STROLLIN REST API SERVER

NOTE:

- The list is separated in Object Related (See the Index).
- Each object will presents the schema and the available API requests.
- Where full URLs are provided in responses they will be rendered as if service is running on 'https://' + IP_SERVER + ':' + PORT_SERVER + '/'.
- All documents in the database has the variable "id" (Some Exceptions). Do not use "_id" given by MongoDB.
- This document show some type as "ObjectID" which represent the "id" of a model. (Example: `UserID` represent user.id)
- In case of successful request, all request will return an JSON object as:
  {
  status: String
  eventual return: Object (See in requests for details)
  }
- In case of failed request, all request will return an error code which correspond of the next list:
  1: You must be authenticated
  2: An error has occured in the server
  3: An internal error has occured
  4: An error in the data has occured
  5: You must be a subscribed to access this functionality
  100: The mail is not valid
  101: The mail is already used
  102: The password must contains 6 characters with at least 1 uppercase, 1 lowercase and 1 digit
  103: The login or the password is incorrect
  104: You made too much attempt. Please retry later.

INDEX:

BLACKLIST RELATED:

DESCRIPTION

Blacklist is used to save the IP of the user and lock the attempt to connect if the user send wrong authentication.

SCHEMA:

- ip `String`
- attempt `Number`
- lock_date `Number`

REQUESTS:

None

COMMENT RELATED:

DESCRIPTION

The object Comment represents the comments and rates left by an user to a location or a course.

SCHEMA:

- id `Number`
- message `String`
- author_id `UserID`
- author_pseudo `String`
- score `String`
- creation_date `String`
- modification_date `Number`

REQUESTS:

- POST new comment: `POST /comment/new_comment`

Description	Headers	Body	Return
Create a new comment in a location or a course with a score. If the user commented before, the comment will be updated	access_token: String location_id: LocationID or course_id: CourseID	message: String score: String	None

- GET get comment by ID: `GET /comment/get_comment_by_id`

Description	Headers	Body	Return
Get comment's data by ID or list of ID.	access_token: String comments_list: [CommentID]	None	comments_list[CommentObject] (see Schema)

CONVERSATION RELATED:

DESCRIPTION

The object Conversation represents the chat room for the user. It contains a list of Message Object.

SCHEMA:

- id `Number`
- participants `[userID]`
- name `String`
- messages_list `[messageID]`
- recent_messages `[{messageObject}]` (See MESSAGE RELATED)

REQUESTS:

- GET conversations: `GET /conversation/get_conversations`

Description	Headers	Body	Return
Get all the conversations where the user appears.	access_token: String	None	conversations: [conversationObject] (See Schema)

- GET get conversation by ID: `GET /conversation/get_conversation_by_id`

Description	Headers	Body	Return
Get conversation's data by ID or list of ID.	access_token: String conversations_list: [ConversationID]	None	conversations_list: [ConversationObject] (see Schema)

COURSE RELATED:

DESCRIPTION

The object Course contains the informations of the course saved in database.

SCHEMA:

- id `Number`
- locations_list `[locationID]`
- name `String`
- score `String`
- number_used `String` : number of usage of the course
- author_id `UserID`
- author_pseudo `String`
- creation_date `String`
- timetable `String`
- price_range `[String, String, String]` : Min, Max, Avg
- comments_list `[commentID]`
- tags_list `[tagID]`
- time_spent: `[String]` (list of time spent to do the course)

REQUESTS:

- ADD new course: `POST /course/new_course`

Description	Headers	Body	Return
Add a course to the database.	access_token: String	locations_list: [LocationID] name: String	course: CourseObject (See Schema)

- GET course: `GET /course/get_course`

Description	Headers	Body	Return
Get a list of courses. Can be sorted by name, number_used, score, tendency (tendency_range is at 30 days by default) or favorites	access_token: String sort: String tendency_range: Number (Optional)	None	courses_list: [CourseObject] (See Schema)

- GET course: `GET /course/get_user_historic`

Description	Headers	Body	Return
Get a list of course in the historic of the user. Size is the number of course to return max	access_token: String size: Number (default: 10)	None	course_historic: [CourseObject] (See Schema)

- GET get course by ID: `GET /course/get_course_by_id`

Description	Headers	Body	Return
Get course's data by ID or list of ID.	access_token: String courses_id_list: [CourseID]	None	courses_list: [CourseObject] (see Schema)

FAQ RELATED:

DESCRIPTION

The object Faq represents the question sent by users and non users about Strollin.

