Web service specification v1

Contents

[Current User Points 2](#_Toc464335423)

[Description 2](#_Toc464335424)

[Parameters 2](#_Toc464335425)

[Request example 2](#_Toc464335426)

[Structure 2](#_Toc464335427)

[Videos filtered by tags 2](#_Toc464335428)

[Description 2](#_Toc464335429)

[Parameters 2](#_Toc464335430)

[Request example 2](#_Toc464335431)

[Structure 2](#_Toc464335432)

[Video comments 3](#_Toc464335433)

[Description 3](#_Toc464335434)

[Parameters 3](#_Toc464335435)

[Request example 3](#_Toc464335436)

[Structure 3](#_Toc464335437)

[Add point and/or view 4](#_Toc464335438)

[Description 4](#_Toc464335439)

[Parameters 4](#_Toc464335440)

[Request example 4](#_Toc464335441)

[Structure 4](#_Toc464335442)

[Add comment to video 5](#_Toc464335443)

[Description 5](#_Toc464335444)

[Parameters 5](#_Toc464335445)

[Request example 5](#_Toc464335446)

[Structure 5](#_Toc464335447)

[Manage video like 5](#_Toc464335448)

[Description 5](#_Toc464335449)

[Parameters 6](#_Toc464335450)

[Request example 6](#_Toc464335451)

[Structure 6](#_Toc464335452)

[Update user information 6](#_Toc464335453)

[Description 6](#_Toc464335454)

[Parameters 6](#_Toc464335455)

[Request example 6](#_Toc464335456)

[Structure 7](#_Toc464335457)

[Log in user 8](#_Toc464335458)

[Description 8](#_Toc464335459)

[Parameters 8](#_Toc464335460)

[Request example 8](#_Toc464335461)

[Structure 8](#_Toc464335462)

[Login succeeded 8](#_Toc464335463)

[Login falied – login information incorrect, deleted user, blocked user, deactivated user (check each one) 8](#_Toc464335464)

[All geographic units 9](#_Toc464335465)

[Description 9](#_Toc464335466)

[Parameters 9](#_Toc464335467)

[Request example 9](#_Toc464335468)

[Structure 9](#_Toc464335469)

[Register user to app 10](#_Toc464335470)

[Description 10](#_Toc464335471)

[Parameters 10](#_Toc464335472)

[Structure 10](#_Toc464335473)

# Current User Points

## Description

Web service returns current user points for telecommunication operator to use that information for discount and benefits for user in their webshop.

## Parameters

user\_id – active user's id currently logged in

telecomm\_operator – active telecomm operator for logged-in user

## Request example

GET localhost/webservice/current\_user\_points?user\_id=1&telecomm\_operator=2

## Structure

{

"status":100,

"requested\_at":"<date and time>",

"data":{

"user\_id\_remote":"<user\_id\_remote>",

"points": ##

}

}

# Videos filtered by tags

## Description

This webservice returns videos and likes for user to choose which video he wants to watch. Those videos are filtered by tags.

## Parameters

gender - logged-in user's gender

birthday/age – logged-in user's birthday date or calculated age (dvoumim se)

geographic\_unit – logged-in user's geographic unit

## Request example

GET localhost/webservice/videos?gender=M&birthday=1992.20.10&geographic\_unit=2

GET localhost/websevice/videos?gender=M&age=23&geographic\_unit=2

## Structure

{

"status":100,

"requested\_at":"<date and time>",

"data":{

[

{

"id":###,

"title":"<title>",

"link":"<link>",

"likes":###,

"dislikes":###,

"points":###,

"created\_at":"<created\_at>"

},

{

"id":###,

"title":"<title>",

"link":"<link>",

"likes":###,

"dislikes":###,

"points":###,

"created\_at":"<created\_at>"

}

]

}

}

# Video comments

## Description

Web service which returns all comments for video that is watched at least one time.

## Parameters

video – watched video's id

## Request example

GET localhost/webservice/video\_comments?video=1

## Structure

{

"status":100,

"requested\_at":"<date and time>",

"data":{

[

{

"id":###,

"author":{

"name":"<name>",

"surname":"<surname>"

},

"content":"<content>",

"created\_at":"date and time of creation"

},

{

"id":###,

"author":{

"name":"<name>",

"surname":"<surname>"

},

"content":"<content>",

"created\_at":"date and time of creation"

}

]

}

}

# Add point and/or view

## Description

Web service adds point and/or view. If user hasn't seen the video yet, points are added to user and video view too. If video has one view, extra view is added to user's video views.

## Parameters

video – video's id that is seen

user – user's id that is logged-in currenty

## Request example

GET localhost/webservice/view\_video?video=1&user=3

## Structure

{

"status":100,

"requested\_at":"<date and time>",

"data":{

"success":##,

"message": "<message>"

}

}

# Add comment to video

## Description

Web service adds new comment to video that is currently open.

## Parameters

video – video's id that is currently open

author – user's id that wrote comment

content – comment's content that shall be added

## Request example

GET localhost/webservice/comment\_video?video=1&author=1&content=This is my comment

## Structure

{

"status":100,

"requested\_at":"<date and time>",

"data":{

"success":##,

"message": "<message>"

}

}

# Manage video like

## Description

Web service that updates user's like status.

## Parameters

video – video's id that is seen

user – user's id that is logged-in currenty

like - current like value

## Request example

GET localhost/webservice/videolike?video=1&user=1&like=-1

## Structure

{

"status":100,

"requested\_at":"<date and time>",

"data":{

"success":##,

"message": "<message>"

}

}

# Update user information

## Description

Web service to update user information

## Parameters

id – user's id that is logged-in currenty

name - user's name

surname – user's surname

birthday – user's birthday

gender – user's gender

email – user's email

username – user's username

password – user's password

geo\_unit – geographic unit's id

## Request example

POST localhost/webservice/updateuser with all parameters and password hashed (mislim si da bi pass izostavili ako se ne mijenja )

NOTICE: use GET method to test web service , POST IS REQUIRED

## Structure

{

"status":100,

"requested\_at":"<date and time>",

"data":{

"success":##,

"message": "<message>"

}

}

# Log in user

## Description

Web service that checks login info for user and returns its information.

## Parameters

username – username for user to be logged in

password - hashed password for user to be logged in

## Request example

POST localhost/webservice/loggin?username=username&password=password

## Structure

### Login succeeded

{

"status":100,

"requested\_at":"<date and time>",

"data":{

"id":##,

"name": "<name>",

"surname": "<surname>",

"birthday": "<birthday>",

"gender": "<gender>",

"email": "<email>",

"username": "<username>",

"password": "<password>",

"password\_changed\_at": "<password\_changed\_at>",

"geographic\_unit": {

"id": ###,

"name": "<name>"

}

}

}

### Login falied – login information incorrect, deleted user, blocked user, deactivated user (check each one)

{

"status":100,

"requested\_at":"<date and time>",

"data":{

"message": "<message>"

}

}

# All geographic units

## Description

Web service returns all geographic units.

## Parameters

None

## Request example

GET localhost/webservice/geographic\_units

## Structure

{

"status":100,

"requested\_at":"<date and time>",

"data":[

{

"id":###,

"name": "<name>"

},

{

"id":###,

"name": "<name>"

}

]

}

# Register user to app

## Description

Web service for registering users to application.

## Parameters

name - user's name

surname – user's surname

birthday – user's birthday

gender – user's gender

email – user's email

username – user's username

password – user's password

geo\_unit – geographic unit's id

## Structure

{

"status":100,

"requested\_at":"<date and time>",

"data":{

"success":##,

"message": "<message>"

}

}