QMATIC

DEPLOYMENT GUIDE

MOBILE TICKET

224.04M

Copyright notice

ⓘ Please send any feedback or questions about this manual to documentation@qmatic.com.

Contents

1. Introduction 1

Overview 2
Document Structure 3
Architecture 4
Limitations 4
Browsers and cookies 4
Software deliverables 5

2. Requirements and Preparations 7

System requirements 8
Orchestra preparations 8
URLs for notifications and QR codes 10

3. Installation and Basic Configuration for API Gateway 13

Installation API Gateway 14
Basic configuration API Gateway 14

4. Installation and Configuration 15

Installation node.js 16
Configuration of proxy/webserver 16
Install Mobile Ticket 17
General configuration for Mobile Ticket 17
Enable visit delay 22
Configure appointment check-in 23
Configure open hours 23
Localization 25
Change logotype 27
Enable HTTPS / SSL 27
Security precautions 30
Install Mobile Ticket as a service 31
Upgrade 32
Database configuration 33

5. Troubleshooting 35

Troubleshooting guide 36
Frequently Asked Questions 37

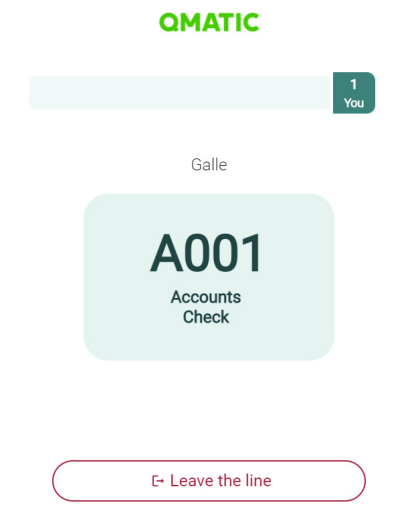
6. Common Use Cases and Walk-through 39

Use cases 40
End user walk-through 41

Introduction

Overview 2
Document Structure 3
Architecture 4
Limitations..... 4
Browsers and cookies 4
Software deliverables 5

Overview



With the Mobile Ticket application, customers can request tickets from their mobile device, via a web page/web application, using a direct URL, or for example a QR code. They can they wait to be called wherever they want, monitoring their position in line on their device.

Main features

- Seamless entry with QR codes and URLs
- Possibility for customer to add phone numbers
- Can be combined with SMS notifications and Customer Feedback
- Check-in for appointments
- Can be linked to Google Analytics
- Support for multi-language and rtl
- Branding for colors and logo
- One-time password and other security features that prevents misuse of system and hacker attacks
- Integrations with communication platforms for remote serving

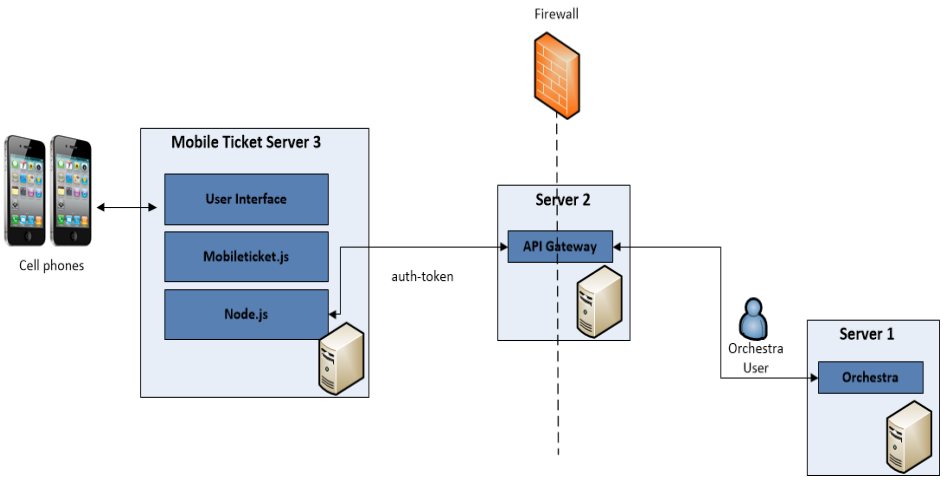
Document Structure

The goal of this manual is that you should get the Mobile Ticket solution up and running, and be able to configure it the way you want.

