Technical Specification document for the speech transcription platform user interface

Prepared by: Neil Kleynhans

Date: 25 July 2016

Document Reference: DAC-STP-UI-TECHSPEC-REV00

# Purpose

This document provides design specifications for the DAC platform user interface (UI) that is coupled to application server (AppServ) developed by NWU. The UI should provide a client facing interface that is user-friendly, robust and visually appealing.

# Requirements

The UI should:

* Deliver the software to MuST in a open format or released on web-accessible repository (such as GitHub or BitBucket).
* Release the software under the license specified by MuST Research Group.
* Focused exclusively on the Chrome desktop web browser only. A specific version of Chrome can be stipulated after consultation.
* Be developed in JavaScript and or any Javascript framework.
* Make use of the MuST provided application interface (API) to facilitate the displaying, capture and storage of user provided data.
* Not be cluttered and tend towards a minimalistic representation.
* Have clear visual flows and communicate the flows to the user.
* Provide basic input data verification and perform user input coherence checking where applicable.
* Communicate all errors effectively to the user.
* Scale to the various user-specific viewport dimensions of their desktop.
* Make use of CKEditor or related free open source JavaScript editor for the transcription editor
* Display audio with WaveSurfer or related free open source audio visualisation JavaScript software.
* Link CKEditor (or related editor) and WaveSurfer (or related audio visualisation) to audio-enable the editor

# NWU Demo

NWU will provide a UI JavaScript demo that captures the functionality that is required. When referring to client interfaces please make sure you are familiar with the demo first.

# Client interfaces

This section provides a breakdown of client interfaces.

|  |  |
| --- | --- |
| **Web browser Page** | **Description** |
|  |  |
| Welcome | Display various interface options which can be used to navigate to the desired interface |
| Admin | Interface that is used to manage users. Each interface will have its own admin interface which separates user’s role. |
| Project/Collator | Interface that provides project management and collation functionality. |
| Editor | Interface that allows the navigation and management of assigned tasks, editing of transcriptions and the use of speech services. |

## Input field character limits

* Username: 30 Characters
* Name: 30 Characters
* Surname: 30 Characters
* Email: 50 Characters
* Project name: 32 Characters
* Project Category: 36 Characters
* Audiofile: 128 Characters
* Error status: 128 Characters

## Welcome

Welcome landing interface:

* Displays clickable elements which can be used to navigate to other interfaces: admin (if admin separate), project and editor.
* The clickable elements should contain a label, brief description of the interface and a tooltip.
* The page should display a welcome message and contain branding specific to the client.

## Admin

The admin interface will be used to manage users from the project and editor roles. The users for project and editors are stored separately which enforces role separation. The admin interfaces can be separate or form part of the role interfaces (project or editor).

### Layout and elements

Login:

|  |  |  |
| --- | --- | --- |
| Item | Elements | AppServ Calls |
| Login | * Username input field * Password input field * Role selection * Login submit button * Tooltip for username and password * Check both username and password were supplied | wsgi/projects|editor/admin/login |
| Remove previous session | Used if the user forgot to logout and token is still valid. A user can only login in once per session. Current credentials are needed to remove previous stale tokens.   * Username input field * Password input field * Logout button * Tooltip for username and password * Check both username and password were supplied | wsgi/projects|editor/admin/logout2 |

User’s display:

|  |  |  |
| --- | --- | --- |
| Item | Elements | AppServ Calls |
| Login display | Display all users that belong to the specific role -- project or editor. Display other items listed below   * Clickable elements that display user information * Button to add users * Button to delete user * Button to update user | wsgi/projects|editor/admin/get\_users |
| Add new user | * Button to bring up add user interface * Interface should request: username, name, surname and email address * Check that username is unique * Check password is valid * Check valid email * Use input limits * Role selection | wsgi/projects|editor/admin/add\_user |
| Delete user | * Select user * Button to delete user * Confirm that the user wants to delete user * Proceed button * Cancel button | wsgi/projects|editor/admin/del\_user |
| Update user information | * Select user * Display current user information * Name input field * Surname input field * Email input field * Button to submit request * Use inputs limits | wsgi/projects|editor/admin/update\_user |
| Change admin’s password | * Button to request change of password * Password input field * Verify password input field * Dialog to confirm reset action * Button to proceed * Button to cancel | wsgi/projects|editor/admin/change\_password |
| Reset user’s password | In the current model the user will request the administrator to change the password.   * Select user * Button to submit change request * Password input field * Verify password input field * Dialog to confirm reset action * Button to proceed * Button to cancel | wsgi/projects|editor/admin/reset\_password |

