# TUTORINGTRAIN

# WEBSERVICE

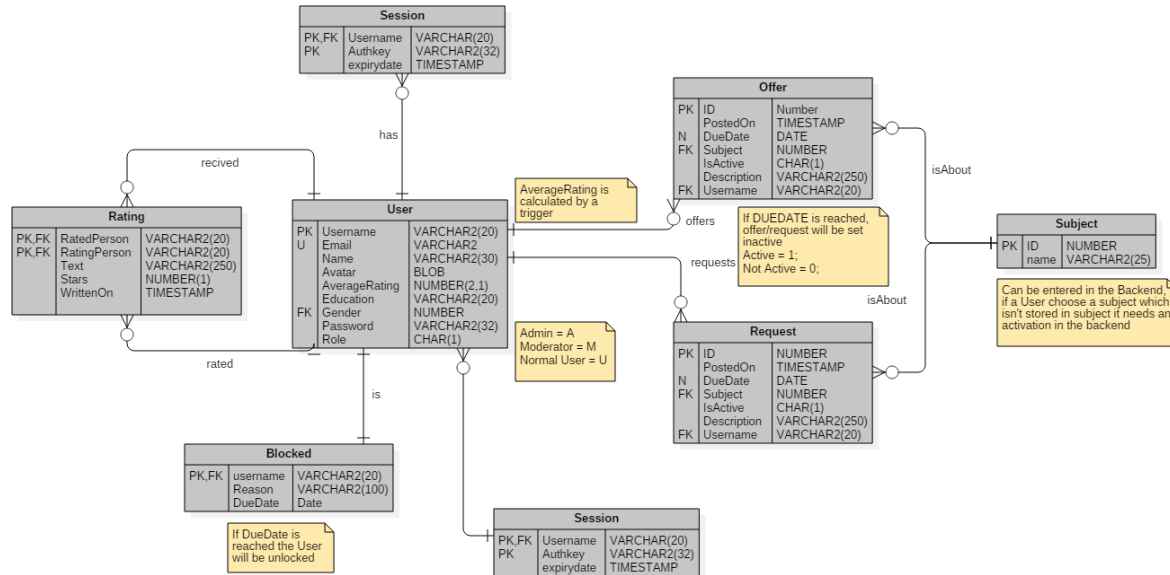# V1.2

REST-DOKU

# TABLE OF CONTENTS

# GENERAL

## Data Model

**Session**

| PK,FK | Username | VARCHAR(20) |
|---|---|---|
| PK | Authkey | VARCHAR2(32) |
| | expirydate | TIMESTAMP |

**Offer**

| PK | ID | Number |
|---|---|---|
| N | PostedOn | TIMESTAMP |
| FK | DueDate | DATE |
| | Subject | NUMBER |
| | IsActive | CHAR(1) |
| | Description | VARCHAR2(250) |
| FK | Username | VARCHAR2(20) |

**Rating**

| PK,FK | RatedPerson | VARCHAR2(20) |
|---|---|---|
| PK,FK | RatingPerson | VARCHAR2(20) |
| | Text | VARCHAR2(250) |
| | Stars | NUMBER(1) |
| | WrittenOn | TIMESTAMP |

**User**

| PK | Username | VARCHAR2(20) |
|---|---|---|
| U | Email | VARCHAR2 |
| | Name | VARCHAR2(30) |
| | Avatar | BLOB |
| | AverageRating | NUMBER(2,1) |
| FK | Education | VARCHAR2(20) |
| | Gender | NUMBER |
| | Password | VARCHAR2(32) |
| | Role | CHAR(1) |

*AverageRating is calculated by a trigger*

*If DUEDATE is reached, offer/request will be set inactive Active = 1; Not Active = 0;*

*Admin = A Moderator = M Normal User = U*

**Subject**

| PK | ID | NUMBER |
|---|---|---|
| | name | VARCHAR2(25) |

*Can be entered in the Backend, if a User choose a subject which isn't stored in subject it needs an activation in the backend*