1. In “Architecture” on page 4 and “Browsers and cookies” on page 4, you get an understanding about the architecture for Mobile Ticket and also information about supported browsers and cookies. You should also acquaint yourself with the installation package presented in “Software deliverables” on page 5.
2. In “ Requirements and Preparations” on page 7, you learn about system requirements and what is needed to configure in Orchestra in order to use Mobile Ticket.
3. Next step is to install API Gateway, see “Installation and Basic Configuration for API Gateway” on page 13.
4. Then the installation and configuration of the Mobile Ticket application is described in “Installation and Configuration” on page 15.
5. Once those steps are performed, Mobile Ticket should be up and running with basic functionality, suitable for demo purposes. However, for production environments, you may need to go on with more advanced configuration, such as “Localization” on page 25, and “Change logotype” on page 27.
6. The upgrade procedure is described in “Upgrade” on page 32.
7. If you are experiencing any problems, please see “Troubleshooting” on page 35.
8. Check out “Common Use Cases and Walk-through” on page 39 to see the most common Mobile Ticket user flow.

Architecture

The picture below illustrates the architecture of the Mobile Ticket application, as well as how the different parts communicate with each other:



Limitations

- When mobile user credentials are needed, *Superadmin* user can not be used.
- Currently, creation of multi-service visits *online* is not supported. However, it is possible to for example create a multi-service visit in a kiosk, or with Qmatic Web Booking, and then follow the multi-service visit in Mobile Ticket.
- When using distributed operations and Mobile Ticket, stable connectivity to Central is required.

Browsers and cookies

Supported Browsers

Mobile Ticket is tested and supported on the following browsers:

- Safari
- Chrome
- Opera
- Firefox (PC)

- Edge (PC)

If you use Mobile Ticket on a browser that is not supported, you will be notified about that.

Cookies

In Mobile Ticket, by default, no cookies are stored. Instead local storage is used.

However, there is an option to enable Google Analytics. If so, it might be required to enable cookie consent in Mobile Ticket in order for follow local regulations related to GDPR or similar.

Software deliverables

Please read the *README.md* file for more information.

The following table describes the contents of the Mobile Ticket package, on a high level:

File / folder	Sub folder	Contents Description
sslcert		This folder contains a script called *create_cert.sh*. This folder is also where you should place your certificate, if your server is supporting SSL. This is defined in the *proxy-config.json* file. It also contains the certificate Global.cer.
node_modules	iconv-lite	These sub folders contain parsers, http-proxy, readme-files etc.
	express	
	express-http-proxy	
	raw-body	
	unpipe	
proxy-config.json		This is where you specify the API Gateway Host address, local web-server port, etc. For more information, see “Configuration of proxy/web-server” on page 16.

File / folder	Sub folder	Contents Description
server.js		Main script file to start the solution.
src	app	This folder contains the following sub folders: • *config* - This folder contains the *config.json* file, where you specify the branch radius, where you find the current version number of Mobile Ticket, and so on. For more information, see “General configuration for Mobile Ticket” on page 17. • *locale* - This folder from start only contains the file *en.json* and this is where you place your translated language files, if applicable. For more information, see “Localization” on page 25. • *resources* - In this folder, you find graphical resources used, such as fonts, images, and buttons. A few of these are then pointed out from the *theme-styles.css* file. Here, you can also find the sound file *notification.mp3*, which is pointed out from the *config.json* file.
	zip	This folder contains the files *bundle.min.css.gz* and *bundle.min.js.gz*.
	The files *favicon.ico* and *index.html*	