SCHEMA:

- id `Number`
- author `String`
- question `String`
- creation_date `String`
- answer `String`
- published `Boolean`
- language `String` : "fr" || "en"

REQUESTS:

- ADD Create a question: `POST /faq/create_question`

Description	Headers	Body	Return
Create a new question for the FAQ.	None	mail: String question: String language: String	None

- ADD answer to a question: `POST /faq/answer_question`

Description	Headers	Body	Return
Add an answer to a question and set the published state.	question_id: QuestionID	answer: String published: Boolean	None

- GET all "fr" questions: `GET /faq/get_question_fr`

Description	Headers	Body	Return
Get questions in french section.	None	None	faqs_list: [questionObject] (see Schema)

- GET all "en" questions: `GET /faq/get_question_en`

Description	Headers	Body	Return
Get questions in english section.	None	None	faqs_list: [questionObject] (see Schema)

- GET get question by ID: `GET /faq/get_question_by_id`

Description	Headers	Body	Return
Get question's data by ID or list of ID.	faqs_list: [questionID]	None	[questionObject] (see Schema)

GENERATOR RELATED:

DESCRIPTION:

Generator course is not a model but it is used for the algorithm of Strollin and interact with the database. The algorithm will return an array of object containing:

- AlgAg: Number,
- AlgDisp: Number,
- City: String,
- Desc: String,
- Dist: Number,
- Id: Number,
- Name: String,
- PopAg: Number,
- PopDisp: Number,
- Pos: [Number],
- Price: Number,
- Tags: [String],
- TagsDisp: [Number],
- Time: Number

REQUESTS:

- GET a generated course: `GET /generator/generate_course`

Description	Headers	Body	Return
Generate a course according to the given parameters. Will automatically create locations in database if new locations are found.	access_token: String time: String budget: String tags: [String] coordinate: [String]	None	generated_course: [AlgorithmObject] course: PartialCourseObject (Not created in database)

- GET a generated popup: `GET /generator/generate_popup`

Description	Headers	Body	Return
Generate a popup of a location owned by a partner.	access_token: String coordinate: [String]	course: CourseID	popup: LocationObject

- GET popup answer: `GET /generator/popup_answer`

Description	Headers	Body	Return
Answer of the user for the previously generated popup.	access_token: String anwser: Boolean popup: LocationObject	None	WIP

IMAGE RELATED:

DESCRIPTION

The object Image represents the stored images in the server for the user's profile or the shared images in conversations.

SCHEMA:

- id `Number`
- author `String`
- uri `String`
- mimetype `String`

REQUESTS:

- GET an image: `GET /image/id`

Description	Headers	Body	Return
Get an image to display.	id: String	None	image: ImageObject

- POST an image: `GET /image/upload`

Description	Headers	Body	Return
Upload an image in the server.	None	access_token: String filename: String mimetype: String	image: ImageObject

LOCATION RELATED:

DESCRIPTION

The object Location represents the locations used in courses. It is also used for the partner user.

SCHEMA:

- id `Number`
- name `String`
- owner_id `UserID`
- owner_pseudo `String`
- score `String`
- latitude `String`
- longitude `String`
- address `String`
- city `String`
- country `String`
- description `String`
- photo `[String]`
- timetable `String`
- comments_list `[commentID]`
- tags_list `[{tag_id: tagID, disp: String}]`
- price_range `[String, String, String]` : Min, Max, Avg
- average_time `String`
- phone `String`
- website `String`
- pop_disp `String`
- pop_ag `String`
- alg_disp `String`
- alg_ag `String`
- food `Boolean`

REQUESTS:

- ADD new location: `POST /location/new_location`

Description	Headers	Body	Return
Add a location to the database.	access_token: String	name: String owner: userID (Optional) latitude: Number longitude: Number address: String city: String (optional) country: String (optional) description: String (optional) price_range: [String, String, String] (optional) timetable: String (optional) tags_list: [tagID] (optional) average_time: String (optional) phone: String (optional) website: String (optional) food: Boolean	None

- UPDATE a location: `POST /location/update_location`

Description	Headers	Body	Return
Update a location's data (at least one parameter in body).	access_token: String location_id: locationID	name: String (optional) owner: userID (optional) latitude: Number (optional) longitude: Number (optional) address: String (optional) city: String (optional) country: String (optional) description: String (optional) price_range: [String, String, String] (optional) timetable: String (optional) average_time: String (optional) phone: String (optional) website: String (optional) food: Boolean	None

