openapi: 3.0.0

info:

description: |

This is a sample Petstore server. You can find

out more about Swagger at

[http://swagger.io](http://swagger.io) or on

[irc.freenode.net, #swagger](http://swagger.io/irc/).

version: "1.0.0"

title: Swagger Petstore

termsOfService: 'http://swagger.io/terms/'

contact:

email: apiteam@swagger.io

license:

name: Apache 2.0

url: 'http://www.apache.org/licenses/LICENSE-2.0.html'

servers:

# Added by API Auto Mocking Plugin

- description: SwaggerHub API Auto Mocking

url: https://virtserver.swaggerhub.com/RICARDOBALTAZARPEDRO/FoPlaces/1.0.0

- description: SwaggerHub API Auto Mocking

url: https://virtserver.swaggerhub.com/FILIPENPEDROSO/Studyyyyyyyyyyyyyyytime/1.0.0

- url: 'https://petstore.swagger.io/v2'

tags:

- name: studyPlaces

description: Collection of the StudyPlaces for each city

- name: nearUserStudyPlaces

description: Handles the user search state (user specifies city for scalability)

- name: studyPlaceSuggestion

description: User's StudyPlace Suggestions

- name: studyPlaceCrowdSourcingInfo

description: In-Time Availability of each StudyPlace

- name: studyPlaceReviewsAndRatings

description: Review and Ratings of each StudyPlace

- name: userProfileManager

description: Everything about the User

paths:

/studyPlaces/{city}/{studyPlace-id}:

get:

tags:

- studyPlaces

summary: Get all the characteristics of a study place with a specific ID;

description: Get all the characteristics of a study place with a specific ID;

operationId: getStudyPlace

parameters:

- name: city

in: path

description: Name of the location of the search

required: true

schema:

type: string

- name: studyPlace-id

in: path

description: StudyPlace id.

required: true

schema:

type: string

responses:

'200':

description: successful operation

content:

application/json:

schema:

type: array

items:

$ref: '#/components/schemas/StudyPlace'

/studyPlaces/{city}:

post:

tags:

- studyPlaces

summary: API call to upload a new StudyPlace

description: After the StudyPlace Suggestion is validated by an admin/moderator, it is transformed into a "real" studyPlace and uploaded to the server.

operationId: uploadStudyPlace

parameters:

- name: city

in: path

description: Name of the location of the StudyPlace

required: true

schema:

type: string

requestBody:

content:

application/x-www-form-urlencoded:

schema:

type: object

properties:

studyPlace:

$ref: '#/components/schemas/StudyPlace'

responses:

'200':

description: successful operation

content:

application/json:

schema:

type: array

items:

$ref: '#/components/schemas/StudyPlace'

/nearUserStudyPlaces/{city}/{user-id}:

get:

tags:

- nearUserStudyPlaces

summary: User requests the nearest StudyPlaces with his preferences

description: API call returns the list of the nearest StudyPlaces to his location, inside the km-radius specified and with the filters specified.

operationId: getNearUserStudyPlaces

parameters:

- name: city

in: path

description: Name of the location of the search

required: true

schema:

type: string

- name: user-id

in: path

description: ID of the user-id

required: true

schema:

type: integer

format: int64

responses:

'200':

description: successful operation

content:

application/json:

schema:

$ref: '#/components/schemas/StudyPlace'

'400':

description: Invalid ID supplied

'404':

description: StudyPlace not found

delete:

tags:

- nearUserStudyPlaces

summary: Deletes user past search

description: This API call allows users to delete their saved StudyPlaces list for a specific city and user ID. Once the request is processed, the saved StudyPlaces search list will be permanently deleted from the server, so the user can create another search

operationId: deleteNearUserStudyPlaces

parameters:

- name: city

in: path

description: Name of the location of the search

required: true

schema:

type: string

- name: user-id

in: path

description: ID of the user-id

required: true

schema:

type: integer

format: int64

responses:

