A7: High-level architecture. Privileges. Web resources specification

This document catalogues the resources used by *SegFault*, and identifies their properties. These include references to the graphical interfaces, and the format of JSON responses. Furthermore, this artifact presents a documentation for the web application, including all operations over data: create, read, update, and delete.

1. Overview

This section presents an overview of the web application to implement, identifies the application's modules and briefly describes them. The web resources associated with each module are detailed in the individual documentation of each module.

M01: Authentication	Web resources associated with user authentication, includes the following system features: login/logout and registration.
M02: Individual Profile	Web resources associated with individual profile management, includes the following system features: view and edit personal profile information, view personal notifications and favorite questions.
M03: Messages	Web resources associated with questions, answers and their comments, includes the following system features: add, view, vote, report and delete questions, add, view, vote, report and delete answers and add, view, vote, report and delete comments.
M04: Categories	Web resources associated with categories, includes the following system features: list categories and search categories
M05: Static Pages	Web resources with static content are associated with this module: about and 404 page.

2. Permissions

This section defines the permissions used in the modules to establish the conditions of access to resources, in increasing order of restrictiveness.

PUB	Public	Group of users without privileges.
USR	User	Group of authenticated users.

OWN	Owner	Group of users that can update their profiles and have privileges regarding their messages (extension over the *USR* permissions).
MOD	Moderator	Group of Moderators that can manage any message (extension over the *USR* permissions).

3. Modules

This section documents each web resource of each module, indicating the URL, HTTP methods, the possible parameters, interfaces with the user (referring to the A3 artefact) — or JSON responses in the event of APIs.

> Web resources organized by module Document and describe the web resources associated with each module, indicating the URL, HTTP method, request parameters and response. Follow the RESTful resource naming At the end of this page is presented some usual descriptors to document the web resources.

3.1 Module M01: Authentication

Endpoints of module *Authentication*

These are the endpoints available in the Authentication module:

- R101: Login Action `/login`
- R102: Logout Action `/logout`
- R103: Register Form `/register`
- R104: Register Action `/register`

R101: Login Action

URL	`/login`
Description	This web resource logs the user into the system.
Request Body	+username: string (account username) +password: string (account password)
Method	POST
Redirects	Reloads the page on success or failure.
Permissions	PUB

R102: Logout Action

URL	`/logout`

Description	This web resource logs out the authenticated user (or moderator).
Method	POST
Redirects	Reloads the page.
Permissions	USR

R103: Register Form

URL	`/register`
Description	Page with a form to register a new user account.
Method	GET
UI	UI06
Submit	R104
Permissions	PUB

R104: Register Action

URL	`/register`
Description	This web resource inserts the user into the system.
Method	POST
Redirects	Success - Main page Failure - Register
Request Body	+username: string (account username) +email: string (account email) +password: string (account password)
Permissions	PUB

3.2 Module M02: Individual Profile

Endpoints of module *Individual Profile*

These are the endpoints available in the Individual Profile module:

- R201: View Profile `/users/{user_id}`
- R202: Edit Profile Form `/users/{user_id}/edit_profile`
- R203: Edit Profile Action `/users/{user_id}`
- R204: Change Password Form `/users/{user_id}/change_password`
- R205: Change Password Action `/users/{user_id}`
- R206: View Favourite Questions `/users/{user_id}/favourites`
- R207: Delete a Favourite Question `/users/{user_id}/favourites/{question_id}`
- R208: Add a Favourite Question `/users/{user_id}/favourites/{question_id}`

R201: View Profile

URL	`/users/{user_id}`
Description	Shows a user's profile page.
Method	GET
Parameters	+user_id: integer (user primary key)
UI	UI05
Permissions	PUB

R202: Edit Profile Form

URL	`/users/{user_id}/edit_profile`
Description	Page with a form to edit profile info.
Method	GET
Parameters	+user_id: integer (user primary key)
UI	UI09
Submit	R203
Permissions	OWN

R203: Edit Profile Action

URL	`/users/{user_id}`
Description	Web resource that changes a user's profile info based on its input.
Method	POST
Parameters	+user_id: integer (user primary key)
Request Body	?background_picture: string (New background picture path) ?profile_picture: string (New profile picture path) ?biography: string (New user's biography)
Response Body	JSON05
Permissions	OWN

R204: Change Password Form

URL	`/users/{user_id}/change_password`
Description	Page with a form to change the user's password.

