Requirements draft

Group 06

March 26th 2020

Title: Creditoro - Functional
Non-functional Requirements

Institution: Syddansk Universitet
Det Tekniske Fakultet, Mærsk Mc-Kinney Møller Instituttet
Campusvej 55, 5230 Odense M

Uddannelse: Softwareteknologi

Term: 2. term

Project group: 06

Version: 0.0.1-DRAFT

figures/signatures/signatureJR.jpg
Jakob Rasmussen, jakra19@student.sdu.dk

Kenneth M. Christiansen, kechr19@student.sdu.dk

figures/signatures/signature_kepet19.png
Kevin K. M. Petersen, kepet19@student.sdu.dk

Kristian N. Jakobsen, kjako19@student.sdu.dk

figures/signatures/Signatur_mara816.png
Mathias N. Rasmussen, mara816@student.sdu.dk

Simon Jørgensen, sijo819@student.sdu.dk

Number of Pages: 4 pages

By signing this document every individual group member confirms that they have contributed equally to the project and is thereby liable for the content within this document.

1 Functional Requirements

Since the REST API and the Desktop client share functionality, by having the client call the REST API to handle the logic, we have merged the two functional requirements. The desktop client will call the REST API to handle the logic, thus only minimal logic will be required in the desktop client.

Desktop-client & REST API		
ID	**Name**	**Description**
D01	Login/out	You should be able to login and out of the system with ease by providing username and password.
D02	Search	You should be able to search for TV-show titles as well as persons
D03	Channel Browse	You should be able to browse channels registered in the system, and see which TV-shows are streamed on these
D04	Display credits associated with TV-program	You should be able to display all credits associated with a TV-program
D05	Create Channnel	The system admin should be able to create new channels.
D06	Add system admin	The system admin should be able to add other system admins.
D07	Add Channel admin	The system admin and channel admin should be able to add channel administrators to a channel. (Channel admin only for own channel)
D08	Add Producer	The system admin and channel admin should be able to add producers for a show (channel admin only for own channel)
D09	Create credit	The system admin, channel admin and producer should be able to create a credit (channel admin and producer only for own channel)
D10	Delete credit	The system admin and channel admin should be able to delete credits (channel admin only for own channel)
D11	Update credit	The system admin, channel admin and producer should be able to update and change credit (channel admin and producer only for own channel)
D12	Update person	System admin, channel admin and producer should be able to update a person, mainly which shows the person has helped produce
D13	Approve or disapprove credit	The system admin and channel admin should be able to approve or disapprove created credits before they are publicly available (channel admin only for own channel)
D14	Create royalty user	The system admin should be able to create a royalty user
D15	Update royalty user	The system admin and royalty user should be able to update royalty users
D16	Change language	You should be able to change the language of the user interface

Tabel 1: Desktop-client & REST API: Functional Requirements

EPG Poller		
ID	**Name**	**Description**
E01	Poll-data	The EPG Poller should be able to poll data from tvtid.dk
E02	Update Database	Update the database with credits

Tabel 2: EPG Poller: Functional Requirements

2 Non-functional Requirements

REST API		
ID	**Name**	**Description**
NR01	Supportability	Centralized error reporting should be available via a common interface (such as Sentry).
NR02	Performance	Commonly used API calls should respond within 300 milliseconds.
NR03	Scalability	The system should be able to handle 10K new users yearly for 25 years.
NR04	Scalability	The system should be able to handle 15K new credits yearly for 25 years.
NR05	Configuration	Data persistence time should be configurable, to auto-cleanup data older than the configured value (defaults to 25 years).
NR06	Availability	The system should start automatically after server restarts.
NR07	Usability	API documentation should be available via Swagger UI.
NR08	Security	The server that the REST API is hosted on should only allow login via SSH.
NR09	Security	The server that the REST API is hosted on should only allow connections from the outside on port 443 (https), 80 (http) and 22222 (ssh).
NR10	Installability	REST API should be deployable within a container (such as Docker).
NR11	Configurability	REST API should be configurable using **environment** files.
NR12	Authentication	REST API should handle authentication via token based authentication with it's clients. A token is valid for 2 hours, and is refreshed automatically after a request with the token when it's time to expire is less than an hour.

Tabel 3: REST API: Non-functional Requirements

Desktop-client		
ID	**Name**	**Description**
ND01	Supportability	Error reporting should be available via pop-out box
ND02	Performance	Commonly used buttons should work within 300 milliseconds
ND03	Scalability	The system should have a nice overview of all credits
ND04	Scalability	The client should be able to query search results in 5 seconds
ND05	Configuration	Configuration should be configurable using **environment** files and GUI
ND06	Availability	The system should start automatically after server restarts.
ND07	Authentication	The system should redirect to the login page when receiving a HTTP 401 (unauthorized) response from the REST API.
ND08	Inactivity	The desktop-client should automatically detect if the authentication token is expired and redirect to login page.

Tabel 4: Desktop Client: Non-functional Requirements

EPG Poller		
ID	**Name**	**Description**
NE01	Supportability	Centralised error reporting should be available via a common interface (such as Sentry).
NE02	Performance	It should be able to poll data every hour and finish within 15 min to update the data
NE03	Scalability	The system should be able to poll 15K new shows every year yearly, for 25 years.
NE04	configuration	How often the poller is run, where to poll data from and where to post data too should be configurable
NE05	Availability	The system should start automatically after server restarts.
NE06	Usability	The system should warn about show's without credit
NE07	Security	The server that the EPG poller is hosted on should only allow connections from the outside on port 22221(ssh).
NE08	Configurability	The EPG Poller should be configurable using **environment** files.

Tabel 5: EPG Poller: Non-functional Requirements

3 Acceptance Criteria

3.1 D01 - Login/out

As PO I want users to be able to authenticate so that we can limit functionality to the right users.

3.1.1 Login

Given the user is not signed in.
And the user clicks the login button.
When the user fills out a valid username.
And the user fills out a valid password.
And the user clicks submit (or enter hotkey)
Then a `post` request is sent to the server with the credentials over https.
And the user is then authenticated and a temporary token is returned to the sender.
And user is now signed in.