National Technical University of Athens,
Software Engineering Semester Project,
2022-2023

Specifications of REST web API and Command-Line Interface

REST web API: General Principles

The *intelliQ* application you will develop should support a RESTful Application Programming Interface (REST API) for managing the data and services you will implement. The API calls specified in this document must be implemented in full, regardless of whether they will be used by the use cases you choose to develop. The Command-Line Interface (CLI) should invoke the API calls without implementing the functions of the calls itself.

API documentation

The REST API you will implement should include documentation in the *OpenAPI 3.0* format (preferred) or at least in the form of a JSON file generated by the *Postman* software.

Base URL

The REST API will be available at the following base URL for all tasks:

```
https://{host}:9103/intelliq_api
```

Where `{host}` is either `localhost` or the deployment location.

The individual resources (REST endpoints) that will be provided will be accessible through the above base URL as follows:

```
{baseURL}/{service}/{path-to-resource}
```

Where `{service}` is one of the services provided, as listed below. For example, the endpoint for retrieving questionnaire questions with the identifier `poll123` is:

```
https://localhost:91003/intelliq_api/info/poll123
```

Data Formats

The REST API will produce data that supports both JSON (`content-type: application/json`) and CSV (`content-type: text/csv`) formats. The choice of format will be determined in the request as follows (query parameter):

```
{baseURL}/{service}/{path-to-resource}?format={json|csv}
```

If the format parameter is not provided, the default should be JSON. The character encoding should be UTF-8. For example, the previous call with a request for the data format "csv" would be as follows:

```
https://localhost:91003/intelliq_api/info/poll123?format=csv
```

User Authentication (Optional)

For access control purposes in the system using the API, you may optionally implement a user authentication mechanism of your choice. In this case, user accounts will be created by the system administrator via the CLI, as described later. When calling the API, user credentials (e.g., user access token) appropriately encoded should be provided in a custom HTTP header specifically for this purpose, named `X-Observatory-Auth`.

Even if user authentication is required for API usage, the responses should be anonymous. This means that for each response, a random string is stored instead of any identifiable information (such as username or email).

Error Management

Every API call should return the appropriate HTTP status codes in case of an error. Specifically, the following error codes will be returned:

200	Success	Successful call returning non-empty data payload
204	No data	In case the parameters provided in a call are not valid (e.g., empty mandatory field)
400	Bad request	If the request is made by an unauthorized user
401	Not authorized	If the response to the call is empty
500	Internal server error	In case of a general error

Access and Management

Login & Logout (Optional, if user authentication is implemented)

Your backend will support two endpoints for user login and logout. Specifically:

1. **{baseURL}/login**: Supports the HTTP POST method. Receives the parameters username and password of the user encoded as `"application/x-www-form-urlencoded"`. In case of successful user authentication, it returns a JSON object with the user's token, for example: `{"token":"FOO"}`.

2. **{baseURL}/logout**: Supports the HTTP POST method. Does not receive parameters (Caution: the user's token that needs to be logged out will be included in the custom HTTP header specifically for this purpose, as mentioned above). In case of success, it returns only the status code 200 (empty response body).

Administrative Endpoints

Your backend will support the following administrative endpoints. In case user authentication is implemented, these will be accessible only by users with the system administrator role:

1. **{baseURL}/admin/healthcheck**: Supports the GET method and confirms end-to-end connectivity between the user and the database. The backend checks the connectivity with the database to respond to the request. In case of successful connection, it returns the JSON object: `{"status":"OK", "dbconnection":[connection string]}`. Otherwise, it returns `{"status":"failed", "dbconnection":[connection string]}`. The connection string contains whatever is required for your chosen database.

2. **{baseURL}/admin/questionnaire_upd**: Supports the HTTP POST method for uploading a JSON file with data for a new questionnaire. In the HTTP call, the file must be encoded as "file" in `multipart/form-data` encoding. The required structure of the JSON file is provided in the appendix.

3. **{baseURL}/admin/resetall**: Supports the POST method and initializes all system data, including questionnaires, responses, and users (if any). In case of success, it returns the JSON object: `{"status":"OK"}`. Otherwise, it returns `{"status":"failed", "reason":"<...>}` with information about the failure.

4. **{baseURL}/admin/resetq/:questionnaireID**: Supports the POST method and deletes all responses of the questionnaire identified by `questionnaireID`. In case of success, it returns the JSON object: `{"status":"OK"}`. Otherwise, it returns `{"status":"failed", "reason":"<...>}` with information about the failure in the reason field.

