STUDY ON SAFE AND
SECURE PARKING
PLACES FOR TRUCKS
MOVE/C1/2017-500

API Design Guidelines

Arad, 19 June 2018

Contents

- Background & Motivation
- Context & Architecture
- Web services
- Functional requirements
- Non-functional requirements

Background & Motivation

- **Intended audience**: Safe and Secure parking providers
- **Why**: for designing Application Programming Interfaces (APIs) through which their services will be provided to mobile apps and other entities, for example online platforms and fleet management services.
- **APIs design**:
  - should enable authorised users to search, list, view details, as well as book, rate and comment on the parking lots.
  - facilitate integration with back office systems, platforms and devices (e.g. smartphones)
  - ensure its security and reliability and allow future updates.
  - Steer the way some user functions are defined and thus enable a more seamless user experience regardless of app provider.

Context & Architecture

- Safe and Secure parking providers should enable access to their systems using approaches common to other systems that make their services available over the Internet.

- **API** → a list of methods and classes exposed by a system or application which can be used by other system or application to access their functionalities.

- Parking service providers could make their service available for individual users and enable them to search and book a parking space

Parking provider ← API → Mobile app / Fleet Management System, ERP ...
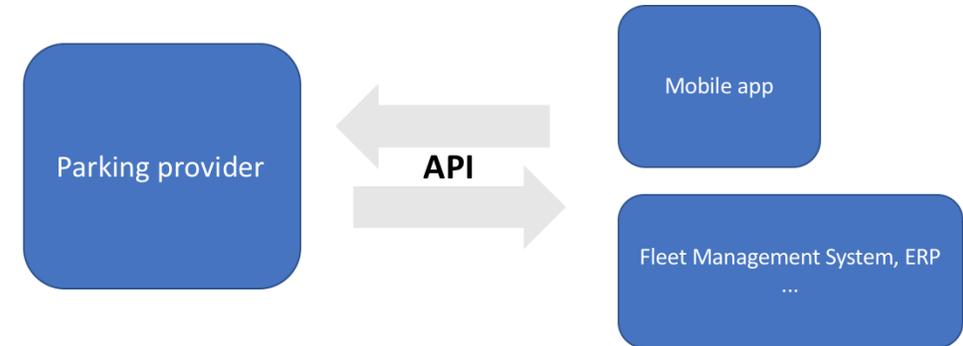
Web services

Main characteristics of an API:

- Easy to learn

- Easy to use, even without documentation

- Hard to misuse

- Easy to read and maintain code that uses it

- Sufficiently powerful to satisfy requirements

- Easy to extend

- Appropriate to audience

Functional requirements

Discovery

- Functionality provides listing of parking locations based on provided criteria.

- Several filters can be added to the search criteria simultaneously, and the results shall be provided as ordered lists.

- The supported search criteria shall include parking location characteristics of various types, e.g:
  - location based on provided GPS coordinates and distance (radius),
  - security rating (e.g. Bronze to Platinum)
  - comfort rating (e.g. 1-5)
  - user rating
  - included facilities
  - price

Parking Information & Rating

- Parking location details provided in a structured manner, with a mandatory set of characteristics and a set of optional ones. The details shall support text, as well as other media formats (e.g. photos).

- Mandatory information:
    - Location coordinates (GPS latitude and longitude)
    - Address, postcode, country
    - Phone number
    - Parking capacity
    - Currently available parking slots
    - Price
    - Currency
    - Security rating
    - Comfort rating
    - Last update (date)

- Optional information:
    - List of available facilities
    - Available fuels
    - User rating
    - User comments

Available facilities/services

- The list of facilities or services available at a selected parking area needs to correspond to an agreed and predefined (coded) categorisation of facilities in order to facilitate the visualization of the information in different languages. The services should also be listed separately for drivers, vehicle or general.

Driver:

- Restaurant / Coffee shop / buffet
- Hotel
- Toilets
- Showers
- Shop
- Lighting
- WiFi
- Medical first aid / assistance
- Cash machine (ATM)
- Camera surveillance
- Electric plugs
- Laundry
- ..

Vehicle:

- Fuel
- Charging
- Snow and ice removal equipment
- Spare parts
- Mechanical assistance
- Vehicle wash
- ..

General:

- Parking reservation and payment
- Capacity information
- Opening hours
- 24/7 security guard
- Camera surveillance
- Dangerous goods parking
- Oversized loads parking
- ..

Booking

- An authorised user shall be able to book a slot (or several) at a selected parking location. Limits need to be defined, such as maximum booking duration, earliest booking date, maximum arrival delay. After a booking request is made and payment is processed, the API shall respond with a booking confirmation.

Booking request:

- User (ID)

- Vehicle registration number

- Selected parking area

- Selected parking lot

- Desired parking start time (dd/mm/yyyy, hh:mm)

- Desired parking duration (in periods of 4, 12, 24 and 48 hours)

- Transport of dangerous goods (ADR, yes/no)

- Refrigerated (yes/no)

- Special requirements

Booking confirmation:

- User (ID)

- Booking ID

- Vehicle registration number

- Selected parking area

- Selected parking lot

- Parking start time (dd/mm/yyyy, hh:mm)

- Parking end time (dd/mm/yyyy, hh:mm)

- Latest parking arrival (dd/mm/yyyy, hh:mm)

- Transport of dangerous goods (ADR, yes/no)

- Special requirements

- Price paid

Other

Payment, cancellation and refunds

- Payment operations need to be provided through the API in order to support the booking functionality.

- Sufficient level of security and stability and a vast support of payment instruments need to be provided, together with a transparent process view, e.g.

  - transaction
  - status
  - error codes
  - status notifications

- Usually, the system is directed to an external service, such as PayPal or similar in order to process the payment.

- Same system should also provide refunds in certain cases.

User-generated content:

- API shall support user-generated content, from text to other media such as photo upload. E.g.:

  - Parking rating (e.g. satisfaction, information accuracy …)
  - Commenting
  - Photos upload
  - Corrections to the information

Social features:

- Enable interaction between different users taking into account their characteristics (e.g. nationality), current location or other preferences.

- Some of the functionalities could include:

  - Matching users of same nationalities, languages, interests
  - Make friends
  - Where to park / follow me (share booked parking info… even location sharing)

Driving time monitoring:

- This additional functionality could assist drivers in managing their driving time based on rest time provided by the booking information.

Non-functional requirements

Non-functional requirements

- Non-functional requirements describe the recommended approach to designing an API in a way to ensure easy integration, security, and use and concern:
  - Protocols (REST, SOAP)
  - Data formats (JSON, XML)
  - Security
  - Information accuracy and reliability
  - Authentication and authorisation
  - Documentation

Mulțumesc!

Oleh Shchuryk
IRU Projects
oleh.shchuryk@iruprojects.org