Authentication Endpoints (v1)

NOTE: Each API request must be authorized with API key (unless particular endpoint's name says that it's public). In order to authorize your requests "Authorization" header must be set and should contain a valid API key.

Check Key (public)

Checks if specified api key is valid

- **URL**

  /api/v1/auth/check/key

- **Method:**

  `POST`

- **Data Params**

  Required:

  `key=[string]`

- **Response Examples:**

  - **Code:** 200
    Content:

```
{
  "msg": "API key is valid",
  "error": 0,
  "user": {
    "id": 1,
    "email": "email@gmail.com",
    "first_name": "John",
    "last_name": "Doe",
    "is_active": 0,
    "last_login": "2017-05-03 17:30:17",
    "subscription_start": "2017-02-11 01:40:49",
    "subscription_end": "2027-02-11 01:40:49",
    "rollback_level": "users",
    "created_at": "2017-02-14 05:40:02",
    "updated_at": "2017-05-03 17:30:17",
    "roles": [
      {
        "id": 1,
        "slug": "users",
        "name": "Users",
        "permissions": {
          "limit_accounts": "1000",
          "limit_posts": "1000",
          "limit_campaigns": "1000",
          "admin": true,
          "recurring": false,
          "recurring_period": 0,
          "allow_tiered": true,
          "allow_sharing": true
        },
        "created_at": "2017-02-08 22:56:34",
        "updated_at": "2017-04-21 03:38:49",
      }
    ]
  }
}
```

- **Code:** 401
  Content: `{"msg":"Key is invalid","error": 1}`

Check Credentials

Checks if specified credentials are matching our records

- **URL**

  /api/v1/auth/check/credentials

- **Method:**

  `POST`

- **Data Params**

  Required:

  ```
  email=[string]
  password=[string]
  ```

- **Response Examples:**

  - **Code:** 200
    Content:

```
{
  "msg": "Credentials are valid",
  "error": 0,
  "user": {
    "id": 1,
    "email": "email@gmail.com",
    "first_name": "John",
    "last_name": "Doe",
    "is_active": 0,
    "last_login": "2017-05-03 17:30:17",
    "subscription_start": "2017-02-11 01:40:49",
    "subscription_end": "2027-02-11 01:40:49",
    "rollback_level": "users",
    "created_at": "2017-02-14 05:40:02",
```

```
    "updated_at": "2017-05-03 17:30:17",
    "roles": [
      {
        "id": 1,
        "slug": "users",
        "name": "Users",
        "permissions": {
          "limit_accounts": "1000",
          "limit_posts": "1000",
          "limit_campaigns": "1000",
          "admin": true,
          "recurring": false,
          "recurring_period": 0,
          "allow_tiered": true,
          "allow_sharing": true
        },
        "created_at": "2017-02-08 22:56:34",
        "updated_at": "2017-04-21 03:38:49",
      }
    ]
  }
}
```

- **Code:** 401
  Content: `{"msg":"Bad credentials","error": 1, "user": []}`

Accounts Endpoints (v1)

List

Get all accounts belonging to currently authorized user

- **URL**

  /api/v1/accounts/list

- **Method:**

  `GET`

- **Data Params**

  `None`

- **Response Example:**

  - **Code:** 200
    Content:

```
{
  "msg": "Ok ",
  "error": 0,
  "data": [
    {
      "id": 79,
      "owner_id": 1,
      "service_id": 4,
      "name": "Twitter Account",
      "post_to": null,
      "login": null,
      "password": null,
      "secret": "McwAS1tsdCK2ARjSq5YSjS4gQjEjVX5oEZaKwjc3r5ObI",
      "access_token": "2064670406-F3MQtVbfxsOzYjfwefzn9uCSqtbYgUnC1ju9hHC",
      "refresh_token": null,
      "token_expires": "2027-03-03 00:28:05",
      "is_active": 1,
      "created_at": "2017-03-03 00:27:58",
      "updated_at": "2017-03-09 04:41:46",
 "service": {
        "id": 4,
        "name": "Twitter",
        "type": "social",
        "auth_type": "oauth",
        "post_to_required": 0,
        "post_to_type": null,
        "class": "App\\Engine\\Services\\Twitter",
        "is_active": 1,
        "selfhosted": 0,
        "updated_at": "2017-03-31 14:57:26",
        "created_at": "2017-02-12 00:00:00"
      }
    }
  ]
}
```

Campaigns Endpoints (v1)

Campaign Creation Process

Campaign creation consists of 3 separate steps.