Method	GET
Parameters	+user_id: integer (user primary key)
UI	Click on “Settings”
Submit	R205
Permissions	OWN

R205: Change Password Action

URL	`/users/{user_id}`
Description	Web resource to change a user’s password.
Method	POST
Parameters	+user_id: integer (user primary key)
Request Body	+old_password: string (The previous password) +new_password: string (The new password)
Permissions	OWN

206: View Favourite Questions

URL	`/users/{user_id}/favourites`
Description	Web resource that gets all favourite questions of a user.
Method	GET
Parameters	+user_id: integer (user primary key)
Response Body	JSON01
Permissions	OWN

207: Delete a Favourite Question

URL	`/users/{user_id}/favourites/{question_id}`
Description	Web resource that deletes a question from a user’s favourites.
Method	DELETE
Parameters	+user_id: integer (user primary key) +question_id: integer (the primary key of the question to delete)
Permissions	OWN

208: Add a Favourite Question

URL	`/users/{user_id}/favourites/{question_id}`
Description	Web resource that adds a question to a user's favourites.
Method	POST
Parameters	+user_id: integer (user primary key) +question_id: integer (the primary key of the question to add)
Response Body	JSON02
Permissions	OWN

3.3 Module M03: Messages

Endpoints of module *Messages*

- R301: Search Questions Page `/questions`
- R302: Get recent questions `/questions/recent`
- R303: Get hot questions `/questions/hot`
- R304: Get highly voted questions `/questions/highly-voted`
- R305: Get active questions `/questions/active`
- R306: Get question's details `/questions/{id}`
- R307: Add a new question - Form `/questions/create`
- R308: Add a new question - Action `/questions`
- R309: Edit a question - Form `/questions/{id}/edit`
- R310: Edit a question - Action `/questions/{id}`
- R311: Delete a question `/questions/{id}`
- R312: Get question's answers `/questions/{id}/answers`
- R313: Get answer's details `/questions/{id}/answers/{answer_id}`
- R314: Add a new answer `/questions/{id}/answers`
- R315: Edit an answer `/questions/{id}/answers/{answer_id}`
- R316: Delete an answer `/questions/{id}/answers/{answer_id}`
- R317: Get question's comments `/questions/{id}/comments`
- R318: Add a new comment to a question `/questions/{id}/comments`
- R319: Edit a question's comment `/questions/{id}/comments/{comment_id}`
- R320: Delete a question's comment `/questions/{id}/comments/{comment_id}`
- R321: Get answer's comments `/questions/{id}/answers/{answer_id}/comments`
- R322: Add a new comment to an answer `/questions/{id}/answers/{answer_id}/comments`
- R323: Edit an answer's comment `/questions/{id}/answers/{answer_id}/comments/{comment_id}`
- R324: Delete an answer's comment `/questions/{id}/answers/{answer_id}/comments/{comment_id}`

R301: Search Questions Page

URL	`/questions`
Description	*Navbar* on the homepage allows searching for a question by it's title.
Method	GET

Parameters	+query: string (String field to search for in questions). ?categories: id[] (Optionally, filter questions by categories, by their foreign keys).
UI	UI01
Response Body	JSON01
Permissions	PUB

R302: Get Recent Questions

URL	`/questions/recent`
Description	Show the 25 most recent questions.
Method	GET
UI	UI01
Response Body	JSON01
Permissions	PUB

R303: Get Hot Questions

URL	`/questions/hot`
Description	Show the 25 *hottest* questions (highly voted questions with most answers but no correct answer).
Method	GET
UI	UI01
Response Body	JSON01
Permissions	PUB

R304: Get Highly Voted Questions

URL	`/questions/highly-voted`
Description	Show the 25 questions with highest score.
Method	GET
UI	UI01
Response Body	JSON01
Permissions	PUB

R305: Get Active Questions

URL	`/questions/active`
Description	Show 25 unanswered questions.
Method	GET
UI	UI01
Response Body	JSON01
Permissions	PUB