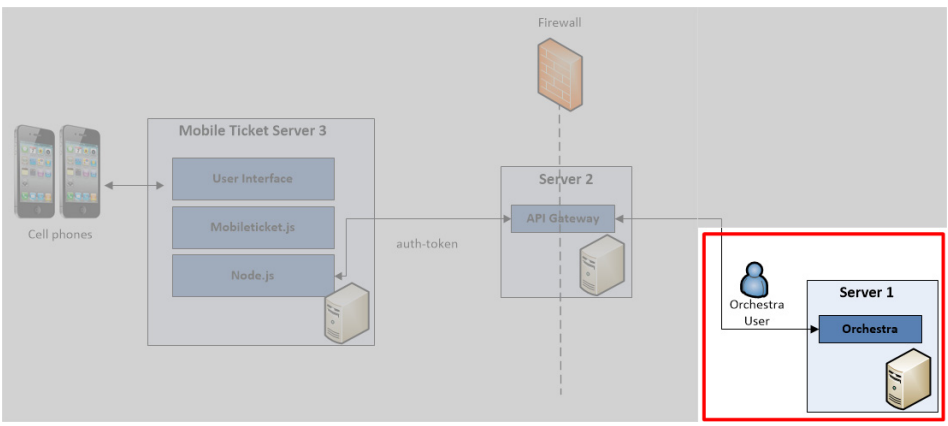
Requirements and Preparations

System requirements 8
Orchestra preparations 8
URLs for notifications and QR codes 10

System requirements

- Orchestra 7.0 update 12, or later, is required for Mobile Ticket. However, some features may require a certain version of a module, see below.
- If you want customers to be able to add their phone number to the visit, you need Notification Admin 4.0.0.49 or later.
- For remote service delivery via Teams, Counter and Qmatic Web Booking need to be installed. For more information, see the configuration guide for Remote Service Teams.
- If service groups are enabled, General Admin application is required.
- HTTPS is needed in order to use location detection (and list branches based on user's position).

Orchestra preparations



To install Mobile Ticket, the following steps need to be completed in Orchestra:

1. In the System Administration application, in the Unit Types tab, add the **VisitApp** unit type. You may also need the Notifications unit type.

   For more information on unit types, see the Orchestra Reference Manual, found on Qmatic World.

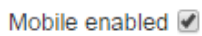
2. Still in the System Administration application, in the Parameters tab, make sure that the **mobile user** has the wanted user credentials, in the Mobile API (Central) section. Make a note of the password that you decide to use here, since it will be used when configuring the API Gateway later.

ⓘ The default password for the mobile user is *ulan*. This, however, should be changed.

3. Some features require that a mobile user with certain access rights is created in User Management. This user needs access to all branches and should have the same credentials as entered in System Administration. This is needed for e.g. appointment check-in, open hours per service and one-time password feature. Make sure this is done if needed.

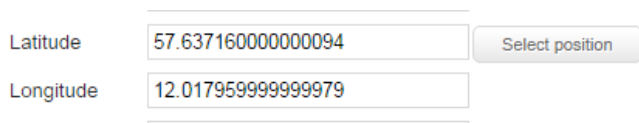
4. In the Business Configuration application, configure the needed **operation profile**, containing the queues, branches and services, that you want to use together with Mobile Ticket. For more information on Business Configuration, see the Orchestra Administrator's Guide, found on Qmatic World.

   The following needs to be configured in Business Configuration:
   - Add the needed unit types to the equipment profile, and configure them to suit your system - both on equipment profile level and branch level. For more information, see the Standard Unit Types Guide, found on Qmatic World.
   - The services and branches that you want to be visible in the Mobile Ticket-application need to have the **Mobile enabled** setting enabled:

     ⓘ Note that it is the external name of the service that will be visible in the *Select service* page, in Mobile Ticket.