Step 1

Create campaign using "create" endpoint below and store returned campaign id.

Step 2

Use "fields" endpoint to get list of all fields that's required for this particular campaign (optional fields list can be found under "finalize" endpoint)

Step 3

Complete campaign creation process by making request to "finalize" endpoint with all the fields we've retrieved on the previous step properly filled.

List

Fetches all campaigns that belongs to currently authorized user

- **URL**

  /api/v1/campaigns/list

- **Method:**

  `GET`

- **Data Params**

  `None`

- **Response Examples:**

  - **Code:** 200
    Content:

```
{
  "msg": "Ok",
  "error": 0,
  "data": [
    {
      "id": 37,
      "owner_id": 1,
      "name": "Test Campaign",
      "starts_at": null,
      "target": "group",
      "target_id": 23,
      "dripping": null,
      "dripping_limit": null,
      "randomized_limit": null,
      "randomized_at": null,
      "is_paused": 0,
      "is_locked": 0,
      "is_completed": 0,
      "is_failed": 0,
      "updated_at": "2017-04-18 02:50:17",
      "created_at": "2017-04-18 02:50:17"
    }
```

- **Code:** 200
  Content:

```
{
  "msg": "No campaigns found",
  "error": 0,
  "data": []
}
```

Create

Create a new syndication campaign

- **URL**

  /api/v1/campaigns/create

- **Method:**

  `POST`

- **Data Params**

  Required:

  `name=[string]`
  `target=[group|account|tiers]`

  `target_id=[int]` *(id of group or account you want to assign to the campaign, not applicable for tiered campaigns*

  `groups=[array]` *(array of group ids order of which represents tiers order, only applicable for `target=tiers`)*

  Optional:

  `ignore=[video|social|blog|bookmark]` *(skip accounts of specified type on campaign creation)*
  `dripping=[per_day|days_total|weeks_total]`

  `dripping_limit=[int]` *(not applicable for `dripping=per_day`)*

  `dripping_limit_from=[int]` *(only applicable for `dripping=per_day`)*

  `dripping_limit_to=[int]` *(only applicable for `dripping=per_day`)*

- **Response Examples:**

  - **Code:** 200
    Content:

```
{
  "msg": "Campaign was successfully created",
  "error": 0,
  "errors": [],
  "data": {
    "id": 38
  }
}
```

- **Code:** 401
  Content:

```
{
  "msg": "Validation failed",
  "error": 1,
  "errors": {
    "target": [
      "The target field is required."
    ],
    "target_id": [
      "The target id field is required."
    ]
  },
  "data": []
}
```

Types

Get all post types used in specified campaign

- **URL**

  /api/v1/campaigns/:id/types

- **Method:**

  `GET`

- **Data Params**

  `None`

- **Response Example:**

  - **Code:** 200
    Content:

```
{
  "msg": "Ok",
  "error": 0,
  "data": [
    "blog",
    "social"
  ]
}
```

Fields

Get all fields required to finalize specified campaign

- **URL**

  /api/v1/campaigns/:id/fields

- **Method:**

  `GET`

- **Data Params**

  `None`

- **Response Example:**

    - **Code:** 200
      Content:

```
{
  "msg": "Ok",
  "error": 0,
  "data": [
    "title_blog",
    "content_blog",
    "tags_blog"
  ]
}
```

Finalize

Last step in campaign creation process. Finalize campaign creation by supplying all the fields required by this specific campaign's services

- **URL**

  /api/v1/campaigns/:id/finalize

- **Method:**

  `POST`

- **Data Params**

  Required:

