted from the query
`search_results`	JSON object	N	Search results found using the paresed query
`bot_replies`	list of objects	N	Rich text bot's replies

Parsed query

In case of hotels intent, the `parsed_query` block may contain the follow parts:

Field name	Type	Always present?	Comment
`accommodation_type`	string	N	Type of accommodation (`hotel`, `apartment`, `bnb`, etc.)
`dates`	JSON object	N	Trip dates (check-in, check-out, duration, etc.)
`location`	JSON object	N	Trip location (coordinates, address, etc.)
`star_rating`	string	N	Number of stars delimited by vertical line
`max_price`	string	N	Max price per night, number as string
`currency`	string	N	Currency code (`USD`, `EUR`, `GBP` or `JPY`)
`facilities`	string	N	List of facilities, delimited by comma

In case of flights intent, the `parsed_query` block may contain the follow parts:

Field name	Type	Always present?	Comment
`dates`	JSON object	N	Trip dates (departure, return, trip duration, etc)
`airport_from`	list	N	IATA codes of departure airports
`airport_to`	list	N	IATA codes of arrival airports
`airline`	list	N	Airline IATA codes
`alliances`	list	N	Airline alliance names
`class_type`	ENUM	N	`ECONOMY` or `BUSINESS`
`currency`	string	N	Currency code (`USD`, `EUR`, `GBP` or `JPY`)
`fare_amount`	int	N	Max price
`round_trip`	bool	N	Oneway or round-trip
`stops`	list	N	Number of stops
`max_flight_duration_hrs`	int	N	Maxim duration of the flight, hours

User queries may include flexible dates, e.g. "leaving on Feb 11, returning between Feb 21 and 23". This would be parsed as: "date_depart": {"start":"2017-02-11","end":"2017-02-11"}, "date_return": {"start":"2017-02-21","end":"2017-02-23"}

In case of `destination_recommendation` and `other` intents, no `parsed_query` block is returned.

Search results

`search_results` contains URLs of the search result pages. It consists of the follow possible fields:

Field name	Type	Always present?	Comment
`search_engine`	ENUM	Y	Search engine used (currently `"hotels_combined"` or `"google_flights"`)
`cheapest_url`	string (URL)	N	URL of the cheapest flight itinerary
`shortest_url`	string (URL)	N	URL of the shortest flight itinerary
`best_value_url`	string (URL)	N	URL of the recommended flight itinerary
`first_result_url`	string (URL)	N	URL of the first hotel search result
`all_results_url`	string (URL)	N	URL of the list of hotels found (SERP)

In case of `destination_recommendation` and `other` intents, no `search_results` block is returned.

Bot replies

`bot_replies` block has the same structure as in API v1.