R306: Get Question's Details

URL	`/questions/{id}`
Description	The details and contents of a question.
Method	GET
UI	UI03
Parameters	+id: integer (The question's *id*)
Response Body	JSON02
Permissions	PUB

R307: Add a New Question - Form

URL	`/questions/create`
Description	Page with a form for creating a new question.
Method	GET
UI	UI04
SUBMIT	R308
Permissions	USR

R308: Add a New Question - Action

URL	`/questions`
Description	Web resource that creates a new question based on the input received. Redirects to the new question page on success, and back to new question form on failure.
Method	POST
Request Body	+title: string (The question's title) +content: string (The question's contents)

	+author: integer (The question author’s id) +categories: id[] (The question’s categories’s foreign keys)
Redirects	R306 - SUCCESS R307 - FAILURE
Permissions	USR

R309: Edit a Question - Form

URL	`/questions/{id}/edit`
Description	Page with a form for editing a question.
Method	GET
UI	UI04
SUBMIT	R310
Permissions	OWN, MOD

R310: Edit a Question - Action

URL	`/questions/{id}`
Description	Web resource that edits a previous question based on the input received. Redirects to the new question page on success, and back to edit question form on failure.
Method	PUT
Request Body	+author: integer (The question author’s id) ?content: string (The question’s contents) ?categories: id[] (The question’s categories’s foreign keys)
Redirects	R306 - SUCCESS R309 - FAILURE
Permissions	OWN, MOD

R311: Delete a question

URL	`/questions/{id}`
Description	Web resource that deletes a question based on its ID. In case of success redirect to Main page, otherwise stay in the question page.
Method	DELETE
Request Body	+question_id: integer (question primary key)
Returns	200 OK - The question was successfully deleted. 400 Bad Request - Error. Error message is specified via a HTTP header. 404 Not Found - Error. No work with the specified primary key exists.
Redirect	R302 - SUCCESS
Permissions	OWN, MOD

R312: Get question's answers

URL	`/questions/{id}/answers`
Description	Get all the answers of a question.
Method	GET
Parameters	+id: integer (The question's *id*)
Response Body	JSON07
Permissions	PUB

R313: Get answer's details

URL	`/questions/{id}/answers/{answer_id}`
Description	The details and contents of a answer.
Method	GET
Parameters	+question_id: integer (The question's *id*)
Parameters	+answer_id: integer (The answer's *id*)
Response Body	JSON08
Permissions	PUB

R314: Add a new answer

URL	`/questions/{id}/answers`

Description	Web resource that creates a new answer based on the input received.
Method	POST
Request Body	+content: string (The question’s contents) +author: integer (The question author’s id)
Permissions	USR

R315: Edit an answer

URL	`/questions/{id}/answers/{answer_id}`
Description	Web resource that edits a previous answer based on the input received.
Method	PUT
Request Body	+question_id: integer (The question’s id) +author: integer (The answer author’s id) ?content: string (The answer’s contents)
Permissions	OWN, MOD

R316: Delete an answer

URL	`/questions/{id}/answers/{answer_id}`
Description	Web resource that delete a answer based on his ID.
Method	DELETE
Request Body	+question_id: integer (question primary key) +answer_id: integer (answer primary key)
Returns	200 OK - The answer was successfully deleted. 400 Bad Request - Error. Error message is specified via a HTTP header. 404 Not Found - Error. No work with the specified primary key exists.
Permissions	OWN, MOD

R317: Get question’s comments

URL	`/questions/{id}/comments`
Description	Get all the comments of a question.
Method	GET

Parameters	+id: integer (The question's *id*)
Response Body	JSON04
Permissions	PUB

R318: Add a new comment to a question

URL	`/questions/{id}/comments`
Description	Web resource to add a new comment to a question.
Method	POST
Parameters	+id: integer (Question id)
Request Body	+content: string (Comment content) +author: integer (User id) +commentable: integer (Question id)
Response body	JSON03
Permissions	USR

R319: Edit a question's comment

URL	`/questions/{id}/comments/{comment_id}`
Description	Web resource that allows the edition of a question's comment.
Method	PUT
Parameters	+id: integer (Question id) +comment_id: integer (Comment id)
Request Body	+content: string (Comment content) +editor: integer (User id) +commentable: integer (Question id) +comment: integer (Comment id)
Response body	JSON03
Permissions	OWN, MOD