  `file_video=[file]` *(for campaigns containing accounts with type of "video")*
  `title_video=[string]` *(for campaigns containing accounts with type of "video")*
  `content_video=[string]` *(for campaigns containing accounts with type of "video")*
  `tags_video=[string]` *(comma separated, for campaigns containing accounts with type of "video")*
  `category_video=[string]` *(for campaigns containing accounts with type of "video")*
  `title_blog=[string]` *(for campaigns containing accounts with type of "blog")*
  `content_blog=[string]` *(for campaigns containing accounts with type of "blog")*
  `tags_blog=[string]` *(comma separated, for campaigns containing accounts with type of "blog")*
  `content_social=[string]` *(for campaigns containing accounts with type of "social")*
  `content_bookmark=[string]` *(for campaigns containing accounts with type of "bookmark")*
  `tags_bookmark=[string]` *(comma separated, for campaigns containing accounts with type of "bookmark")*
  `title_bookmark=[string]` *(for campaigns containing accounts with type of "bookmark")*
  `url_bookmark=[string]` *(for campaigns containing accounts with type of "bookmark")*

  Optional:

  `starts_at=[datetime]` *(no need to use it if you want campaign to start right away)*
  `image_blog=[file]` *(for campaigns containing accounts with type of "blog")*

`image_bookmark=[file]` *(for campaigns containing accounts with type of "bookmark")*

`video_embed_blog` *(for campaigns containing accounts with type of "blog")*

- **Response Example:**

  - **Code:** 200
    Content:

```
{
  "msg": "Campaign was successfully finalized",
  "error": 0,
  "errors": [],
}
```

Edit

Edit specified existing campaign

- **URL**

  /api/v1/campaigns/:id/edit

- **Method:**

  `POST`

- **Data Params**

  Optional:

  `starts_at=[datetime]` *(can be changed only if campaign has not started yet)*
  `name=[string]`

- **Response Example:**

  - **Code:** 200
    Content:

```
{
  "msg": "Campaign was successfully updated",
  "error": 0,
  "errors": [],
}
```

Get

Get all the information about specified campaign (including accounts, files and posts)

- **URL**

  /api/v1/campaigns/:id/get

- **Method:**

  `GET`

- **Data Params**

  None

- **Response Example:**

  - **Code:** 200
    Content:

```
{
  "msg": "Ok",
  "error": 0,
  "data": {
    "status": "Required fields are missing",
    "campaign": {
      "id": 38,
      "owner_id": 1,
      "name": "Api Campaign",
      "starts_at": null,
      "target": "account",
      "target_id": 83,
      "dripping": null,
      "dripping_limit": null,
      "randomized_limit": null,
      "randomized_at": null,
      "is_paused": 0,
      "is_locked": 0,
      "is_completed": 0,
      "is_failed": 0,
      "updated_at": "2017-04-18 06:20:13",
      "created_at": "2017-04-18 06:20:13",
      "accounts": [
        {
          "id": 83,
          "owner_id": 1,
          "service_id": 7,
          "name": "Wordpress.org Account",
          "post_to": "http://localhost/wordpress",
          "login": "user",
          "password": "user",
          "secret": null,
          "access_token": null,
          "refresh_token": null,
          "token_expires": null,
          "is_active": 1,
          "created_at": "2017-03-03 01:43:28",
          "updated_at": "2017-03-09 05:04:46",
          "pivot": {
            "campaign_id": 38,
            "account_id": 83,
            "success": 0,
            "attempts": 0,
            "attempted_at": null
          }
        }
      ]
    },
    "accounts_syndicated": [],
    "accounts_pending": [],
    "accounts_failed": [],
    "total_syndications_count": 1,
    "successful_syndications_count": 0,
    "pending_syndications_count": 0,
    "failed_syndications_count": 0,
    "posts": [],
    "files": []
  }
}
```

Pause

Pauses specified campaign

- **URL**

  /api/v1/campaigns/:id/pause

- **Method:**

  `GET`

- **Data Params**

  None

- **Response Example:**

  - **Code:** 200
    Content:

```
{
  "msg": "Campaign is now paused",
  "error": 0
}
```

