

**Customer Flow Management System (CFMS) Modernization Last updated on: April 25, 2018**

Version: 1.4

Table of Contents

[1 Proposal Evaluation Criteria 1](#_Toc512330811)

[2 High Level Requirements 1](#_Toc512330812)

[3 Phase 1: Proof of Concept 2](#_Toc512330813)

[4 Phase 2: Build Back End 2](#_Toc512330814)

[5 Phase 3: Build Front end 3](#_Toc512330815)

[6 Appendices 4](#_Toc512330816)

[6.1 Appendix 1: Data Model 4](#_Toc512330817)

[6.2 Appendix 2: Rest Endpoint Details 4](#_Toc512330818)

[6.3 Appendix 3: CSR Screens 5](#_Toc512330819)

[6.3.1 The Main Screen 5](#_Toc512330820)

[6.3.2 The Add Customer / Search Services Screen 7](#_Toc512330821)

[6.3.3 The Serve Customer Screen 11](#_Toc512330822)

[6.3.4 The Feedback Screen 12](#_Toc512330823)

[6.4 Appendix 4: GA Status Screen 13](#_Toc512330824)

[6.5 Appendix 5: Smartboard Digital Signage Screens 13](#_Toc512330825)

# Proposal Evaluation Criteria

* Your prior experience with queue management systems specifically where the queue display/status is dynamically updated based on other logged in users interactions. (10 points).
* Some experience working with BC Government (10 points).
* Two years of experience using in Python, Javascript, Flask, PostgreSQL and experience contributing code to public code repositories (10 points).
* One year of experience with OpenShift (5 points).
* Two years of experience with Docker, Jenkins and/or other container based systems (20 points).
* Two years of experience in BDD and Unit Testing (10 points)
* Your ability to satisfy the Acceptance Criteria on or before [15 August, 2018] (10 points).

# High Level Requirements

Using the Government’s Open Shift DevOps infrastructure and the technology stack adopted by Service BC, replace the current QSystem java application (<https://github.com/bcgov/sbc-qsystem>) with an application designed to run in a container environment using the attached existing Business Requirements, current system UI and current system business flow.

1. Must use the BC Government OpenShift Environment
2. Must use the following Policy Framework: https://github.com/bcgov/BC-Policy-Framework-For-GitHub
3. All new code must be written using the Apache 2.0 LICENSE and be placed in the BCGov section of github.
4. Must use Jenkins for deployments. See here for templates: https://github.com/BCDevOps/openshift-tools
5. Must use Sonar-Cube to ensure no vulnerabilities or Code Smells through the pipeline.
6. Must build BDD tests and unit tests into the design. BDD tests should use the following: https://github.com/BCDevOps/BDDStack, or any updates recommended in the #bddstack devopspathfinder Slack channel
7. Must use a database migration tool (we currently use https://flywaydb.org/)
8. Must use the KeyCloak security framework (https://github.com/mbucknell/keycloak-python-poc)
9. Must use the government’s most recent Bootstrap styles, keeping the UI as similar to the current UI as possible.
10. Must build using an API based architecture

# Phase 1: Proof of Concept

Phase 1: Create a container friendly proof of concept code in the Government’s Openshift environment which includes but is not limited to:

1. JavaScript with Vue.js framework for front-end components.
2. Python code and Flask framework for the back-end components.
3. A Python/Flask (if possible) code library which implements the Publish / Subscribe pattern. A possibility is Flask-WebSub 0.2.1 (https://pypi.python.org/pypi/Flask-WebSub).
4. PostgreSQL (or alternative, if needed) database implementation.
5. Stubs for BDD and Unit tests
6. Python library using Hibernate or other suitable technology, or Openshift container features that will maintain application availability and state during Openshift server / software upgrades, and pod deployment/redeployment.
7. Test the horizontal scalability and high-availability failover within the OpenShift Framework.
8. Ensure the solution meetings all RedHat Guidelines.
9. Develop new OpenShift build and deploy templates for the project.
10. Transfer knowledge and documentation of the above to Service BC technical staff