**Request**

| PK | ID | NUMBER |
|---|---|---|
| N | PostedOn | TIMESTAMP |
| FK | DueDate | DATE |
| | Subject | NUMBER |
| | IsActive | CHAR(1) |
| | Description | VARCHAR2(250) |
| FK | Username | VARCHAR2(20) |

**Blocked**

| PK,FK | username | VARCHAR2(20) |
|---|---|---|
| | Reason | VARCHAR2(100) |
| | DueDate | Date |

*If DueDate is reached the User will be unlocked*

**Session**

| PK,FK | Username | VARCHAR(20) |
|---|---|---|
| PK | Authkey | VARCHAR2(32) |
| | expirydate | TIMESTAMP |

recived, rated, has, is, offers, requests, isAbout

## Only JSON format is supported.

All JSON attributes marked in **THIS COLOR** are optional.

| | |
|---|---|
| Dates format is: | "**yyyy-MM-dd'T'hh:mm:ssZ**" |
| Example: | **2017-09-03T09:45:12+0200** |

# METHODS

## [1] SECURITY & AUTHENTICATION

Authentication is managed via session tokens. This is a randomly generated 32-character string which is used to identify the logged in user.

### 1. Obtaining a session token

**Note:** A new session token is generated every time this method is called with valid credentials.

A token expires 24 hours after its creation.

**Request**

| Method | URL | Role |
|--------|-----|------|
| POST | url/authentication | None |

| MediaType | Sample Request Body |
|-----------|---------------------|
| application/json | {<br>  "username": "testuser",<br>  "password": "098f6bcd4621d373cade4e832627b4f6",<br>  "requiredRole": "A"<br>} |

**Response**

| Status | Response |
|--------|----------|
| 200 | 654d4e57c9a868dd87759268b225ae44 |
| 401 | Password and/or username are invalid. |
| 403 | User cannot authenticate himself in current context. (eg. normal user tries to log in with admin tool → set requiredRole attribute for this to work) |
| 450 | User is currently blocked. (Reason is given in response body) |

### 2. Using the session token

Once a session token has been acquired it can be used to call all other methods. An exception is registering a User as this method does not require any authentication.

For every request the token has to be placed in the "Authorization"-Header in the following format:

"Bearer <token>"

**Example:**

| Key | Value |
|-----|-------|
| Authorization | Bearer 4d8068152f82f0ae9ed117365db71cec |

A method called without a session token returns **401** HTTP status.

If **401** is returned when a session token was provided it can be assumed that the token has expired or is invalid. In this case you can try to acquire a new token.

If **403** is returned the user associated with the session key has insufficient privileges to perform that method.

If **450** is returned the user associated with the session key has been blocked.

If **451** is returned, the provided JSON-text is invalid. Further details might be given in the returned message.

If **500** is returned an unknown error happened. Further details might be given in the returned message.

These response codes apply to all methods described below but are not mentioned there. Only method-specific response codes are listed.

## 3. User roles

This System currently supports three roles:

User (U), Moderator (M) and Admin (A)

These roles extend the privileges of their inferior roles, this means a Moderator can perform all actions a User can, and an Admin can perform all actions a Moderator can.

# [2]  ERROR HANDLING AND LOCALIZATION

In the event of an error / exception, an Error object of the following structure is returned:

```
{
   "code": <error code>,
   "message": <error message>
}
```