Resume

Resume specified campaign

- **URL**

/api/v1/campaigns/:id/resume

- **Method:**

  `GET`

- **Data Params**

  None

- **Response Example:**

  - **Code:** 200
    Content:

```
{
  "msg": "Campaign successfully resumed",
  "error": 0
}
```

Delete

Delete specified campaign

- **URL**

  /api/v1/campaigns/:id/delete

- **Method:**

  `GET`

- **Data Params**

  None

- **Response Example:**

  - **Code:** 200
    Content:

```
{
  "msg": "Campaign was successfully deleted",
  "error": 0
}
```

Groups Endpoints (v1)

List

Fetches all groups (and their assigned accounts) that belongs to currently authorized user

- **URL**

  /api/v1/groups/list

- **Method:**

  `GET`

- **Data Params**

  `None`

- **Response Examples:**

  - **Code:** 200
    Content:

```
{
  "msg": "Ok",
  "error": 0,
  "data": [
    {
      "id": 1,
      "name": "Group 1",
      "owner_id": 1,
      "updated_at": "2017-04-16 06:40:59",
      "created_at": "2017-03-11 23:43:26",
      "accounts": []
    },
    {
      "id": 2,
      "name": "Group 2",
      "owner_id": 1,
      "updated_at": "2017-04-11 03:25:48",
      "created_at": "2017-04-11 03:25:48",
      "accounts": []
    }
  ]
}
```

- **Code:** 200
  Content:

```
{
  "msg": "User doesn't have groups",
  "error": 0,
  "data": []
}
```

Create

Create a new accounts group

- **URL**

  /api/v1/groups/create

- **Method:**

  `POST`

- **Data Params**

  Required:

  `name=[string]`

- **Response Example:**

  - **Code:** 200
    Content:

```
{
  "msg": "Group successfully created",
  "error": 0,
  "errors": [],
  "data": {
    "id": 30
  }
}
```

Edit

Edit specified group

- **URL**

  /api/v1/groups/:id/edit

- **Method:**

  `POST`

- **Data Params**

  Required:

  `name=[string]`

- **Sample Response:**

  - **Code:** 200
    Content:

```
{
  "msg": "Group successfully updated",
  "error": 0,
  "errors": []
}
```

Get

Fetches specified group along with its assigned accounts

- **URL**

  /api/v1/groups/:id/get

- **Method:**

  `GET`

- **Sample Response:**

  - **Code:** 200
    Content:

```
{
  "msg": "Ok",
  "error": 0,
  "data": {
    "group": {
      "id": 30,
      "name": "Group 1",
      "owner_id": 1,
      "updated_at": "2017-04-18 05:23:03",
      "created_at": "2017-04-18 05:23:03",
      "accounts": []
    },
    "accounts": []
  }
}
```

Delete

Deletes specified group

- **URL**

  /api/v1/groups/:id/delete

- **Method:**

  `GET`

- **Sample Response:**

  - **Code:** 200
    Content:

```
{
  "msg": "Group was successfully deleted",
  "error": 0
}
```

Account Assign

Assign account to group

- **URL**

  /api/v1/groups/:id/account/assign

- **Method:**

  `POST`

- **Data Params**

  Required:

  `account_id=[int]`

- **Response Examples:**

  - **Code:** 200
    Content:

```
{
  "msg": "Account was successfully assigned to the group",
  "error": 0,
  "errors": []
}
```

- **Code:** 401
  Content:

```
{
  "msg": "Validator failed",
  "error": 1,
  "errors": {
    "account_id": [
      "The account id field is required."
    ]
  }
}
```

Accounts Assign

Assign multiple accounts to group

- **URL**

  /api/v1/groups/:id/accounts/assign

- **Method:**

  `POST`

- **Data Params**

  Required:

  `accounts=[array]`
  (account ids)

- **Response Examples:**

  - **Code:** 200
    Content:

```
{
  "msg": "Accounts were successfully assigned to the group",
  "error": 0,
  "errors": []
}
```