***Completion on 31st of May, 2018 for Payment of $16,000.***

# Phase 2: Build Back End

Phase 2: Build the Back End components

1. Verify current data model and adjust as needed (See Appendix 1: Data Model)
2. Develop an API specification that the back end will implement, to provide services for the front end.
3. Build REST endpoints for underlying data model and moving clients through workflow. The required endpoints are (See Appendix 2: Rest Endpoint Details for required endpoint details):
4. List services
5. List service categories
6. List channels
7. List customers
8. Invite next customer
9. Invite specific customer
10. Update customer data
11. Provide feedback / bug report
12. Write automated unit tests for back-end components
13. Transfer knowledge and documentation of the above to Service BC technical staff

**Completion expected in 30st of June, 2018 for payment of $18,000.**

# Phase 3: Build Front end

Phase 3: Build the Front End components, consisting of:

1. The Customer Service Representative (CSR) User Interface
2. A Government Agent (GA) status screen, to indicate the status of CSRs and citizens in an office
3. A Smartboard system, to display information to citizens waiting for services

Specifically, Phase 3 is to:

1. Design the front end layout for CSRs to serve clients based on current system and government Bootstrap based style sheets. A list of required screens is in Appendix 3: CSR Screens.
2. Link front end actions with back end REST end points
3. Implement a GA status screen. See Appendix 4: GA Status Screen for details.
4. Write Smartboard code to allow the display of information (general and status) to citizens waiting for service. See Appendix 6: Smartboard Screens for details.
5. Write BDD tests according to the specifications in Section 1 above.
6. Transfer knowledge and documentation of the above to Service BC technical staff

***Completion expected in 15th of August, 2018 for $28,000.***

There may be a possibility of an extension to add an Administrative Portal to simplify the addition of services, additional offices, users, etc.

# Appendices

## Appendix 1: Data Model

The Data model will be similar to the current system. There will be changes to the data model and we will provide them to the successful proponent.

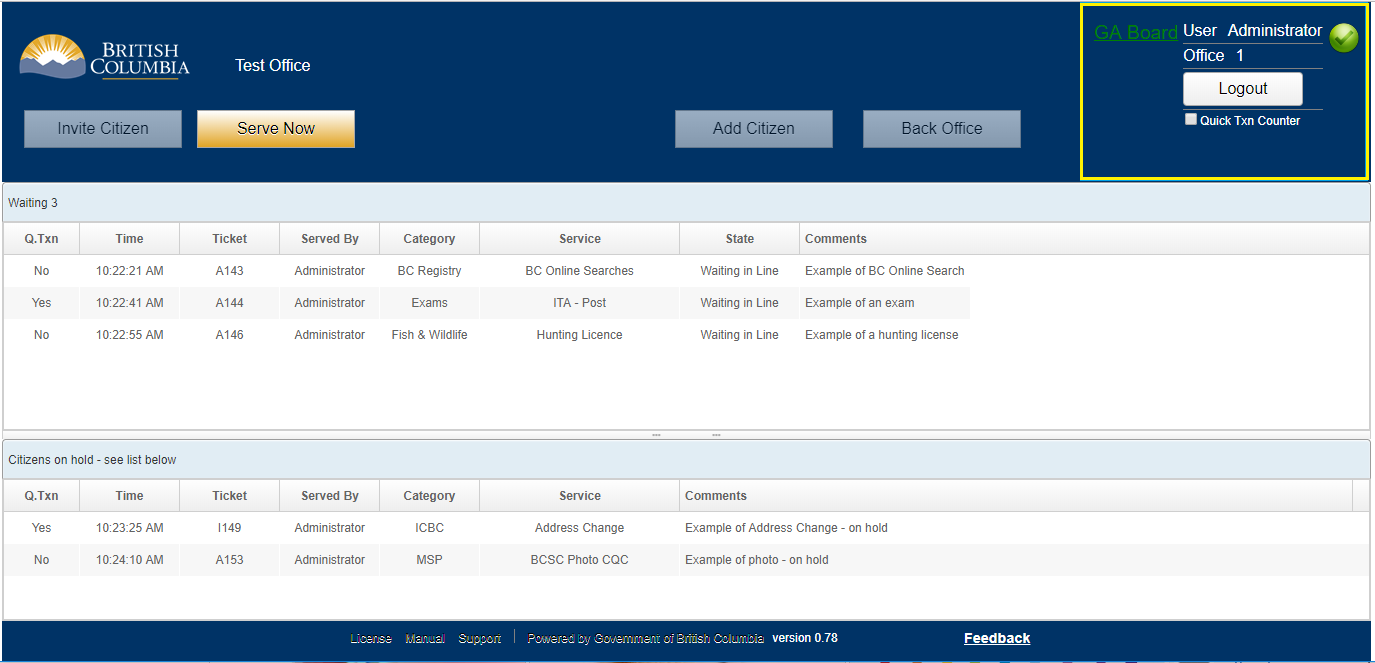
## Appendix 2: Rest Endpoint Details

The following are details of the REST endpoints required:

1. List services
2. No argument: Return a list of all services. Data required: ID, category and name
3. Argument: Office ID. Data required: As above, but only return the services offered by that office
4. List service categories
5. No argument: Return a list of all service categories: Data required: ID, name
6. List channels
7. No argument: Return a list of all service channels: Data required: ID, name
8. List customers
9. No argument: Return a list of all customers in an office. Data required:
10. ID, Quick Transaction Status, Time ticket created, Ticket number, CSR that last served the citizen this visit, service category, service name, customer state, customer comments
11. Invite next customer
12. No argument: Serve the next customer to be served in the office.
13. **Note:** The next customer to be served is the first customer in the wait queue whose Quick Transaction status matches the Quick Transaction Status of the CSR inviting the next customer
14. If there is no customer in line whose Quick Transaction Status matches the Quick Transaction Status of the CSR inviting a customer, then the first customer in line is the next customer to be served
15. Action: Return data as per 4, for display by the front end. Change customer status.
16. Invite specific customer
17. Argument: Customer ID.
18. Action: Return data as per 4, for display by the front end. Change customer status
19. Update customer data
20. Arguments: Customer ID, other arguments to be determined.
21. Action: Update customer data as determined by arguments
22. Provide feedback / bug report
23. Arguments: To be determined
24. Action: Send CSR feedback to Slack channel.

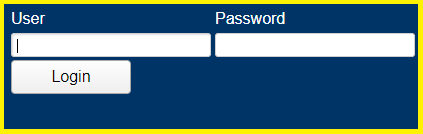
## Appendix 3: CSR Screens

### The Main Screen

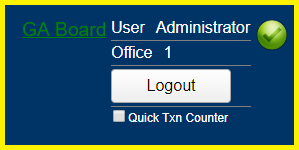


**Please note:** The top right portion of the screen, outlined in yellow, has two forms.

Before login:



After login:

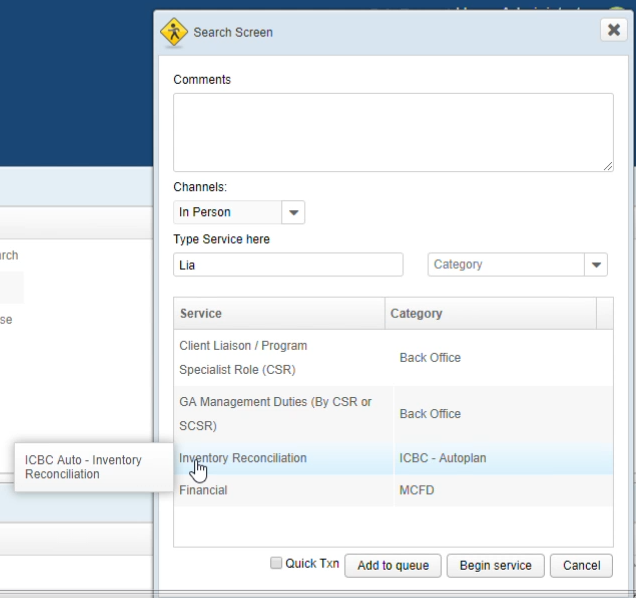


Functionality:

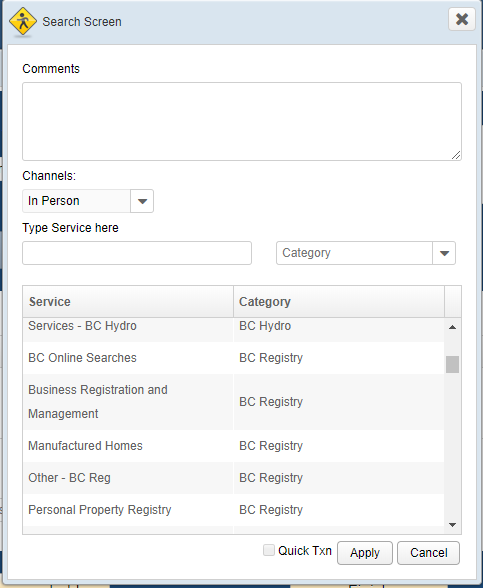
1. When the main screen is displayed after login, the Invite Citizen, Add Citizen, and Back Office buttons will be enabled and display as the Serve Now button is displayed. The Serve Now button will not be displayed.
2. The screen as shown above is what will appear if the CSR closes their browser, or if the system crashes, when the CSR was in the middle of serving a citizen. In such a case, when that CSR reloads the system, the only button they will be allowed to click will be the Serve Now button. The functionality of this button is described below.
3. User and Password allow user text to be entered
4. The Login button takes the user’s input, verifies it, and logs the user into the system
5. Verification will be done using Keycloak, optionally using Siteminder
6. The Logout button will log the user out of the system
7. The Quick Transaction Counter checkbox may be checked and unchecked by a CSR
8. When the Quick Transaction Counter checkbox is checked, it indicates that this CSR will only serve citizens in line that have a quick transaction service, unless there are no citizens in line waiting for a quick transaction. In that case, the Quick Transaction CSR will serve the next citizen in line.
9. The Invite Citizen button will invite the next citizen in line, whose Quick Transaction Status matches the Quick Transaction status of the CSR. The system will then display the Serve Citizen Screen (Section 5.3.3) containing information for this next citizen.
10. If no citizens in line match the quick transaction status of the CSR, then the Invite Citizen button will invite the next citizen in line, regardless of Quick Transaction status.
11. If there is no citizen in line, clicking the Invite Citizen button will display an error message for the CSR.
12. Mechanisms will be put in place so that if two CSRs in an office click on the Invite Citizen button at the same time, these CSRs will not invite the same citizen in line
13. The Serve Now button will bring up the Serve Citizen Screen (Section 5.3.3) containing information for the citizen they were serving when their system went down.
14. The Add Citizen button will bring up the Add Citizen / Search Services screen (Section 5.3.2)
15. The Back Office button will function in an identical manner to the Add Citizen button, except when it is pressed, the Back Office channel will be pre-selected.
16. A CSR may click any citizen in the wait queue, to serve that citizen. The system will then display the Serve Citizen Screen (Section 5.3.3) containing information for the citizen clicked on.
17. Mechanisms will be put in place so that if two CSRs click the same citizen in line, one CSR will serve the citizen, while the other CSR will see a message on their screen indicating another CSR is serving the citizen. It is desirable that the name of the CSR serving the citizen is displayed in the message.
18. Mechanisms will be put in place so that if one CSR clicks the first citizen in line, and a second CSR clicks the Invite button, one CSR will serve the citizen, while the other CSR will get a message indicating another CSR is serving the citizen. It is desirable that the name of the CSR serving the citizen is displayed in the message.
19. A CSR may click on any citizen in the hold queue, to serve that citizen. The system will then display the Serve Citizen Screen (Section 5.3.3) containing information for the citizen clicked on.
20. Mechanisms will be put in place, similar to 6 and 7, so that if two CSRs click on the same citizen in the hold queue, one will serve them, and the other will see an error message.
21. The Feedback link (bottom right of the screen) will bring up the Feedback Screen (Section 5.3.4).

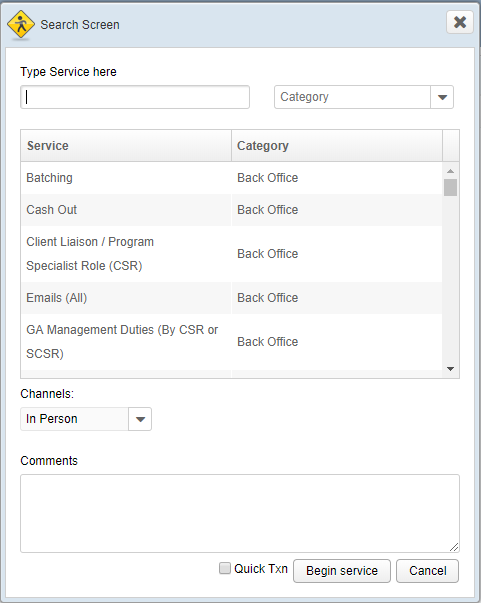
### The Add Customer / Search Services Screen

