

# **Documentacion App Server**

**version 0.2.0**

**Grupo 5**

**June 17, 2016**



# Contents

<b>75.52 Taller de Programación Grupo 5 App Server</b>	<b>1</b>
API Documentation	1
Login (Authorization)	1
GET	1
Parametros	1
Respuesta	1
Ejemplo de la respuesta:	1
Users	1
GET ALL	1
Parámetros	2
Respuesta	2
Ejemplo de la respuesta:	2
POST	2
Parámetros	2
Respuesta	3
Ejemplo de la respuesta:	3
GET SPECIFIC	3
Parámetros	3
Respuesta	3
PUT	4
Parámetros	4
Respuesta	4
DELETE	5
Parámetros	5
Respuesta	5
User Matches	5
GET	5
Parámetros	5
Respuesta	5
Ejemplo del payload en la respuesta	5
User Likes	6
POST	6
Parámetros	6
Respuesta	6
Ejemplo de respuesta	6
Messages	6
GET	6
Parámetros	6
Respuesta	6
Ejemplo de respuesta	7

POST	7
Parámetros	7
Respuesta	7
Ejemplo de response:	7
Conversations	8
GET	8
Parámetros	8
Ejemplo respuesta:	8
Candidates	8
GET	8
Parámetros	9
Respuesta	9
Ejemplo de payload de la respuesta	9
<b>Indices and tables</b>	<b>9</b>

# 75.52 Taller de Programación Grupo 5 App Server

Contents:

## API Documentation

En esta sección estará documentada la REST API del App Server

### Login (Authorization)

URI del recurso /users/login

#### GET

Realizar pedido de autenticación para usar ciertos servicios de la API que lo requieren. El correcto acceso de login (con un usuario y contraseña validos) devuelve un token el cual puede ser usado en otras llamadas que necesiten permiso de un usuario registrado.

#### Parametros

Por header via Basic Auth: *Ejemplo* "Authorization" : "Basic base64\_encode(username:password)"

#### Respuesta

- errorNum: código de error (0 indica success)
  - message: mensaje asociado al código de error
  - metadata: Información de metadata sobre la respuesta
1. size: Cantidad de elementos devueltos en el payload
- payload:
1. result (**OK** en caso satisfactorio, **FAIL** En caso erroneo)
  2. token (token para usar en otros servicios en caso de volver con result **OK**)

#### Ejemplo de la respuesta:

```
{
  "errorNum": 0,
  "message": "Success",
  "metadata": {
    "size": 2
  },
  "payload": {
    "result": "OK",
    "token": "as54das564dasd45s64da6s4d="
  }
}
```

## Users

URI del recurso /users

#### GET ALL

Obtener información sobre todos los usuarios.

**Parámetros**

No toma parametros

**Respuesta**

- errorNum: código de error (0 indica success)
  - message: mensaje asociado al código de error
  - metadata: Información de metadata sobre la respuesta
1. size: Cantidad de elementos devueltos en el payload
- payload: Elementos pertinentes a devolver (si no necesita devolver nada éste esta vacio)
1. id
  2. name
  3. alias
  4. email
  5. photo\_profile
  6. interests
  7. location

**Ejemplo de la respuesta:**

```
{
  "errorNum": 0,
  "message": "",
  "metadata": {
    "size": 1
  },
  "payload": {
    "Users": [
      {
        "latitude": 0,
        "longitude": 0,
        "matches": [],
        "name": "juan",
        "password": "hola",
        "perfilImage": "",
        "token": "juanmahola",
        "username": "juanma"
      }
    ]
  }
}
```

**POST**

Dar de alta a un usuario nuevo

**Parámetros**

**username, name, password** Opcionales: **location, interests, age, alias, sex, distance**

Ejemplo:

```
{ "username": "juanma", "name": "juan", "password": "hola", "sex": "M" }
```

Sex puede ser "M" o "F"

**Respuesta**

- errorNum: código de error (0 indica success)
  - message: mensaje asociado al código de error
  - metadata: Información de metadata sobre la respuesta
1. size: Cantidad de elementos devueltos en el payload
- payload: Elementos pertinentes a devolver (si no necesita devolver nada éste esta vacío)

0

**Ejemplo de la respuesta:**

```
{
  "errorNum": 0,
  "message": "Registered",
  "metadata": {
    "size": 0
  },
  "payload": 0
}
```