Logout:

|  |  |  |
| --- | --- | --- |
| Item | Elements | AppServ Calls |
| Logout | User should be able to logout on any follow-up interface   * Logout button | wsgi/projects|editor/admin/logout |

### Suggestions

* The welcome page could display the project and editor admin interfaces
* The admin interface could be requested from either the editor or project interfaces

## Project / Collator

The project interface will be used by the project manager(and collator) to create a new project or browse previously created projects. When creating a new project the the project manager will upload the audio, either automatically or manually segment the audio, save segments and assign the segments to editors. Assigning a task to the editor transfers ownership to the editor. The ownership model controls read/write access to the task. The project will be assigned to a collator at this stage.

The project manager can view created projects which are assigned to the collator but not access them. When selecting current projects the collator can view the status of the individual tasks and view the editor’s transcriptions. If ownership has been transferred back to the collator then they can edit the transcription.

### Notes

The project manager/collator will have to log in as both the project manager and editor API. The UI will have to keep track of two tokens. The editor token is needed to edit transcriptions.

### Layout and elements

Login:

|  |  |  |
| --- | --- | --- |
| Description | Elements | AppServ Calls |
| Login | * Username input field * Password input field * Login button * Tooltip for username and password * Check both username and password were supplied | wsgi/projects/login  wsgi/editor/login |
| Remove previous session | * Username input field * Password input field * Logout button * Tooltip for username and password * Check both username and password were supplied | wsgi/projects/logout2 |

All projects display:

|  |  |  |
| --- | --- | --- |
| Description | Elements | AppServ Calls |
| Display all projects | There are two types of projects: owned by user and not owned by user (just created)   * Clickable elements displaying projects * Split created projects but not accessible and projects that can be edited * Display project name, category, locked status and error status * Calendar to filter projects by date * Project category filter * Button to add new project * Button to delete project * Button to unlock project | wsgi/projects/listprojects  wsgi/projects/listcreatedprojects  wsgi/projects/listcategories |
| Create new project | * New Interface * Project name input field * Project category dropdown field * Button to save project * Check project name is unique * Check category selected | wsgi/projects/createproject |
| Delete project | * Select project * Dialog to confirm delete * Button to cancel * Button to submit delete request | wsgi/projects/deleteproject |
| Unlock project | Assumes speech job locked project.   * Select project * Verify project is locked * Dialog to confirm unlocking of project * Button to cancel * Button to submit unlock request | wsgi/projects/unlockproject |
| Change password | * Button to request change of password * Password input field * Verify password input field * Dialog to confirm reset action * Button to proceed * Button to cancel | wsgi/projects/changepassword |
| Logout | * Logout button | wsgi/projects/logout  wsgi/editor/logout |

Project display (once user has clicked on project):

|  |  |  |
| --- | --- | --- |
| Description | Elements | AppServ Calls |
| Return to projects | * Button to return to main project interface * Verify no changes should be saved * Dialog to ignore saves * Button to cancel * Button to proceed | None |
| Assign collator | Project manager should select the collator. During `assign tasks` the project ownership will be moved to the collator.   * Drop down field of collators | wsgi/projects/users |
| Load project | Get project information from AppServ call   * Should happen when user selected project | wsgi/projects/loadproject |
| Upload audio | * File select input field with label * Verify audio file is OGG * Button to submit audio to server * If audio present: dialog to confirm audio override * If tasks are present tell user to delete project and create a new project | wsgi/projects/uploadaudio |
| Get Audio | Initialize Wavesurfer to download audio from AppServ call if project has audio already   * Audio control bar should be added * Playback, pause, stop, skip forward, skip backward, tempo change, zoom | wsgi/projects/getaudio |
| Manually created segments/tasks | * Use wavesurfer regions plugin to create segments on the canvas * Display tasks in time order below * Delete task if segment is deleted * Confirm that the user wants to delete the task * Shift segment and task order if moved in time order by user | None |
| Diarize project | Request the speech server to create tasks. Audio upload must happen before this. This should lock the project and move the user to the projects interface   * Button to request diarize * Check tasks have not been created by the user * Check audio has been uploaded * Dialog to confirm requested diarize | wsgi/projects/diarizeproject |
| Task details assignment | Area that shows the tasks that have been created/saved for the project   * Start and end times field * Editor drop down field * Language drop down field * Speaker input field * Ownership boolean: 0=editor or 1=project * Transfer to editor * Button to remove task * Dialog to confirm task removal | wsgi/editors/users  wsgi/projects/languages |
| Save project | * Button to save project details to server * Hide if no tasks are defined | wsgi/projects/saveproject |
| Assign project | Assumes save has happened before this is called.   * Button to assign tasks to editors * Hide if no tasks have been defined * Check that tasks have been defined * Check projects have been saved * For each task check that all details have been selected by user * Check the segments are non-overlapping and touching * Check segments are contiguous and spanning entire duration * Check collator has been set | wsgi/projects/assigntasks |
| Update project information | Can only occur after a task has been assigned   * Button to update project information * Can change: project name, project category, collator * Can change: editor, language, speaker and ownership | wsgi/projects/updateproject |
| Logout | User should be able logout at any stage   * Logout button * Check that nothing should be saved | wsgi/projects/logout |