'200':

description: successful operation

'400':

description: Invalid ID supplied

'404':

description: StudyPlace not found

/nearUserStudyPlaces/{city}/{user-id}/searchInfo:

post:

tags:

- nearUserStudyPlaces

summary: User uploads his locations and preferences

description: The User uploads his locations, prefered radius distance, and optionally, a filter list, to the StudyPlace Manager microService.

operationId: postNearUserStudyPlaces

parameters:

- name: city

in: path

description: Name of the location of the search

required: true

schema:

type: string

- name: user-id

in: path

description: ID of the user-id

required: true

schema:

type: integer

format: int64

responses:

'200':

description: successful operation

'400':

description: Invalid ID supplied

'404':

description: StudyPlace not found

requestBody:

content:

application/x-www-form-urlencoded:

schema:

type: object

properties:

filters:

type: array

items:

$ref: '#/components/schemas/Filter'

km-radius:

description: Radius in km

type: number

format: double

longitude:

description: Longitude of the user

type: number

format: double

latitude:

description: Latitude of the user

type: number

format: double

'/ratings/{city}/{study-place-id}':

post:

tags:

- studyPlaceReviewsAndRatings

summary: Submit the rating of a StudyPlace

description: Create the rating of a StudyPlace; Some characteristics can be null; These ratings are the multiple filters available;

operationId: postRating

parameters:

- name: city

in: path

description: Name of the location of the search

required: true

schema:

type: string

- name: study-place-id

in: path

description: ID of studyplace to return

required: true

schema:

type: integer

format: int64

responses:

'405':

description: Invalid input

requestBody:

$ref: '#/components/requestBodies/ListOfFilters'

get:

tags:

- studyPlaceReviewsAndRatings

summary: Get the ratings of a study place

description: Get the ratings of a study place with a specific ID;

operationId: getRatings

parameters:

- name: city

in: path

description: Name of the location of the search

required: true

schema:

type: string

- name: study-place-id

in: path

description: ID of studyplace to return

required: true

schema:

type: integer

format: int64

responses:

'200':

description: successful operation

content:

application/json:

schema:

type: array

items:

$ref: '#/components/schemas/Rating'

'400':

description: Invalid ID supplied

'404':

description: StudyPlace not found

'/reviews/{city}/{study-place-id}':

post:

tags:

- studyPlaceReviewsAndRatings

summary: Review of a StudyPlace

description: Submits a description of the place and if possible, the description of the experience;

operationId: postReview

parameters:

- name: city

in: path

description: Name of the location of the search

required: true

schema:

type: string

- name: study-place-id

in: path

description: ID of studyplace to return

required: true

schema:

type: integer

format: int64

responses:

'405':

description: Invalid input

security:

- petstore\_auth:

- 'write:pets'

- 'read:pets'

requestBody:

$ref: '#/components/requestBodies/Review'

get:

tags:

- studyPlaceReviewsAndRatings

summary: Get the reviews of a study place

description: Get the reviews of a study place with a specific ID; A review could be information that is missing in the ratings;

operationId: getReviews

parameters:

- name: city

in: path

description: Name of the location of the search

required: true

schema:

type: string

- name: study-place-id

in: path

description: ID of studyplace to return

required: true

schema:

type: integer

format: int64

responses:

'200':

description: successful operation

content:

application/json:

schema:

type: array

items:

$ref: '#/components/schemas/Review'

'400':

description: Invalid ID supplied

'404':

description: StudyPlace not found

/crowdSourcing/{city}/{study-place-id}:

get:

tags:

- studyPlaceCrowdSourcingInfo

summary: Get info about the current crowdness of the StudyPlace

operationId: getCrowdSourcing

parameters:

- name: city

in: path

description: Name of the location of the search

required: true

schema:

type: string

- name: study-place-id

in: path

description: ID of studyplace to return

required: true

schema:

type: integer

format: int64

responses:

'200':

description: successful operation

content:

application/json:

schema:

type: array

items:

$ref: '#/components/schemas/StudyPlaceCrowdSourcingInfo'

post:

tags:

- studyPlaceCrowdSourcingInfo

summary: User updates current information about StudyPlace

description: If possible, the user updates information about the characteristics of his/her visit to the local study place when entering or leaving. (e.g Crowdness or Quality of Wi-Fi). For example, we can update the current information if a place is. (e.g. Full, Fairly Full, Half Full/Empty, Fairly Empty, Empty).

operationId: submitCrowdSourcing

parameters:

- name: city

in: path

description: Name of the location of the search

required: true

schema:

type: string

- name: study-place-id

in: path

description: ID of studyplace to return

required: true

schema:

type: integer

format: int64

responses:

'200':

description: successful operation

requestBody:

content:

application/json:

schema:

$ref: '#/components/schemas/StudyPlaceCrowdSourcingInfo'

description: order placed for purchasing the pet

required: true

/user/{id}/history/:

get:

tags:

- userProfileManager

summary: Get user search history

description: Retrieve a list of study places that the user has searched and clicked before.

operationId: getUserSearchHistory

parameters:

- name: id

in: path

description: ID of the user to retrieve search history for.

required: true

schema:

type: integer

format: int64

responses:

'200':

description: Successful operation

content:

application/json:

schema:

type: array

items:

$ref: '#/components/schemas/StudyPlace'

'400':

description: Invalid user ID supplied

'404':

description: User not found

/user/{id}/likedStudyPlaces/user/{id}/filters/:

get:

tags:

- userProfileManager

summary: Get user's liked Study Places

description: Retrieve a list of study places that the user has liked before.

operationId: getUserLikedStudyPlaces

parameters:

- name: id

in: path

description: ID of the user to retrieve liked studyplaces.

required: true

schema:

type: integer

format: int64

responses:

'200':

description: Successful operation

content:

application/json:

schema:

type: array

items:

$ref: '#/components/schemas/StudyPlace'

'400':

description: Invalid user ID supplied

'404':

description: User not found

/user/{id}/filters/:

get:

tags:

- userProfileManager

summary: Get user's defined filters

description: Retrieve a list of filters of the user.

operationId: getUserFilters

parameters:

- name: id

in: path

description: ID of the user to retrieve filters

required: true

schema:

type: integer

format: int64

responses:

'200':

description: Succesful operation

content:

application/json:

schema:

type: array

items:

$ref: '#/components/schemas/Filter'

put:

tags:

- userProfileManager

summary: Update user's defined filters

description: Update the values of the user's filters.

operationId: updateUserFilters

parameters:

- name: id

in: path

description: ID of the user to update filters

required: true

schema:

type: integer

format: int64

requestBody:

required: true

content:

application/json:

schema:

$ref: '#/components/schemas/Filter'

responses:

'200':

description: Successful operation

content:

application/json:

schema:

$ref: '#/components/schemas/Filter'

'404':

description: User not found

'500':

description: Internal server error

/user/{id}/account/:

get:

tags:

- userProfileManager

summary: Get user account info

description: Checks the user info, and if needed, if the user is a moderator

operationId: getUserAccount

parameters:

- name: id

in: path

description: ID of the user to return account info for

required: true

schema:

type: integer

format: int64

responses:

'200':

description: successful operation

content:

application/json:

schema:

$ref: '#/components/schemas/User'

'400':

description: Invalid user ID supplied

'404':

description: User not found

put:

tags:

- userProfileManager

summary: Create/Update User Profile

description: Create or update user profile with the new data. The account type can be a regular user, an owner, or a moderator.

operationId: updateUserProfile

parameters:

- name: id

in: path

description: User ID

required: true

schema:

type: integer

format: int64

- name: name

in: query

description: User's full name

required: true

schema:

type: string

- name: email

in: query

description: User's email address

required: true

schema:

type: string

format: email

- name: password

in: query

description: User's password

required: true

schema:

type: string

- name: phoneNumber

in: query

description: User's phoneNumber

required: false

schema:

type: string

- name: listOfFilters

in: query

description: User's preferences for study places.