5. [Optional] **{baseURL}/admin/usermod/:username/:password**: Supports the HTTP POST method for adding a new user or changing the password if the user already exists.

6. [Optional] **`{baseURL}/admin/users/:username`**: Supports the HTTP GET method for reading the details of a user.

System Functionality

Below are 5 endpoints that implement functionalities necessary for interacting with the *intelliQ* software. You are free to implement additional endpoints if required for the chosen use cases. Also, you can implement variations of the requested endpoints with new names.

However, in each case, the requested endpoints must be implemented exactly as specified.

a. {baseURL}/questionnaire/:questionnaireID

HTTP GET request that returns an object containing general information and questions of the questionnaire with the identifier `questionnaireID`, sorted by the identifier of the question.

Field	Type	Description
questionnaireID	String	The identifier of the questionnaire.
questionnaireTitle	String	The title of the questionnaire.
keywords	List	A list of keywords. Each keyword is a string.
questions	List	A list of questions. For each question, the JSON object contains the following:
qID	String	The identifier of the question.
qtext	String	The text of the question.
required	String	"true" or "false"
type	String	"question" or "profile"

b. {baseURL}/question/:questionnaireID/:questionID

HTTP GET request that returns an object containing the complete details of the question with `questionID` in the questionnaire `questionnaireID`. The choices of the question (answers) are sorted by the identifier of the answer.

Field	Type	Description
questionnaireID	String	The identifier of the questionnaire.
qID	String	The identifier of the question.
qtext	String	The text of the question.
required	String	"true" or "false"
type	String	"question" or "profile"
options	List	A list of options. Each option is represented by a JSON object with the following fields:
optID	String	The identifier of the answer.
opttxt	String	The text of the answer.
nextqID	String	The identifier of the next question.

c. {baseURL}/doanswer/:questionnaireID/:questionID/:session/:optionID

HTTP POST request that records the answer `optionID` given in the answer session for the question `questionID` of the questionnaire `questionnaireID`. It does not return any object. The identifier session is a string with 4 random characters corresponding to the answer session of the questionnaire (note: not the question).

d. {baseURL}/getsessionanswers/:questionnaireID/:session

HTTP GET request that returns an object containing the answers given to all the questions of the questionnaire `questionnaireID` during the answer session `session`. The answers are sorted by the identifier of the question.

Field	Type	Description
questionnaireID	String	The identifier of the questionnaire.
session	String	The identifier of the answer session.
answers	List	A list of choices made during the answer session. Each choice is represented by a JSON object with the following fields:

qID	String	The identifier of the question.
ans	String	The identifier of the answer given during the answer session.

e. {baseURL}/getquestionanswers/:questionnaireID/:questionID

HTTP GET request that returns an object containing the answers given to the question `questionID` in all answer sessions. The answers are sorted by the order in which they were given.

Field	Type	Description
questionnaireID	String	The identifier of the questionnaire.
questionID	String	The identifier of the question.
answers	List	A list of choices made for the specified question in various answer sessions. Each choice is represented by a JSON object with the following fields:
session	String	The identifier of the answer session.
ans	String	The identifier of the answer given during the specified answer session.

Command Line Interface – CLI

General Principles

The specifications of the CLI include commands that are equivalent to those of the REST API. The CLI should be accessible only from the console (command line, ssh) of the system hosting the application you are building. User accounts, if implemented, are created only through the CLI.

Data Retrieval

The CLI will support both JSON and CSV formats, like the REST API. The choice of the format will be determined by a corresponding mandatory parameter, as described below. The CLI will be invoked from the command line with commands of the following form:

```
se22XX scope --param1 value1 [--param2 value2 ...] --format fff
```

Where XX is the identifier of your group, and `scope` is as specified in the table that follows. In case no parameters are provided, the supported parameters for all scopes should be displayed by the CLI. For example, assuming se22XX is the executable file of the implemented CLI, the command to retrieve answers for question E15 in questionnaire Q9A would be:

```
./se22XX getquestionanswers --questionnaire_id Q9A --question_id E15 --format json
```

The following table outlines the supported parameters for each scope:

Scope	Requires Authentication (optional)	Mandatory Parameters	Corresponding REST API Call
login (optional)	No	--username --passw	/login
logout (optional)	Yes		/logout
healthcheck	Yes		/admin/healthcheck
resetall	Yes		/admin/resetall
questionnaire_upd	Yes	--source	/admin/questionnaire_upd
resetq	Yes	--questionnaire_id	/resetq
questionnaire	Yes	--questionnaire_id	/questionnaire
question	Yes	--questionnaire_id --question_id	/question
doanswer	Yes	--questionnaire_id --question_id --session_id --option_id	/doanswer
getsessionanswers	Yes	--questionnaire_id --session_id	/getsessionanswers

getquestionanswers	Yes	--questionnaire_id --question_id	/getquestionanswers
admin	Yes, must be admin	See next paragraph.	/admin/usermod /admin/users

The parameter `--source` carries the name of the JSON file containing the questionnaire. The JSON structure is provided in the annex.