   - The branches that are to be used in Mobile Ticket need to have their current position entered in the **Latitude** and **Longitude** fields, as in the following example:

   Also, make sure that you enter the address of your branches, so that they will be displayed correctly in the branch selection list.

5. If you are going to use notifications, please do the following:

   a) Enter the **Mobile Ticket Base URL** in the System Administration application, in the Parameters tab

   b) Configure notifications according to the Orchestra Administrator's Guide. For more information about which URLs to use, please see below section.

URLs for notifications and QR codes

It is possible to use URLs with Mobile Ticket application. URLs can be used to:

- Launch Mobile Ticket where the user can select a service and possibly a branch (if branch is not predefined in link)
- Create a visit for a specific branch and service
- Provide a way for users to check in for appointments.

These URLs can either be sent to customers via sms or email, or be turned into QR codes which can then be placed on for example paper tickets or signs.

URL formats

ⓘ In the URLs, both http and https can be used.

Table 1: URL formats

Use case	URL format
Launch Mobile Ticket start page.	https://<*mobile_ticket_server*>:<port>/ticket
Create visit with predefined branch and service	https://<*mobile_ticket_server*>:<port>/branches/{branchId}/services/{serviceId}
Show service list for predefined branch	https://<mobile_ticket_server>:<port>/branches/{branchId}
Go to appointment check-in	https://<*mobile_ticket_server*>/appointment?appId={{appointmentPublicId}}

QR codes

A typical example of how QR codes could work, would be a customer browsing a department store, looking at mobile devices. Next to the merchandise there is a poster saying “If you need help, scan this QR code!”. The customer scans the QR code and automatically gets a mobile ticket with a visit number for the service Mobile Devices.

A QR code could also be placed on the paper ticket. For that, follow these steps:

1. In the Surface Editor application, create a new ticket layout.
2. Drag and drop the **barcode** component onto the surface.
3. As barcode *type*, select **QR Code**.
4. In the System information drop down list, select **Mobile Ticket URL**.

For more information, see the Orchestra Administrator's Guide, found on Qmatic World.

Notifications

ⓘ For notifications to work with phone numbers added from Mobile Ticket UI, you need notification module 4.0.0.49, or later.

Notifications can be used with Mobile Ticket in the following ways (for example):

- **Visit create**: If the visit is created using for example Concierge and there is contact information available on the visit/customer, you can use the Visit create notification type to send a link to open this specific ticket.
- **Visit update**: If a customer has taken a paper ticket and wants a mobile ticket instead, he or she can walk up to a staff member who can find the visit in the queue and then add a phone number to the visit and send SMS.

  ⓘ For this functionality to work, Orchestra 7.1 or later is required.

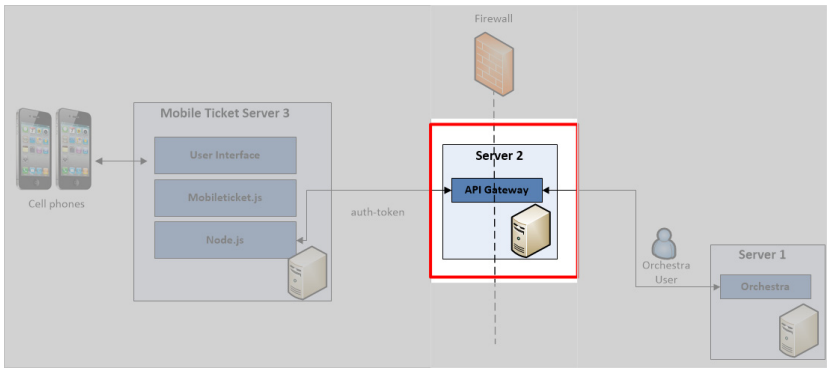
  In both cases above, use Mobile Ticket URL parameter ({{mobileFullURL}}) in the message.