required: false

schema:

type: array

items:

$ref: '#/components/schemas/Filter'

- name: userType

in: query

description: Type of the account- regular user, owner, or moderator

required: false

schema:

type: string

enum:

- Owner

- Admin

- Moderator

- User

responses:

'200':

description: User profile updated successfully

'400':

description: Invalid input provided

'404':

description: User not found

/studyPlaceSuggestions/:

post:

tags:

- studyPlaceSuggestion

summary: Submit a suggestion for a place to study

description:

The user uploads the name and if needed, the description of a study place. Uploads n (>=3) photos of that study place. Uploads the type of study place, like a coffee shop, park, library, or college room. The user also clicks on a checkbox to define the study place characteristics such as Good Wi-Fi, Medium Wi-Fi, Bad Wi-Fi, or No Wi-Fi. Other Examples would be Bar Service, No Bar Service. If the owner wants to be responsible for the StudyPlace information, he uploads his phone number for future validation of the place;

operationId: submitSuggestion

requestBody:

content:

application/x-www-form-urlencoded:

schema:

type: object

properties:

studyPlace:

$ref: '#/components/schemas/StudyPlaceSuggestion'

responses:

'201':

description: Suggestion created successfully.

'400':

description: Invalid input.

'500':

description: Internal server error.

get:

tags:

- studyPlaceSuggestion

summary: Gets All pending suggestions for validation.

description: Moderator/Admin downloads all the StudyPlaces Suggestion that need further validation.

operationId: getAllPendingSuggestions

responses:

'200':

description: successful operation

content:

application/json:

schema:

type: array

items:

$ref: '#/components/schemas/StudyPlaceSuggestion'

/studyPlaceSuggestions/{id}:

delete:

tags:

- studyPlaceSuggestion

summary: Deletes a suggestion of a StudyPlace.

description: The moderator/admin deletes an invalid user suggestion.

operationId: deleteSubmitSuggestion

parameters:

- name: id

in: path

description: id of the suggestion

required: true

schema:

type: string

responses:

'201':

description: Suggestion deleted successfully.

'400':

description: Invalid input.

'500':

description: Internal server error.

/studyPlaceSuggestions/validated:

post:

tags:

- studyPlaceSuggestion

summary: Moderator/Admin uploads to the microservice the suggestions validated.

description: Moderator/Admin uploads to the microservice the suggestions validated so it can be transformed into a "real" studyPlace.

operationId: validateSuggestion

requestBody:

content:

application/x-www-form-urlencoded:

schema:

type: object

properties:

studyPlace:

$ref: '#/components/schemas/StudyPlaceSuggestion'

responses:

'201':

description: Suggestion created successfully.

'400':

description: Invalid input.

'500':

description: Internal server error.

externalDocs:

description: Find out more about Swagger

url: 'http://swagger.io'

components:

schemas:

StudyPlace:

type: object

properties:

id:

type: integer

format: int64

name:

type: string

city:

type: string

coordinates:

type: object

properties:

latitude:

type: number

format: double

longitude:

type: number

format: double

photos:

type: array

items:

$ref: '#/components/schemas/Photo'

studyPlaceType:

type: string

filters:

type: array

items:

$ref: '#/components/schemas/Filter'

Photo:

type: object

properties:

url:

type: string

tag:

type: string

Filter:

type: object

properties:

type:

type: string

description: The type of the filter.

enum: [wifi, power\_outlets, bar Service, availability]

value:

type: string

description: The value of the filter.

enum:

- Perfect Wi-Fi

- Good Wi-Fi

- Reasonable Wi-Fi

- Bad Wi-Fi

- No Wi-Fi

- Excelent power outlets

- Good power outlets

- Enough power outlets

- Not enough power outlets

- No power outlets

- Bar Service

- No Service

- Full

- Fairly Full

- Half Full/Empty

- Fairly Empty

- Empty

- Noise

- No Noise

- Has Group Seating

- Has Outdoor Seating

- Has Accessibility (Elevado, Rampas);

User:

type: object

properties:

id:

type: integer

format: int64

username:

type: string

firstName:

type: string

lastName:

type: string

email:

type: string

password:

type: string

phone:

type: string

userProfilePhotoUrl:

$ref: '#/components/schemas/Photo'

filtersPreference:

type: array

items:

$ref: '#/components/schemas/Filter'

userType:

type: string

enum:

- Owner

- Admin

- Moderator

- User

studyPlacesIdHistory:

type: array

items:

type: integer

xml:

name: User

StudyPlaceSuggestion:

type: object

properties:

id:

type: integer

format: int64

studyPlace:

$ref: '#/components/schemas/StudyPlace'

userId:

type: integer

format: int64

status:

type: string

description: Describes the status of the submission. A reported submission was a malicious submission.

enum:

- Waiting for Validation

- Validated

- Declined

- Reported

submissionDate:

type: string

format: date-time

moderatorId:

type: integer

format: int64

xml:

name: User

StudyPlaceCrowdSourcingInfo:

type: object

properties:

id:

type: integer

format: int64

studyPlaceId:

type: integer

format: int64

userId:

type: integer

format: int64

filters:

type: array

description: User updates the wi-fi and the crowdness of the StudyPlace

items:

$ref: '#/components/schemas/Filter'

submissionDate:

type: string

format: date-time

Rating:

type: object

properties:

id:

type: integer

format: int64

studyPlaceId:

type: integer

format: int64

userId:

type: integer

format: int64

submissionDate:

type: string

format: date-time

filters:

type: array

items:

$ref: '#/components/schemas/Filter'

Review:

type: object

properties:

id:

type: integer

format: int64

studyPlaceId:

type: integer

format: int64

userId:

type: integer

format: int64

submissionDate:

type: string

format: date-time

reviewText:

type: string

requestBodies:

StudyPlace:

content:

application/json:

schema:

$ref: '#/components/schemas/StudyPlace'

application/xml:

schema:

$ref: '#/components/schemas/StudyPlace'

description: StudyPlace object that needs to be added to the store

required: true

Filter:

content:

application/json:

schema:

$ref: '#/components/schemas/Filter'

application/xml:

schema:

$ref: '#/components/schemas/Filter'

required: true

ListOfFilters:

content:

application/json:

schema:

type: array

items:

$ref: '#/components/schemas/Filter'

application/xml:

schema:

$ref: '#/components/schemas/Filter'

required: true

User:

content:

application/json:

schema:

$ref: '#/components/schemas/User'

application/xml:

schema:

$ref: '#/components/schemas/User'

required: true

StudyPlaceSuggestion:

content:

application/json:

schema:

$ref: '#/components/schemas/StudyPlaceSuggestion'

application/xml:

schema:

$ref: '#/components/schemas/StudyPlaceSuggestion'

required: true

StudyPlaceCrowdSourcingInfo:

content:

application/json:

schema:

$ref: '#/components/schemas/StudyPlaceCrowdSourcingInfo'

application/xml:

schema:

$ref: '#/components/schemas/StudyPlaceCrowdSourcingInfo'

required: true

Rating:

content:

application/json:

schema:

$ref: '#/components/schemas/Rating'

application/xml:

schema:

$ref: '#/components/schemas/Rating'

required: true

Review:

content:

application/json:

schema:

$ref: '#/components/schemas/Review'

application/xml:

schema:

$ref: '#/components/schemas/Review'

required: true

securitySchemes:

petstore\_auth:

type: oauth2

flows:

implicit:

authorizationUrl: 'http://petstore.swagger.io/oauth/dialog'

scopes:

'write:pets': modify pets in your account

'read:pets': read your pets