**GET SPECIFIC**

Obtener usuario específico por su username

**Parámetros**

**username:** ID del usuario (mail del usuario)

**Respuesta**

- errorNum código de error (0 indica success)
  - message mensaje asociado al código de error
- metadata: Información de metadata sobre la respuesta
1. size: Cantidad de elementos devueltos en el payload
- payload: Elementos pertinentes a devolver (si no necesita devolver nada éste esta vacío)
1. id
  2. name
  3. alias
  4. email
  5. photo\_profile
  6. interests
  7. location

**Ejemplo de payload!**

```
{
  "id": 1,
  "name": "usuario",
  "alias": "not a user",
  "email": "usuario@usuario.com",
  "photo_profile": "< base64 >",
  "sex": "M",
  "interests": [
    { "category": "music/band", "value": "michael jackson" },
  ]
}
```

```

    { "category": "music/band", "value": "pearl jam" },
    { "category": "outdoors", "value": "running" }
  ],
  "location": { "latitude": -121.45356, "longitude": 46.51119 },
  "disantece": 10,
  "ageRange": {
    "min": 20,
    "max": 26
  }
}

```

## PUT

Editar usuario específico /users/username

## Parámetros

**username:** ID del usuario (mail del usuario) Requerido

Opcionales: **location, interests, age, alias, sex, distance, name, password, ageRange**

## Respuesta

- errorNum: código de error (0 indica success)
  - message: mensaje asociado al código de error
  - metadata: Información de metadata sobre la respuesta
1. size: Cantidad de elementos devueltos en el payload
- payload: Elementos pertinentes a devolver (si no necesita devolver nada éste esta vacío)
1. id
  2. name
  3. alias
  4. email
  5. photo\_profile
  6. interests
  7. location

*Ejemplo de PUT parametros de body de PUT request:*

```

{
  "username": "username",
  "name": "usuario",
  "alias": "not a user",
  "email": "usuario@usuario.com",
  "sex": "M",
  "age": 30,
  "photo_profile": "< base64 >",
  "interests": [{
    "category": "music / band",
    "value": "radiohead"
  }, {
    "category": "music / band",
    "value": "pearl jam"
  }, {
    "category": "outdoors",
    "value": "running"
  }
]
}

```



```

    }],
    "location": {
      "latitude": -121.45356,
      "longitude": 46.51119
    },
    "distance": 10,
    "ageRange": {
      "min": 19,
      "max": 26
    }
  }
}

```

## DELETE

Eliminar usuario específico por username

### Parámetros

**username:** ID del usuario (mail del usuario)

### Respuesta

- errorNum: código de error (0 indica success)
- message: mensaje asociado al código de error
- metadata: Información de metadata sobre la respuesta

1. size: Cantidad de elementos devueltos en el payload - payload: Elementos pertinentes a devolver (si no necesita devolver nada éste está vacío) 1. id 2. name 3. alias 4. email 5. photo\_profile 6. interests 7. location

*Ejemplo de body de Delete request*

```

{
  "username": "ff5"
}

```

## User Matches

**URI del recurso** /users/getMatches

## GET

Obtiene los matches ocurridos para el user

### Parámetros

**username:** ID del usuario (mail del usuario)

### Respuesta

- errorNum: código de error (0 indica success)
- message: mensaje asociado al código de error
- metadata: Información de metadata sobre la respuesta

1. size: Cantidad de elementos devueltos en el payload - payload: Elementos pertinentes a devolver (si no necesita devolver nada éste está vacío) 1. matches (Json array con usernames)

### Ejemplo del payload en la respuesta

```
{
  "matches": [
    "username1",
    "username2",
    "username3"
  ]
}
```

## User Likes

### POST

Enviar aceptación (like) o rechazo (not like) de un usuario "candidato".

**URI del recurso** /likes

#### Parámetros

user1 - ID del usuario que envía si acepta o rechaza (username) user2 - ID del usuario, aceptado o rechazado (username) like - Aceptación o rechazo (boolean)

**Ejemplo de body request:**

```
{
  "user1": "username1",
  "user2": "username2",
  "like": true
}
```

#### Respuesta

errorNum - código de error (0 indica success) message - mensaje asociado al código de error metadata: - Información de metadata sobre la respuesta 1. size - Cantidad de elementos devueltos en el payload payload: - Elementos pertinentes a devolver (si no necesita devolver nada éste está vacío) 0

#### Ejemplo de respuesta

```
{
  "errorNum": 0,
  "message": "Like Saved",
  "metadata": {
    "size": 0
  },
  "payload": 0
}
```