R320: Delete a question's comments

URL	`/questions/{id}/comments/{comment_id}`
Description	Web resource that allows the deletion of a question's comment.
Method	DELETE
Parameters	+id: integer (Question id) +comment_id: integer (Comment id)
Request Body	+commentable: integer (Question id) +comment: integer (Comment id)

Returns	200 OK - The comment was successfully edited. 400 Bad Request - Error. Could not delete the comment
Permissions	OWN, MOD

R321: Get answer's comments

URL	`/questions/{id}/answers/{answer_id}/comments`
Description	Web resource that allows to show all the answer's comments.
Method	GET
Parameters	+id: integer (Question id) +answer_id: integer (Answer id) +comment_id: integer (Comment id)
Response body	JSON04
Permissions	PUB

R322: Add a new comment to an answer

URL	`/questions/{id}/answers/{answer_id}/comments`
Description	Web resource that allows to add a new comment to an answer.
Method	POST
Parameters	+id: integer (Question id) +answer_id: integer (Answer id)
Request Body	+content: string (Answer content) +author: integer (User id) +commentable: integer (Answer id)
Response body	JSON03
Permissions	USR

R323: Edit an answer's comment

URL	/questions/{id}/answers/{answer_id}/comments/{comment_id}
Description	Web resource that allows the edition of a answer's comment.
Method	PUT
Parameters	+id: integer (Question id) +answer_id: integer (Answer id) +comment_id: integer (Comment id)
Request Body	+content: string (Comment content) +editor: integer (User id) +commentable: integer (Answer id) +comment: integer (Comment id)

Response body	JSON03
Permissions	OWN, MOD

R324: Delete an answer's comment

URL	`/questions/{id}/answers/{answer_id}/comments/{comment_id}`
Description	Web resource that allows the deletion of a answer's comment.
Method	DELETE
Parameters	+id: integer (Question id) +answer_id: integer (Answer id) +comment_id: integer (Comment id)
Request Body	+commentable: integer (Answer id) +comment: integer (Comment id)
Returns	200 OK - The comment was successfully edited. 400 Bad Request - Error. Could not delete the comment
Permissions	OWN, MOD

3.4 Module M04: Categories

Endpoints of module _Categories These are the endpoints available in the Categories module: * R401: Categories page `/categories` * R402: Search Categories `/categories/search` * R403: Get category's questions `/categories/{id}/questions`

R401: Categories Page

URL	`/categories`
Description	Get categories page.
Method	GET
UI	UI05
Permissions	PUB

R402: Search Categories

URL	`/categories/search`
Description	Web resource that allows searching for a category by it's name.
Method	GET
Parameters	+query: string (String field to search for in categories).

Response Body	JSON06
Permissions	PUB

R403: Get category's questions

URL	`/categories/{id}/questions`
Description	Web resource that shows the questions associated with a category.
Method	GET
Parameters	+id: integer (The category's *id*)
Response Body	JSON01
Permissions	PUB

3.5 Module M05: Static Pages

Endpoints of module *Static Pages*

These are the endpoints available in the Static module: * R501: About Page `/about` * R502: 404 Page `/404`

R501: About Page

URL	`/about`
Description	Get about page.
Method	GET
UI	UI02
Permissions	PUB

R502: 404 Page

URL	`/404`
Description	Get 404 page.
Method	GET
UI	UI08
Permissions	PUB

4. JSON/XML Types

JSON01: Questions {question}[]

JSON response containing a list of questions and their contents.

```
{
  "question": [
    {
      "id": 137,
      "title": "Is there multiple inheritance in JAVA?",
      "content": "I'm a C++ developer and have always been intrigued by the
        fact that some languages allow for multiple inheritance and others
        don't.\nIs there a deeper thought process behind this?",
      "author": "sudomakemeasandwich",
      "score": 120,
      "correct_answer": 43,
      "creation_time": "2018-04-07 14:11",
      "category": [
        {
          name: "JAVA",
          id: 1
        },
        {
          name: "OOP",
          id: 2
        }
      ]
    },
    {
      "id": 12,
      "title": "Is javascript dynamically typed?",
      "content": "I was wondering if javascript was a dynamically typed language?",
      "author": "sudormrf",
      "score": 10,
      "correct_answer": 435,
      "creation_time": "2018-03-01 10:03",
      "category": [
        {
          name: "JAVA",
          id: 1
        },
        {
          name: "OOP",
          id: 2
        }
      ]
    }
  ]
}
```