- ADD tag(s) to a location: `POST /location/add_location_tag`

Description	Headers	Body	Return
Update a location's tags.	access_token: String location_id: locationID	tags_list: [TagName]	None

- GET place: `GET /location/get_place`

Description	Headers	Body	Return
Get a place from Google Place API.	access_token: String place_name: String	None	result: place Object from Google Place API

- GET location: `GET /location/get_locations`

Description	Headers	Body	Return
Get a list of all locations.	access_token: String	None	locations_list: [LocationObject]

- GET location: `GET /location/get_partner_location`

Description	Headers	Body	Return
Get the location of the partner.	access_token: String	None	location: LocationObject

- GET get location by ID: `GET /location/get_location_by_id`

Description	Headers	Body	Return
Get location's data by ID or list of ID.	access_token: String locations_id_list: [LocationID]	None	locations_list: [LocationObject] (see Schema)

MESSAGE RELATED:

DESCRIPTION

The object Message is the message sent in conversations. It can contains simple messages, images or courses.

SCHEMA:

- id `Number`
- expeditor_id `userID`
- expeditor_pseudo `String`
- conversation `conversationID`
- creation_date `String`
- type `String` : "message" || "image" || "course"
- message `String`

REQUESTS:

- GET messages: `GET /message/get_message`

Description	Headers	Body	Return
Get the data of messages provided in an array of message's ID.	access_token: String messages_id: messageID	None	message: [messageObject] (See Schema)

- GET get message by ID: `GET /message/get_message_by_id`

Description	Headers	Body	Return
Get message's data by ID or list of ID.	access_token: String message_id: [MessageID]	None	messages_list: [MessageObject] (see Schema)

SUBSCRIPTION RELATED:

DESCRIPTION

The subscription routes manages the subscription of users. The subscription uses Stripe to manage the payment.

SCHEMA:

None

REQUESTS:

- CREATE a subscription session: `POST /subscription/create_session`

Description	Headers	Body	Return
Create/Retrieve a customer in Stripe and create a securised session of payment with Stripe. Return a link to the session.	access_token: String	None	url: String

- GET a subscription's data: `GET /subscription/get_subscription`

Description	Headers	Body	Return

Get informations of an user's subscription. Return Null if no information.	access_token: String	None	subscription: {
			current_period_start: String
			currend_period_end: String
			cancel_at_period_end: Boolean
			}

- STOP a subscription: `POST /subscription/stop_subscription`

Description	Headers	Body	Return

Cancel an user's subscription. The subscription remains active for the duration of the paid period.	access_token: String	None	subscription: {
			current_period_start: String
			currend_period_end: String
			cancel_at_period_end: Boolean
			}

TAGS RELATED:

DESCRIPTION

The object Tag is used for users, locations and courses.

SCHEMA:

- id `Number`
- name `String`
- description `String`
- number_used `String`
- locations_list `[locationID]`
- courses_list `[courseID]`

REQUESTS:

- POST new tag: `POST /tags/new_tag`

Description	Headers	Body	Return
Create a new tag.	access_token: String	name: String description: String (Optional)	None

- GET tag: `POST /tags/get_tag`

Description	Headers	Body	Return
Get a list of tag with sort (name, number_used) and filter with name search.	access_token: String sort: String (optional) search: String (optional)		tags: [tagObject] (See Schema)

- GET get tag by ID: `GET /tag/get_tag_by_id`

Description	Headers	Body	Return
Get tag's data by ID or list of ID.	access_token: String tags_list: [TagID]	None	tags_list: [TagObject] (see Schema)

USER RELATED:

DESCRIPTION

The object User represents the users of the application Strollin and the partner.