The following error codes exist:

| Code | Descrption |
|---|---|
| 1 | Unknown error – further details might be given in message |
| 2 | Admin only – happens when user without Admin role tries to authenticate when "requiredRole" is set to 'A' |
| 3 | Token invalid – the authentication token does not exist |
| 4 | Token expired – the authentication token has expired |
| 5 | Insufficient privileges – the user does not have to required role to perform this action |
| 6 | Username is null – No username was given |
| 7 | Name is null – No name was given (e.g. Create Subject) |
| 8 | User is null – No user was given |
| 9 | Offer is null – No offer was given |
| 10 | Gender is null – No gender was given |
| 11 | Subject is null – No subject was given |
| 12 | User not found – The user was not found |
| 13 | Subject not found – The subject was not found |
| 14 | Offer not found – The offer was not found |
| 15 | Gender not found – The gender was not found |
| 16 | Invalid email – The given email is of an invalid format |
| 17 | Query parameters missing – "start" and "pageSize" query parameters were not provided |
| 18 | Subject name conflict – The subject name already exists |
| 19 | Username conflict – The username is not available (already in use) |
| 20 | Email conflict – The email is not available (already in use) |
| 21 | Cannot block own user |
| 22 | Cannot unblock own user |
| 23 | Cannot block other admins |
| 24 | Authentication failed – Invalid credentials were provided |
| 25 26 27 28 | Blocked – The user is blocked from using the application |
| 29 | Invalid JSON – The provided JSON-Request is malformed |

While the "code" identifies the error, the "message" attribute provides a more detailed description of the error.

To receive a localized error message, set the "Accept-Language"-HTTP Header to the desired language. Currently only English (en) and German (de) are supported.
If no "Accept-Language" – Header is provided, English is used by default.

# [3]  USER RESOURCE

## 1. Register User

### Request

| Method | URL | Role |
|--------|-----|------|
| POST | url/user/register | None |

| MediaType | Sample Request Body |
|-----------|---------------------|
| application/json | {<br>  "name": "Kevin Paul",<br>  "username": "kevin",<br>  "email": "kevin.paul@gmx.at",<br>  "password": "098f6bcd4621d373cade4e832627b4f6",<br>  "education": "HTL",<br>  "gender": {<br>   "id": 2<br>  }<br>} |

### Response

| Status | Response |
|--------|----------|
| 200 | {<br>  "username": "kevin",<br>  "role": "U",<br>  "email": "kevin.paul@gmx.at",<br>  "name": "Kevin Paul",<br>  "education": "HTL",<br>  "gender": {<br>   "id": 2,<br>   "name": "Male"<br>  }<br>} |
| 409 | Username already exists. |

## 2. Get all Users

### Request

| Method | URL | Role |
|--------|-----|------|
| GET | url/user/all | Admin |

### Parameters

| Type | Name | Description |
|------|------|-------------|
| Query | start | Start index of users to get |
| Query | pageSize | Maximum number of offers |

**Response**

| Status | Response |
|--------|----------|
| **200** | <pre>[<br>  {<br>    "username": "kevin",<br>    "role": "U",<br>    "email": "kevin.paul@gmx.at",<br>    "name": "Kevin Paul",<br>    "education": "HTL",<br>    "gender": {<br>      "id": 2,<br>      "name": "Male"<br>    }<br>  },<br>  {<br>    "username": "test",<br>    "role": "U",<br>    "email": "santnere@edu.htl-villach.at",<br>    "name": "wacco",<br>    "education": "HTL",<br>    "gender": {<br>      "id": 2,<br>      "name": "Male"<br>    },<br>    "block": {<br>      "reason": "test"<br>    }<br>  },<br>  {<br>    "username": "wacco",<br>    "role": "A",<br>    "email": "santnere@edu.htl-villach.at",<br>    "name": "wacco",<br>    "education": "HTL",<br>    "gender": {<br>      "id": 2,<br>      "name": "Male"<br>    }<br>  }<br>]</pre> |

## 3. Get own User

**Request**

| Method | URL | Role |
|--------|-----|------|
| GET | url/user | User |

**Response**

| Status | Response |
|--------|----------|
| **200** | {<br>  "username": "wacco",<br>  "role": "A",<br>  "email": "santnere@edu.htl-villach.at",<br>  "name": "wacco",<br>  "education": "HTL",<br>  "gender": {<br>   "id": 2,<br>   "name": "Male"<br>  }<br>} |