- **Appointment confirmation** or **Appointment reminder**: If a customer has a booked appointment, you can send a link with the appointment confirmation or the appointment reminder so that the customer can check-in. Use the parameter appointmentPublicId. For full check-in URL, see the table above.

Installation and Basic Configuration for API Gateway

Installation API Gateway 14
Basic configuration API Gateway 14

Installation API Gateway



Prerequisite - Windows Control Panel

- On the machine where API Gateway is installed, your firewall must be opened to **port 9090**. For more information, see the API Gateway Manual, found on Qmatic World.

Installation

- Install API Gateway, version 1.3.2.0, or later. Installation file and API Gateway manual can be found on Qmatic World.

Basic configuration API Gateway

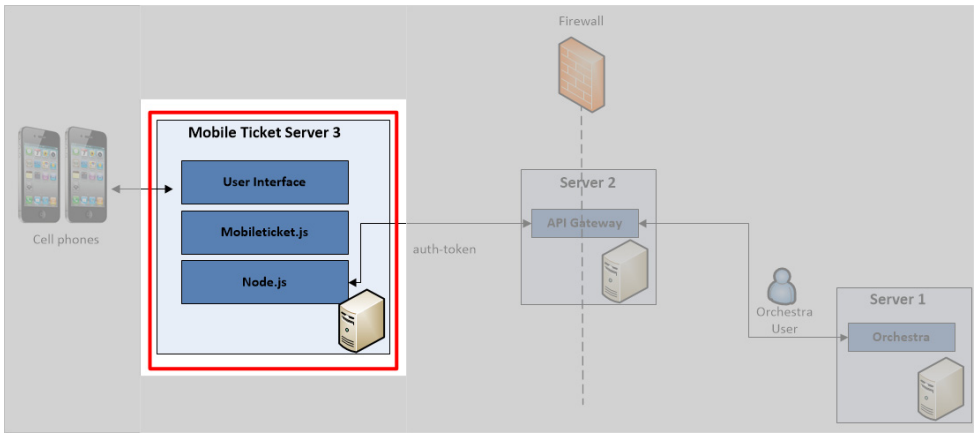
If you are planning to use API Gateway for Mobile Ticket, follow these steps:

1. In the application.yml file, change the **url** to the Orchestra server so that it points to your Orchestra server instead.
2. Generate an **API token**, by running the script **api-token-generator.bat**, located in the bin folder of the QP_APIGateway_win64-<*version_number*>.zip. For more information, see the API Gateway Manual, found on Qmatic World.
3. Enter the generated API token at the mobile user in the application.yml file.
4. Create an encrypted password for the mobile user, by running the script **password-encoder.bat**. For this, use the password that you entered in step 2 of the Orchestra configuration, above. For more information, see the API Gateway Manual, found on Qmatic World.
5. Enter the encoded password, at the mobile user, in the application.yml file.
6. If you are going to use the OTP functionality, please see step 1 in “Enable one-time password functionality” on page 30.

Installation and Configuration

Installation node.js 16
Configuration of proxy/webserver 16
Install Mobile Ticket 17
General configuration for Mobile Ticket 17
Enable visit delay 22
Configure appointment check-in 23
Configure open hours 23
Localization 25
Change logotype 27
Enable HTTPS / SSL 27
Security precautions 30
Install Mobile Ticket as a service 31
Upgrade 32
Database configuration 33

Installation node.js



- Download and install **node.js** from http://www.nodejs.org.

  ⓘ Make sure that you select the LTS version!.

Configuration of proxy/webserver

Open the file **proxy-config.json** in the Mobile Ticket folder. The following parameters need to be configured, to get started with Mobile Ticket:

- **apigw_ip_port** - API Gateway host adress (IP and port).
- **local_webserver_port** - Local webserver port.
- **auth-token** - Mobile API User authentication token.

  ⓘ Note that the **superadmin** user should not be used.