JSON02: Question {question}

JSON response containing a question's details and contents.

```
{
  "question":
    {
      "id": 137,
      "title": "Is there multiple inheritance in JAVA?",
      "author": "sudomakemeasandwich",
      "score": 120,
      "correct_answer": 43,
      "category": [
        {
          name: "JAVA",
          id: 1
        },
        {
          name: "OOP",
          id: 2
        }
      ],
      "was_edited": true,
      "content":
        {
          "version": "I'm a C++ developer and have always been intrigued by the
          fact that some languages allow for multiple inheritance and others
          don't.\nIs there a deeper thought process behind this?",
          "creation_time": "2018-04-07 14:11",
          "author": "sudomakemeasandwich"
        }
    }
}
```

JSON03: Comment {comment}

JSON response containing a comment's details and contents.

```
{
  "comment":
    {
      "id": 236,
      "author": "sudomakemeasandwich",
      "score": 10,
      "was_edited": true,
      "content":
        {
          "version": "Heyyy wadup",
          "creation_time": "2018-03-07 01:11",
          "author": "sudomakemeasandwich"
        }
    }
}
```

JSON04: Comments {comment}[]

JSON response containing a list of comments and their contents.

```
{
  "comment": [
    {
      "id": 236,
      "author": "sudomakemeasandwich",
      "score": 10,
      "was_edited": true,
      "content":
        {
          "version": "Heyy I agreeeee",
          "creation_time": "2018-04-08 11:10",
          "author": "sudomakemeasandwich"
        }
    }
  ]
}
```

JSON05: Profile {user}

JSON response containing the contents of a user's profile.

```
{
  "username": "sudomakemeasandwhich",
  "background_img": "imgs/profile/standard-bg.png",
  "profile_img": "imgs/profile/linuxislove.png",
  "biography": "Better than uml only ..."
}
```

JSON06: Categories {category}[]

JSON response containing a list of categories and their information.

```
{
  "category": [
    {
      "id": 21,
      "name": "java",
      "description": "Java is a general-purpose computer-programming language
        that is concurrent, class-based, object-oriented,[15] and specifically
        designed to have as few implementation dependencies as possible",
      "num_posts" : 54
    }
  ]
}
```

JSON07: Answers {answer}[]

JSON response containing a list of answers and their information.

```
{
  "answer": [
    {
      "id": 702,
      "author": "amccoy0",
      "score": 3,
      "was_edited": false,
      "content":
        {
          "version": "Use php7.1 dummy",
          "creation_time": "2018-04-07 14:51",
          "author": "sudomakemeasandwich"
        }
    }
  ]
}
```

JSON08: Answer {answer}

JSON response containing an answer's details and contents.

```
{
  "answer":
    {
      "id": 137,
      "author": "sudomakemeasandwich",
      "score": -1,
      "was_edited": false,
      "content":
        {
          "version": "Your question makes no sense...",
          "creation_time": "2018-04-07 18:11",
          "author": "sudomakemeasandwich"
        }
    }
}
```

Revision history

- 12/04/2018: Deleted “Login Form” resource, as its not a separate page;
- 12/04/2018: Added links to all UIs;
- 12/04/2018: Reassessed the usfulness of response with status code 202 or 204, deleted most;
- 12/04/2018: Deleted UI field of R311-R314;
- 12/04/2018: Modified JSONs to include flag *was_edited* instead of all versions of a message;
- 12/04/2018: Corrected R402’s URL;
- 12/04/2018: Added categories’ foreign keys to JSON responses, and restricted to only categories foreign keys when adding a question (instead of a string) - R301, R308, R310;

GROUP1763, 12/04/2018

André Cruz, up201503776@fe.up.pt
Daniel Marques, up201503822@fe.up.pt
Edgar Carneiro, up201503784@fe.up.pt
João Carvalho, up201504875@fe.up.pt