## 4. Update any User

**Request**

| Method | URL | Role |
|--------|-----|------|
| PUT | url/user/update | Admin |

| MediaType | Sample Request Body |
|-----------|---------------------|
| application/json | {<br>  "username": "kevin",<br>  "name": "Paul Kevin",<br>  "email": "coolerkev@gmx.at",<br>  "password": "9d5e3ecdeb4cdb7acfd63075ae046672",<br>  "education": "HTL",<br>  "gender": {<br>   "id": 1<br>  }<br>} |

**Response**

| Status | Response |
|--------|----------|
| **200** | |
| **453** | User not found |

## 5. Update own User

**Request**

| Method | URL | Role |
|--------|-----|------|
| PUT | url/user/update/own | User |

| MediaType | Sample Request Body |
|-----------|---------------------|
| application/json | {<br>  "name": "Elias Santler",<br>  "email": "santnere@edu.htl-villach.at",<br>  "password": "098f6bcd4621d373cade4e832627b4f6", |

| | "education": "HTL",<br>"gender": {<br>  "id": 1<br>}<br>} |
|---|---|

**Response**

| Status | Response |
|---|---|
| **200** | |

## 6. Block User

**Request**

| Method | URL | Role |
|---|---|---|
| POST | url/user/block | Admin |

| MediaType | Sample Request Body |
|---|---|
| application/json | {<br>  "username": "test",<br>  "reason": "test",<br>  "duedate": "2017-09-03T09:45:12+0200"<br>} |

**Response**

| Status | Response |
|---|---|
| **200** | |
| **456** | Error blocking user. (Further details may be provided via body) |

## 7. Unblock User

**Request**

| Method | URL | Role |
|---|---|---|
| GET | url/user/unblock/{username} | Admin |

**Parameters**

| Type | Name | Description |
|---|---|---|
| Path | Username | Username of user to unblock |

**Response**

| Status | Response |
|---|---|
| **200** | |

## 8. Get all Genders

**Request**

| Method | URL | Role |
|--------|-----|------|
| GET | url/user/gender | User |

**Response**

| Status | Response |
|--------|----------|
| 200 | `[`<br>`  {`<br>`    "id": 3,`<br>`    "name": "Female"`<br>`  },`<br>`  {`<br>`    "id": 2,`<br>`    "name": "Male"`<br>`  },`<br>`  {`<br>`    "id": 1,`<br>`    "name": "Unspecified"`<br>`  }`<br>`]` |

# [4]  OFFER RESOURCE

## 1. Create Offer

**Request**

| Method | URL | Role |
|--------|-----|------|
| POST | url/offer | User |

| MediaType | Sample Request Body |
|-----------|---------------------|
| application/json | ```<br>{<br>  "description": "This is just a test offer",<br>  "subject": {<br>   "id": 3<br>  }<br>}<br>``` |

**Response**

| Status | Response |
|--------|----------|
| 200 | ```<br>{<br>  "id": 1,<br>  "postedon": "2017-10-04T07:18:39+0200",<br>  "isactive": "1",<br>  "description": "This is just a test offer",<br>  "subject": {<br>   "id": 1,<br>   "name": "English"<br>  },<br>  "user": {<br>   "username": "wacco"<br>  }<br>}<br>``` |
| 452 | Subject not found |

## 2. Update Offer

**Request**

| Method | URL | Role |
|--------|-----|------|
| PUT | url/offer | User |

| MediaType | Sample Request Body |
|-----------|---------------------|
| application/json | ```<br>{<br>  "id": 1,<br>  "isactive": "1",<br>  "description": "This is just a test offer",<br>  "subject": {<br>   "id": 1<br>  }<br>}<br>``` |

## Response

| Status | Response |
|--------|----------|
| 200 | |
| 452 | Subject not found |
| 454 | Offer not found |