Reception office view (Service selection at bottom of screen):



Reception office view when CSR clicks on Next Service in Serve Citizen screen (Service selection at bottom of screen):



Non-reception office view (Service selection at top of screen):

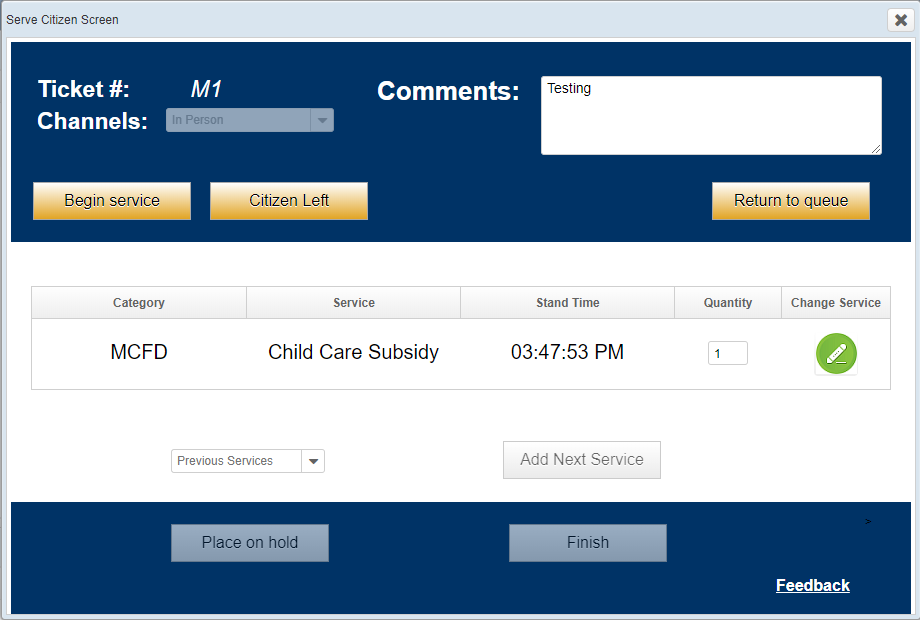
Functionality:

1. The CSR may type in the comments box. Comments are stored ONLY for the duration of a customer visit. Clicking on the Citizen Left or the Finish button will delete comments in the database. Comments are displayed when a citizen is in the wait or hold queues.
2. The CSR may select what channel is being used to serve the citizen by clicking in the Channels drop down box.
3. The user may type text in the Type Service here box. Typing in this box will filter the list of services displayed, to only those services containing whose name or tooltip contain the text string typed in this box.
4. The user may select a Category from the Category drop down box. Selecting a category will filter the list of services displayed, to only those services in the selected category.
5. The user may type in the Category drop down box. Typing in this box will do two things:
6. Filter the list of categories displayed to only those categories containing the text string typed in this box.
7. Filter the list of services to only those services that match one of the categories filtered in A) above.
8. The user may click on any service displayed in the Service area of the screen. Clicking on a service will cause the name of that service to be displayed in the Type Service here box.
9. **Note:** Hovering over a service will cause a tooltip to display, describing that service for the CSR
10. Checking the Quick Txn box will indicate that the citizen is here for a Quick Transaction, and should be served by a Quick Transaction CSR.
11. Clicking on the Add to queue button will add the citizen to the Wait Queue
12. **Note:** The add to queue button will not display for non-reception offices. CSRs in non-reception offices do not have the option of adding citizens to the queue.
13. **Note:** When a citizen wants a second service, the Add Citizen / Search screen will also display. In such cases the Add to queue button will not display, and the Begin Service will be replaced with a button labelled Apply, with different functionality than the Begin Service button.
14. Clicking on the Begin Service button will cause the Serve Citizen Screen (Section 5.3.3) containing information for this citizen to appear. The state will be as if the CSR had clicked the Begin Service button.
15. Clicking on the Apply button, for a citizen’s Next Service, will cause the Serve Citizen Screen (Section 5.3.3) to appear, containing the customer’s previous information, except information for their new service will be displayed.
16. Clicking on the Cancel button will cause no citizen or new service to be added.