- **embed_iFrame** - To prevent clickjacking, the use of iFrame is blocked by default. If you need to allow iframe, set to ‘true’.

ⓘ For SSL and HTTPS configuration, please see “Enable HTTPS / SSL” on page 27.

Install Mobile Ticket

After completing the setup above, and checking that the configuration described in “General configuration for Mobile Ticket” below is set to your liking, follow these steps to do a fresh install of Mobile Ticket:

1. Go to 'upgrade-helper' folder inside the source folder. (e.g: >cd <*installation_path*>\mobile-ticket-1.9.2\upgrade-helper)
2. Execute npm install to install all dependencies (one-time execution).
3. Execute below cmd to install (replace source and destination paths accordingly):

   Windows
   >gulp install --src <*installtion_path*>\mobile-ticket-1.10.0 --dest <*installation_path*>\mobile-ticket-1.9.2

   Linux
   >gulp install --src <*installation_path*>/mobile-ticket-1.10.0 --dest <*installation_path*>/mobile-ticket-1.9.2

   ⓘ All content of the replaced folder will be replaced with the new build.

4. Then, open a browser and go to the ip address and port that you have configured, for example http://localhost:81 to see if Mobile Ticket is working.

If this fails, please see “Troubleshooting” on page 35.

ⓘ The application needs to have internet access on first start-up in order to fetch dependencies.

ⓘ If you want to install Mobile Ticket as a service, please see “Install Mobile Ticket as a service” on page 31.

General configuration for Mobile Ticket

Most settings for Mobile Ticket can be set in **config.json**, located in MobileTicket\src\app\config**.**

The configuration in this section only describes the settings that only require changing a single setting in config.json. For configuration that involves further steps, see later sections such as “Configure appointment check-in” on page 23, “Configure open hours” on page 23, “Change logotype” on page 27 etc.

ⓘ Changes to config.json are implemented directly, no restart is needed.

Geofencing

geo_fencing - Set to ‘enable’, ‘disable’ or ‘mandatory’. For geofencing to work, HTTPS needs to be enabled. If set to enabled, users can still choose not to share location and a full branch list will be displayed. If set to mandatory, users must share location, otherwise they cannot continue.

branch_radius - Define a value in meters to include branches within on the *Branch Selection* page.

appointment_arrive_radius - Define a value in meters if you want to enforce geofence when appointment check-in is used.

Configure unit of distance

system_of_units - Configure the unit of distance shown in the branch list page. Set to ‘metrics’, ‘imperial’ or ‘auto’. Auto means unit will be based on browser locale.

Redirect to Customer Feedback

If using Customer Feedback, enter the URL in **customer_feedback.** The user will be redirected to this page when a visit is ended.

The URL should look like this example where the *xxxxxx* part should be replaced by the id that you got when ordering Customer Feedback.

https://cf.qmatic.cloud/xxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxx/index.html

Enable Google Analytics and cookie consent

If your are using Google Analytics, enter your Google Analytics Track-ID in **ga_track_id**.

If using Google Analytics you may also need to ask for cookies consent. To enable active cookies consent, set **cookie_consent** to ‘enable’. If enabled, a pop-up will be displayed until the user accepts or declines.

ⓘ You can change the cookies information page that the cookie consent message links to by editing the locale/cookie-consent-files>cookie-LANG.html. Replace LANG with the country code. The browser will pick language file depending on language set in the browser.

Change the notification sound

notification_sound. - Change the sound file used for notifications (for desktop browsers). The default file is *notification.mp3* and it is located in the *MobileTicket\src\app\resources* folder.

Change the service fetch interval

service_fetch_interval - Set the interval at which service information is refreshed. Default value is 15 seconds. This setting is only used if the setting 'Show number of waiting customers' is enabled.

Set the screen timeout for service page

service_screen_timeout - Set the screen timeout duration in minutes. This determines how long the app will stay in the services screen, without creating a ticket. Default value is 10 minutes.