- **Code:** 401
  Content:

```
{
  "msg": "Validator failed",
  "error": 1,
  "errors": {
    "account_id": [
      "The accounts field is required."
    ]
  }
}
```

Account Unassign

Unassign account from group

- **URL**

  /api/v1/groups/:id/account/unassign

- **Method:**

  `POST`

- **Data Params**

  Required:

  `account_id=[int]`

- **Response Examples:**

  - **Code:** 200
    Content:

```
{
  "msg": "Account was successfully un-assigned from the group",
  "error": 0,
  "errors": []
}
```

- **Code:** 401
  Content:

```
{
  "msg": "Validator failed",
  "error": 1,
  "errors": {
    "account_id": [
      "The account_id field is required."
    ]
  }
}
```

Syndications Endpoints (v1)

All

Fetches all syndications for currently authorized user

- **URL**

  /api/v1/syndications/all

- **Method:**

  `GET`

- **Data Params**

    None

- **Response Examples:**

  - **Code:** 200
    Content:

```
{
  "msg": "Ok",
  "error": 0,
  "data": [
    {
      "id": 89,
      "campaign_id": 37,
      "account_id": 85,
      "tier_id": null,
      "attempts": 0,
      "success": 0,
      "skipped": 0,
      "attempted_at": null,
      "post_id": null,
      "post_url": null,
      "updated_at": "2017-04-18 02:50:17",
      "created_at": "2017-04-18 02:50:17"
    }
  ]
}
```

- **Code:** 200
  Content:

```
{
  "msg": "You don't have any syndications",
  "error": 0,
  "data": []
}
```

Pending

Fetches pending syndications for currently authorized user

- **URL**

  /api/v1/syndications/pending

- **Method:**

  `GET`

- **Data Params**

  `None`

- **Response Examples:**

    - **Code:** 200
      Content:

```
{
  "msg": "Ok",
  "error": 0,
  "data": [
    {
      "id": 89,
      "campaign_id": 37,
      "account_id": 85,
      "tier_id": null,
      "attempts": 0,
      "success": 0,
      "skipped": 0,
      "attempted_at": null,
      "post_id": null,
      "post_url": null,
      "updated_at": "2017-04-18 02:50:17",
      "created_at": "2017-04-18 02:50:17"
    }
  ]
}
```

- **Code:** 200
  Content:

```
{
  "msg": "You don't have any pending syndications",
  "error": 0,
  "data": []
}
```

Failed

Fetches failed syndications for currently authorized user

- **URL**

  /api/v1/syndications/failed

- **Method:**

  `GET`

- **Data Params**

  `None`

- **Response Examples:**

    - **Code:** 200
      Content:

```
{
  "msg": "Ok",
  "error": 0,
  "data": [
    {
      "id": 89,
      "campaign_id": 37,
      "account_id": 85,
      "tier_id": null,
      "attempts": 0,
      "success": 0,
      "skipped": 0,
      "attempted_at": null,
      "post_id": null,
      "post_url": null,
      "updated_at": "2017-04-18 02:50:17",
      "created_at": "2017-04-18 02:50:17"
    }
  ]
}
```

- **Code:** 200
  Content:

```
{
  "msg": "You don't have any failed syndications",
  "error": 0,
  "data": []
}
```

Successful

Fetches successful syndications for currently authorized user

- **URL**

  /api/v1/syndications/successful

- **Method:**

  `GET`

- **Data Params**

  `None`

- **Response Examples:**

    - **Code:** 200
      Content:

```
{
  "msg": "Ok",
  "error": 0,
  "data": [
    {
      "id": 89,
      "campaign_id": 37,
      "account_id": 85,
      "tier_id": null,
      "attempts": 0,
      "success": 0,
      "skipped": 0,
      "attempted_at": null,
      "post_id": null,
      "post_url": null,
      "updated_at": "2017-04-18 02:50:17",
      "created_at": "2017-04-18 02:50:17"
    }
  ]
}
```

- **Code:** 200
  Content:

```
{
  "msg": "You don't have any successful syndications yet",
  "error": 0,
  "data": []
}
```