### The Serve Customer Screen

**Note:** Non-reception offices do not see the Begin Service, Citizen Left, or Return to queue buttons.

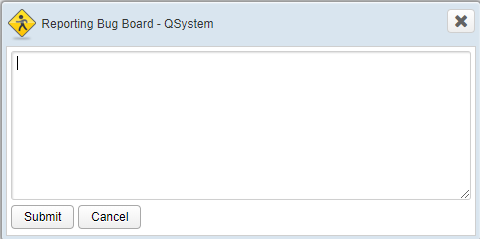
**Note:** When Begin Service is clicked, an Inaccurate Time checkbox appears below the Finish button.



Functionality:

1. The CSR is allowed to update comments in the Comments field.
2. The CSR will only click the Begin service button once a citizen has arrived at their station, after they have been invited.
3. Clicking on Citizen Left will button cause the Citizen to be removed from the system, and any comments for this citizen to be deleted from the database.
4. Clicking on the Return to Queue button will cause the citizen to be placed back in the wait queue.
5. The CSR may enter a numeric value in the Quantity field.
6. Clicking on the Change Service button will cause the Search / Add citizen Screen (Section 5.3.2) to appear, and allow the service they are receiving to be corrected.
7. The system will maintain a list of services this citizen has received this visit. Clicking on the Previous Citizen dropdown box will allow the citizen to continue receiving a previous service.
8. Clicking on the Add Next Service button will cause the Search / Add citizen Screen (Section 5.3.2) to appear, and allow the citizen to receive an additional service.
9. Clicking on the Place on Hold button will place the citizen in the hold, rather than the wait queue.
10. Clicking on Finish will stop services for the Citizen, and delete any comments that are stored in the database for this citizen.
11. Once the CSR clicks on Begin Service, an Inaccurate Time checkbox will appear below the Finish button. Checking this checkbox will set a flag in the database indicating that the time values for the last service this citizen received are inaccurate, most likely because a CSR went on break and forgot to click Finish.
12. Clicking on the Feedback button will bring up the Feedback Screen (Section 5.3.4)

### The Feedback Screen

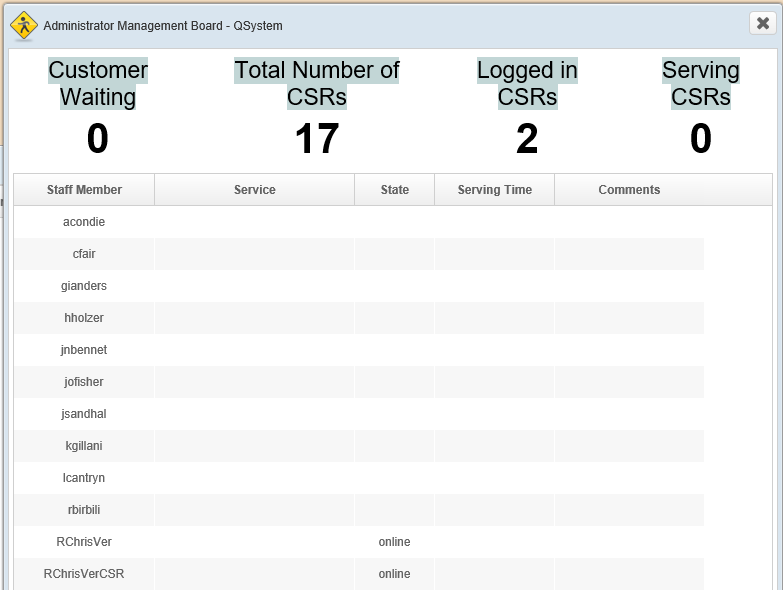


Functionality:

1. Clicking on the Submit button will send text entered in the text box to one or more of the following possible destinations:
2. The #sbc-cfms Slack channel (this is currently the case)
3. The Service BC Service Now application
4. Clicking on the Cancel button will cause the window to be closed, and no action to be taken.