The parameter `--format` is mandatory for all calls and can have the value "json" or "csv."

System Management

For administrative operations (scope: admin), the following parameters should be supported.

Parameter	Allowed Values	Mandatory Parameters	Function	Return Value
--usermod (optional)		--username --passw	Create a user or change password	
--username (optional)	Alphanumeric Latin characters	--passw		
--passw (optional)	Usual password characters (all characters excluding spaces)			
--users (optional)	Usual username characters (alphanumeric Latin characters)		Show user status	Execution result: username

Appendix

Example JSON format for the /admin/questionnaire_upd call

```
{
  "questionnaireID": "QQ000",
  "questionnaireTitle": "My first research questionnaire",
```

```
"keywords": ["footbal", "islands", "timezone"],
"questions": [
  {
    "qID ": "P00",
    "qtext": "What is your email?",
    "required": "FALSE",
    "type": "profile",
    "options": [
      {
        "optID": "P00TXT",
        "opttxt": "<open string>",
        "nextqID": "P01"
      }
    ]
  },
  {
    "qID ": "P01",
    "qtext": "What is your age?",
    "required": "TRUE",
    "type": "profile",
    "options": [
      {
        "optID": "P01A1",
        "opttxt": "<30",
        "nextqID": "Q01"
      },
      {
        "optID": "P01A2",
        "opttxt": "30-50",
        "nextqID": "Q01"
      },
      {
        "optID": "P01A3",
        "opttxt": "50-70",
        "nextqID": "Q01"
      },
      {
        "optID": "P01A4",
        "opttxt": ">70",
        "nextqID": "Q01"
      }
    ]
  },
  {
    "qID ": "Q01",
    "qtext": "What is your favorite color?",
    "required": "TRUE",
    "type": "question",
```

```
    "options": [
      {
        "optID": "Q01A1",
        "opttxt": "Green",
        "nextqID": "Q02"
      },
      {
        "optID": "Q01A2",
        "opttxt": "Red",
        "nextqID": "Q02"
      },
      {
        "optID": "Q01A3",
        "opttxt": "Yellow",
        "nextqID": "Q02"
      }
    ]
  },
  {
    "qID ": "Q02",
    "qtext": "Do you like football?",
    "required": "TRUE",
    "type": "question",
    "options": [
      {
        "optID": "Q02A1",
        "opttxt": "Yes",
        "nextqID": "Q03"
      },
      {
        "optID": "Q02A2",
        "opttxt": "No",
        "nextqID": "Q04"
      }
    ]
  },
  {
    "qID ": "Q03",
    "qtext": "What is your favorite football team?",
    "required": "TRUE",
    "type": "question",
    "options": [
      {
        "optID": "Q03A1",
        "opttxt": "Panathinaikos",
        "nextqID": "Q04"
      },
      {
```

```
      "optID": "Q03A2",
      "opttxt": "Olympiakos",
      "nextqID": "Q04"
    },
    {
      "optID": "Q03A3",
      "opttxt": "AEK",
      "nextqID": "Q04"
    }
  ]
},
{
  "qID ": "Q04",
  "qtext": "Have you lived on an island?",
  "required": "TRUE",
  "type": "question",
  "options": [
    {
      "optID": "Q04A1",
      "opttxt": "Yes",
      "nextqID": "Q05"
    },
    {
      "optID": "Q04A2",
      "opttxt": "No",
      "nextqID": "Q06"
    }
  ]
},
{
  "qID ": "Q05",
  "qtext": "Since you answered [*Q04A1] on the question [*Q04]: \
           How familiar are you with water skiing?",
  "required": "TRUE",
  "type": "question",
  "options": [
    {
      "optID": "Q05A1",
      "opttxt": "Not at all",
      "nextqID": "Q07"
    },
    {
      "optID": "Q05A2",
      "opttxt": "A little bit",
      "nextqID": "Q07"
    },
    {
      "optID": "Q05A3",
```

```
      "opttxt": "A lot",
      "nextqID": "Q07"
    }
  ]
},
{
  "qID ": "Q06",
  "qtext": "Are you a winter swimmer?",
  "required": "TRUE",
  "type": "question",
  "options": [
    {
      "optID": "Q06A1",
      "opttxt": "Yes",
      "nextqID": "Q07"
    },
    {
      "optID": "Q06A2",
      "opttxt": "No",
      "nextqID": "Q07"
    }
  ]
},
{
  "qID ": "Q07",
  "qtext": "How often do you go snow skiing?",
  "required": "TRUE",
  "type": "question",
  "options": [
    {
      "optID": "Q07A1",
      "opttxt": "Rarely or never",
      "nextqID": "Q08"
    },
    {
      "optID": "Q07A2",
      "opttxt": "Sometimes",
      "nextqID": "Q08"
    },
    {
      "optID": "Q07A3",
      "opttxt": "Regularly",
      "nextqID": "Q08"
    }
  ]
},
{
  "qID ": "Q08",
```

```
      "qtext": "Do you agree with the time change?",
      "required": "TRUE",
      "type": "question",
      "options": [
        {
          "optID": "Q08A1",
          "opttxt": "Yes",
          "nextqID": "Q09"
        },
        {
          "optID": "Q08A2",
          "opttxt": "No",
          "nextqID": "-"
        }
      ]
    },
    {
      "qID ": "Q09",
      "qtext": "Since you answered [*Q08A2] on the question [*Q08]: \
               Do you prefer summer or winter time?",
      "required": "TRUE",
      "type": "question",
      "options": [
        {
          "optID": "Q09A1",
          "opttxt": "Summer",
          "nextqID": "-"
        },
        {
          "optID": "Q09A2",
          "opttxt": "Winter",
          "nextqID": "-"
        }
      ]
    }
  ]
}
```