SCHEMA:

- id `Number`
- id_image_profile `ImageID`
- mail `String`
- password `String`
- creation_date `String`
- pseudo `String`
- partner `Boolean`
- first_name `String`
- last_name `String`
- access_token `String`
- tags_list `[tagID]`
- friends_list `[userID]`
- friends_request `[userID]`
- groups `[[group_name, userID, ...]]`
- course_historic `[[courseID, String]]` : ID and date
- course_favorites `[courseID]`
- socket_id `String`
- facebook_id `String`
- stripe_id `String`
- subscription_id `String`
- verify `Boolean`

REQUESTS:

- ADD new user for registration: `POST /users/register`

Description	Headers	Body	Return
Register a new user for database. Return a access token on success. An email will be sent to check the validity.	None	mail: String password: String partner: Boolean pseudo: String (optional) first_name: String (optional) last_name: String (optional)	access_token: String

- RESET the password: `POST /users/reset_password`

Description	Headers	Body	Return
Reset the password of the user. An email will be sent confirm the action.	None	mail: String	None

- VERIFY the user's mail: `GET /users/verify`

Description	Headers	Body	Return
Check and set to true the variable "verify".	id: String	None	None

- CONFIRM the password's reset: `POST /users/reset_password_confirme`

Description	Headers	Body	Return
Confirm the reset and edit the password.	None	user_id: String password: String confirm_password: String	None

- UPDATE user's information: `POST /users/edit_profile`

Description	Headers	Body	Return
Edit user's information.	access_token: String	password: String (optional) pseudo: String (optional) first_name: String (optional) last_name: String (optional)	None

- ADD a friend: `POST /users/add_friend` (Beta Version)

Description	Headers	Body	Return
Add to "friends_list" a user by mail.	access_token: String	friend_mail: String	None

- ADD new tags in user's tags: `POST /users/add_tag`

Description	Headers	Body	Return
Add existing tag in user's list.	access_token: String	tags_list: [tagID]	None

- REMOVE tags in user's tags: `POST /users/remove_tag`

Description	Headers	Body	Return
Remove existing tag in user's list.	access_token: String	tags_list: [tagID]	None

- ADD a course in the user's historic: `POST /users/add_historic`

Description	Headers	Body	Return
Add existing course in user's historic list with the current date.	access_token: String	course: courseID	None

- ADD a course in the user's historic: `POST /users/add_favorite`

Description	Headers	Body	Return
Add existing course in user's favorites list.	access_token: String	course: courseID	None

- REMOVE user's friend: `POST /users/remove_friend`

Description	Headers	Body	Return
Remove a friend from the friends list. It will remove your id from the friend's friends list	access_token: String	friend_id: UserID	None

- REMOVE user's favorite: `POST /users/remove_favorite`

Description	Headers	Body	Return
Remove a course from the favorite list.	access_token: String	course_id: CourseID	None

- REMOVE user's tags: `POST /users/remove_tags`

Description	Headers	Body	Return
Remove a tag from the user's list.	access_token: String	tags_list: tags_list	None

- GET log in: `GET /users/login`

Description	Headers	Body	Return
Log in an user to get a valid access token.	mail: String password: String	None	access_token: String

- GET log out: `GET /users/logout`

Description	Headers	Body	Return
Log out an user. Make unavailable the current access token.	access_token: String	None	None

- GET user's profile: `GET /users/get_own_profile`

Description	Headers	Body	Return

Get the current user's profile.	access_token: String	None	profile: {
			id: UserID,
			mail: String,
			creation_date: Date,
			pseudo: String,
			partner: Boolean
			first_name: String,
			last_name: String,
			tags_list: [tagID],
			friends_list: [userID],
			friends_request: [userID],
			groups: [[group_name, userID1, userID2, ...]],
			course_historic: [[courseID, date (String)]],
			course_favorites: [courseID]
			}

- GET an user's profile: `GET /users/get_user_profile`

Description	Headers	Body	Return

Get an user's profile by its ID.	access_token: String
	user_id: String
		None	profile: {
			id: UserID,
			mail: String,
			creation_date: Date,
			pseudo: String,
			partner: Boolean
			first_name: String,
			last_name: String,
			tags_list: [tagID],
			friends_list: [userID]
			}

- GET an user's tags: `GET /users/get_user_tags`

Description	Headers	Body	Return
Get tags of users provided in array of userID.	access_token: String user_id: String	None	all_user_tags: [TagObject] (See TAG RELATED)

- GET get users: `GET /user/get_users`

Description	Headers	Body	Return
Get all user's data.	access_token: String	None	users_list: [UserObject] (see Schema)

- GET get user by ID: `GET /user/get_user_by_id`

Description	Headers	Body	Return
Get user's data by ID or list of ID.	access_token: String user_id: [UserID]	None	users_list: [UserObject] (see Schema)

- REMOVE user's account: `DEL /users/remove_account`

Description	Headers	Body	Return
Delete an user's account.	access_token: String password: String	None	None