## Appendix 4: GA Status Screen

The following is the current GA Status Screen:



Functionality:

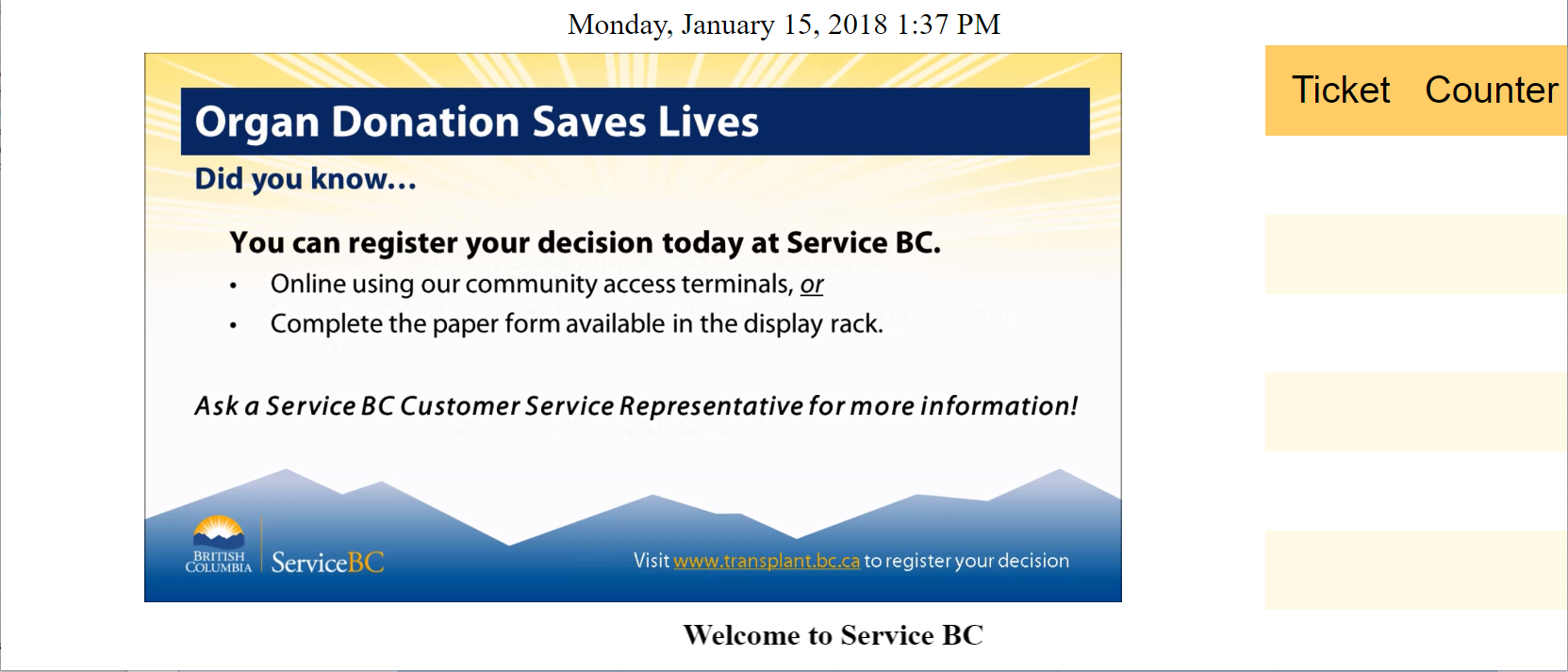
1. This screen will refresh every second, or optionally whenever a value changes, with the values shown above being displayed.

## Appendix 5: Smartboard Digital Signage Screens

Digital signage has the following requirements:

1. Access must be via non-authenticated web URL, which will be blocked access by NGINX based on IP.
2. Access will be via Raspberry Pi running Chromium.
3. The current URL Format is: https://<server>/qsmartboard/?office\_id=61
4. Must allow for Non-Reception and Reception Office Views
5. Reception Office View must include number of citizens waiting, excluding tickets with a category of Back Office
6. Must have the ability to play and change .MP4 video files

Screen for Call by Ticket, Reception Office:



Screen for Call by Name, Reception Office:



Screen for Non-Reception Office:



Functionality:

1. Current date and time must be display at the top of the screen
2. An .mp4 movie, supplied by Service BC, must display in the centre of all displays
3. Text (marquee), specific to each office, to scroll across the bottom of the screen. This is optional and can be added later.
4. The waiting panel and count values are taken from the wait queue, but exclude all Back Office category service tickets.