Example JSON format for the /questionnaire/:questionnaireID call

```
{
  "questionnaireID": "QQ000",
  "questionnaireTitle": "My first research questionnaire",
  "keywords": ["footbal", "islands", "timezone"],
  "questions": [
    {
      "qID ": "P00",
```

```
  "qtext": "What is your email?",
  "required": "FALSE",
  "type": "profile"
},
{
  "qID ": "P01",
  "qtext": "What is your age?",
  "required": "TRUE",
  "type": "profile"
},
{
  "qID ": "Q01",
  "qtext": "What is your favorite color?",
  "required": "TRUE",
  "type": "question"
},
{
  "qID ": "Q02",
  "qtext": "Do you like football?",
  "required": "TRUE",
  "type": "question"
},
{
  "qID ": "Q03",
  "qtext": "What is your favorite football team?",
  "required": "TRUE",
  "type": "question"
},
{
  "qID ": "Q04",
  "qtext": "Have you lived on an island?",
  "required": "TRUE",
  "type": "question"
},
{
  "qID ": "Q05",
  "qtext": "Since you answered [*Q04A1] on the question [*Q04]: \
           How familiar are you with water skiing?",
  "required": "TRUE",
  "type": "question"
},
{
  "qID ": "Q06",
  "qtext": "Are you a winter swimmer?",
  "required": "TRUE",
  "type": "question"
},
{
```

```
      "qID ": "Q07",
      "qtext": "How often do you go snow skiing?",
      "required": "TRUE",
      "type": "question"
    },
    {
      "qID ": "Q08",
      "qtext": "Do you agree with the time change?",
      "required": "TRUE",
      "type": "question"
    },
    {
      "qID ": "Q09",
      "qtext": "Since you answered [*Q08A2] on the question [*Q08]: \
               Do you prefer summer or winter time?",
      "required": "TRUE",
      "type": "question"
    }
  ]
}
```

Example JSON format for the / getsessionanswers/:questionnaireID/:sessioncall

```
{
  "questionnaireID": "QQ000",
  "session": "ATBP",
  "answers": [
    {
      "qID ": "P00",
      "ans": "<*>"
    },
    {
      "qID ": "P01",
      "ans": "P01A2"
    },
    {
      "qID ": "Q01",
      "ans": "Q01A2"
    },
    {
      "qID ": "Q02",
      "ans": "Q02A2"
    },
    {
      "qID ": "Q03",
      "ans": "Q03A3"
    },
```

```
    {
      "qID ": "Q04",
      "ans": "Q04A1"
    },
    {
      "qID ": "Q05",
      "ans": "Q05A2"
    },
    {
      "qID ": "Q06",
      "ans": "Q06A2"
    },
    {
      "qID ": "Q07",
      "ans": "Q07A3"
    },
    {
      "qID ": "Q08",
      "ans": "Q08A1"
    },
    {
      "qID ": "Q09",
      "ans": "Q09A2"
    }
  ]
}
```