## Messages

**URI del recurso:** /messages

### GET

Obtener todos los mensajes que existen

#### Parámetros

No

#### Respuesta

errorNum - código de error (0 indica success) message - mensaje asociado al código de error metadata: - Información de metadata sobre la respuesta 1. size - Cantidad de elementos devueltos en el payload payload: - Elementos pertinentes a devolver (si no necesita devolver nada éste esta vacío) 1. messages (Json array con objetos json con lo siguiente) - id (del mensaje) - sender (quien envió el mensaje, username1) - receptor (quien recibió el mensaje, username2) - date (fecha del mensaje) - data (contenido del mensaje)

### Ejemplo de respuesta

```
{
  "errorNum": 0,
  "message": "",
  "metadata": {
    "size": 1
  },
  "payload": {
    "messages": [
      {
        "id": 10,
        "sender": "username1",
        "receptor": "username2",
        "date": "00:32:44 - 2016/06/01.",
        "data": "como estas?!"
      },
      {
        "id": 15,
        "sender": "username2",
        "receptor": "username1",
        "date": "00:33:35 - 2016/06/01.",
        "data": "todo tranqui vos?"
      }
    ]
  }
}
```

## POST

Enviar un mensaje en cierta conversación entre usuarios

### Parámetros

user1 - ID del usuario (username) user2 - ID del usuario (username) data - mensaje a enviar

### Ejemplo de body request

```
{
  "user1": "username1",
  "user2": "username2",
  "data": "hola"
}
```

### Respuesta

Content-Type: application/json errorNum - código de error (0 indica success) message - mensaje asociado al código de error metadata: - Información de metadata sobre la respuesta 1. size - Cantidad de elementos devueltos en el payload payload: - Elementos pertinentes a devolver (si no necesita devolver nada éste esta vacío)

### Ejemplo de response:

```
{
  "errorNum": 0,
  "message": "Message Saved",
  "metadata": {
    "size": 1
  },
  "payload": {
    "messages": [
      {
        "id": 10,
        "sender": "username1",
        "receptor": "username2",
        "date": "00:32:44 - 2016/06/01.",
        "data": "como estas?!"
      },
      {
        "id": 15,
        "sender": "username2",
        "receptor": "username1",
        "date": "00:33:35 - 2016/06/01.",
        "data": "todo tranqui vos?"
      }
    ]
  }
}
```

```

"metadata": {
  "size": 0
},
"payload": 0
}

```

## Conversations

**URI del recurso** /conversations

### GET

Devuelve la conversación entre dos usuarios

### Parámetros

user1 - ID del usuario host (username) user2 - ID del usuario guest (username)

Respuesta

errorNum - código de error (0 indica success) message - mensaje asociado al código de error metadata: - Información de metadata sobre la respuesta 1. size - Cantidad de elementos devueltos en el payload payload: - Elementos pertinentes a devolver (si no necesita devolver nada éste esta vacío) 1. messages - Json Array con los mensajes de la conversación. Estos mensajes contienen: - id - sender - receptor - date - data

### Ejemplo respuesta:

```

{
  "errorNum": 0,
  "message": "",
  "metadata": {
    "size": 1
  },
  "payload": {
    "messages": [
      {
        "data": "buenas",
        "date": "00:32:44 - 2016/05/31.",
        "id": "9",
        "receptor": "5",
        "sender": "4"
      },
      {
        "data": "hola",
        "date": "00:39:19 - 2016/05/31.",
        "id": "11",
        "receptor": "5",
        "sender": "4"
      }
    ]
  }
}

```

## Candidates

**URI del recurso** /candidates

### GET

Obtener candidatos para un usuario específico

**Parámetros**

username - ID del usuario (mail del usuario)

**Respuesta**

errorNum - código de error (0 indica success) message - mensaje asociado al código de error metadata: - Información de metadata sobre la respuesta 1. size - Cantidad de elementos devueltos en el payload payload: - Elementos pertinentes a devolver (si no necesita devolver nada éste esta vacío) 0. candidates: es un arrayJson que contiene varios usuarios candidatos con lo siguiente: 1. id 2. name 3. alias 4. email 5. photo\_profile 6. interests (en común) 7. distance (en km) 8. sex (M o F) 9. age

**Ejemplo de payload de la respuesta**

```
{
  "candidates": [
    {
      "id": 1,
      "name": "usuario",
      "alias": "not a user",
      "email": "usuario@usuario.com",
      "photo_profile": "< base_64 >",
      "sex": "M",
      "age": 20,
      "interests": [
        {"category": "music/band", "value": "michael jackson"},
        {"category": "music/band", "value": "pearl jam"},
        {"category": "outdoors", "value": "running"}
      ],
      "distance": 10.0
    }
  ]
}
```

**Indices and tables**

- `genindex`
- `modindex`
- `search`