## 3. Get newest Offers

### Request

| Method | URL | Role |
|--------|-----|------|
| GET | url/offer/new | User |

### Parameters

| Type | Name | Description |
|------|------|-------------|
| Query | start | Start index of offers to get |
| Query | pageSize | Maximum number of offers |

### Response

| Status | Response |
|--------|----------|
| 200 | ```[
  {
    "id": 2,
    "postedon": "2017-10-04T07:27:58+0200",
    "isactive": "1",
    "description": "Test offer 2",
    "subject": {
      "id": 1,
      "name": "English"
    },
    "user": {
      "username": "wacco"
    }
  },
  {
    "id": 1,
    "postedon": "2017-10-04T07:18:39+0200",
    "isactive": "0",
    "description": "This is just a test offer",
    "subject": {
      "id": 1,
      "name": "English"
    },
    "user": {
      "username": "wacco"
    }
  }
]``` |
| 455 | Query parameters were not given |

## 4. Get newest Offers of User

### Request

| Method | URL | Role |
|--------|-----|------|
| GET | url/offer/new/{username} | User |

### Parameters

| Type | Name | Description |
|------|------|-------------|
| Path | Username | Username of user to get newest offers of |
| Query | start | Start index of offers to get |
| Query | pageSize | Maximum number of offers |

### Response

| Status | Response |
|--------|----------|
| 200 | <pre>[<br>  {<br>    "id": 2,<br>    "postedon": "2017-10-04T07:27:58+0200",<br>    "isactive": "1",<br>    "description": "Test offer 2",<br>    "subject": {<br>      "id": 1,<br>      "name": "English"<br>    },<br>    "user": {<br>      "username": "wacco"<br>    }<br>  },<br>  {<br>    "id": 1,<br>    "postedon": "2017-10-04T07:18:39+0200",<br>    "isactive": "0",<br>    "description": "This is just a test offer",<br>    "subject": {<br>      "id": 1,<br>      "name": "English"<br>    },<br>    "user": {<br>      "username": "wacco"<br>    }<br>  }<br>]</pre> |
| 453 | User not found |
| 455 | Query parameters were not given |

# [5]   SUBJECT RESOURCE

## 1. Create Subject

**Request**

| Method | URL | Role |
|--------|-----|------|
| POST | url/subject | Admin |

| MediaType | Sample Request Body |
|-----------|---------------------|
| application/json | {<br>  "name": "Mathe"<br>} |

**Response**

| Status | Response |
|--------|----------|
| 200 | {<br>  "id": 3,<br>  "name": "Mathe"<br>} |
| 409 | Subject name already exists (currently not implemented) |

## 2. Update Subject

**Request**

| Method | URL | Role |
|--------|-----|------|
| PUT | url/subject | Admin |

| MediaType | Sample Request Body |
|-----------|---------------------|
| application/json | {<br>  "id": 2,<br>  "name": "Deutsch"<br>} |

**Response**

| Status | Response |
|--------|----------|
| 200 | |
| 452 | Subject not found |

## 3. Delete Subject

**Request**

| Method | URL | Role |
|--------|-----|------|
| DELETE | url/subject/{id} | Admin |

**Parameters**

| Type | Name | Description |
|------|------|-------------|
| Path | Id | Id of subject to delete |

**Response**

| Status | Response |
|--------|----------|
| **200** | |
| **452** | Subject not found |

## 4. Get all Subjects

**Request**

| Method | URL | Role |
|--------|-----|------|
| GET | url/subject | User |

**Response**

| Status | Response |
|--------|----------|
| **200** | ```[``` <br> `{` <br> `"id": 2,` <br> `"name": "Deutsch"` <br> `},` <br> `{` <br> `"id": 1,` <br> `"name": "English"` <br> `},` <br> `{` <br> `"id": 3,` <br> `"name": "Mathe"` <br> `}` <br> `]` |