Editor:

|  |  |  |
| --- | --- | --- |
| Description | Elements | AppServ Calls |
| Transfer collator to editor interface | The project manager can view an editor’s transcription by clicking on a button. This will transfer the project manager to an editor UI -- same as the editor defined in the following sections. Ownership is important here: if the editor still has ownership then the project manager can only view the transcriptions not edit at all.   * Ownership information should be transferred to Editor UI i.e. disable all functionality that can modify the text |  |

## Editor

The editor interface is used by the editor to browse the assigned tasks and transcribe the audio that is associated to each task. The editor can manually edit or make use of the speech services to help in transcribing. The editor marks the task as done when finished editing which transfers ownership back to the collator. If the editor has ownership then the editor can edit the document else it becomes read-only.

### Layout and elements

Login:

|  |  |  |
| --- | --- | --- |
| Description | Elements | AppServ Calls |
| Login | * Username input field * Password input field * Login button * Tooltip for username and password * Check both username and password were supplied | wsgi/editor/login |
| Remove previous session | * Username input field * Password input field * Logout button * Tooltip for username and password * Check both username and password were supplied | wsgi/editor/logout2 |

Task display:

|  |  |  |
| --- | --- | --- |
| Description | Elements | AppServ Calls |
| Load tasks belonging to editor | Occurs after login   * Clickable elements displaying tasks * Display task date assigned, task last modified, task locked status, task ownership, task error status, project name, category * Calendar to filter task by date * Task filter by project category * Button unlock task * Button clear task error | wsgi/editor/loadtasks |
| Unlock Task | Assumes task currently locked by speech job   * Select task * Verify task is locked * Dialog to confirm unlocking of task * Button to cancel * Button to submit unlock request | wsgi/editor/unlocktask |
| Clear task error | A task error has occurred and must be cleared   * Select task * Verify task has an error * Dialog to confirm unlocking of task * Button to cancel * Button to submit unlock request | wsgi/editor/clearerror |
| Task done | Assign task is done and transfer ownership   * Select task * Verify task is not locked or in error * Dialog to confirm task is done * Button to cancel * Button to submit unlock request | wsgi/editor/taskdone |
| Logout | * Logout button | wsgi/projects/logout |

Editor:

|  |  |  |
| --- | --- | --- |
| Description | Elements | AppServ Calls |
| Return to tasks | * Button to return to main editor interface * Verify no changes should be saved * Dialog to ignore saves * Button to cancel * Button to proceed | None |
| Get Text | Load text into CKEditor | wsgi/editor/gettext |
| Get Audio | Initialize Wavesurfer to download audio from AppServ call. Audio control should be added to CKEditor   * Audio control bar should be added * Playback, pause, stop, skip forward, skip backward, tempo change, zoom | wsgi/editor/getaudio |
| Save text | * Button to save text * Dialog to confirm unlocking of task * Button to cancel * Button to submit unlock request | wsgi/editor/saveaudio |
| Diarize | Request diarize speech service   * Button to request diarize * Verify there is no text * Dialog to confirm request * Button to cancel * Button to proceed * Return to main task page as the task will be locked | wsgi/editor/diarize |
| Recognize | Request recognize speech service   * Button to request recognize * Verify there is no text or at least a time region with no text * Dialog to confirm request * Button to cancel * Button to proceed * Return to main task page as the task will be locked | wsgi/editor/recognize |
| Align | Request align speech service   * Button to request align * Verify there is text * Dialog to confirm request * Button to cancel * Button to proceed * Return to main task page as the task will be locked | wsgi/editor/align |

### CKEditor

CKEditor will be modified to enable audio support. These include:

* Audio playback: play, pause, stop, skip forward, skip backward
* Insert timestamp
* Click on timestamp and shift audio to location
* Double-click on text word and move to audio location

NWU has written plugins for these and will provide them to GeckoTech.