Remove option to create new ticket from app

create_new_ticket – Set to 'disable' to block access to creating visits using the Mobile Ticket UI. This means the Get new ticket button will not be visible and access to branch and service pages (through url) is blocked.

Show or hide number of waiting customer

show_number_of_waiting_customers - Set to 'enable' or 'disable' to show or hide the number of waiting customers in service list.

Allow customers to enter personal information

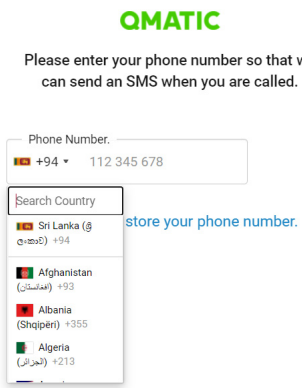
ⓘ Make sure the system fulfils the requirements for this functionality, see "System requirements" on page 8.

If you are going to allow customers to enter personal information please configure the following in config.json:

- **phone_number –** Set to 'enable' to allow entering phone number, 'mandatory' to make it a mandatory field, and 'disable' to hide the field.
- **customer_id –** Set to 'enable' to allow entering customer ID or another customer-specific token. The data entered in customer ID will be saved in custom field 2 (can be enabled in the unit type to show in Counter).
- **privacy_policy** - Enable or disable showing privacy policy when the customer enters personal information

  ⓘ If you enable privacy policy, you need to decide if you want to use an integrated privacy policy or link to an external privacy policy. If you want to link to an external privacy policy, add a link to **privacy_policy_link.** If you want to use the integrated text, leave the link setting empty and edit the text for privacy policy in the translation file (en.json). You can use inline css to style the text.

- **active_consent** - Select whether customer need to actively press button to agree privacy policy.
- **country_code** - Specify the country code that should prepopulate the phone number field. Use either ISO alpha-2 format or International phone number format.
- **preferred_country_list** - Define country codes you want to show up in the top of the drop-down list when selecting a country code. Use ISO alpha-2 format. To add several country codes, add a comma between them.



Enable additional data

additional_data – Enable by setting any string value.The value has to match to the query parameter of the URL for creating a ticket, e.g. http://<baseURL>:81/branches/1/services/1?sammelkaufid=xxxx, where “sammelkaufid” is the query parameter. The value will be saved in custom field 3 (can be enabled in the unit type to show in Counter).

Show queue position / appointment time

Above the ticket, in the queue status bar, you can select if you want to show queue position or appointment time.

show_queue_position – Show or hide queue position

show_appointment_time – Show or hide appointment time for appointment visits.

- If both are enabled and it’s a normal visit -> Queue position is shown
- If both are enabled and it’s an appointment -> Appointment time is shown

- If only queue position is enabled and it's a normal visit -> Queue position is shown
- If only queue position is enabled and it's an appointment -> Queue position is shown
- If only appointment time is enabled and it's an appointment -> Appointment time is shown
- If only appointment time is enabled and it's a normal visit -> Nothing is shown
- If none are enabled -> Nothing is shown

Enable service groups

It is possible to arrange services in service groups on the page where the customer selects a service.

ⓘ For this feature to work, service groups must be configured in the extension General Admin. For more information, see the General Admin manual.

Enable service groups

To enable service groups, set 'value' below to 'enable'.

```
"service_group": {
    "value": {
      "availability": {
        "value": "enable",
```

Enable single service grouping selection

This feature is used to control how many service groups the user can expand at the same time. If enabled, the user can only expand one service group. If disabled, the user can expand all of them.

```
 "single_selection": {
   "value": "enable",
```

Customize the footer

By default, Mobile Ticket shows a footer with the text "Powered by Qmatic" where "Qmatic" is a png file with the Qmatic logo. This footer can be customized with the following settings:

`logo -` Show ('enable') or hide ('disable') the brand logo. A custom logo can be added to app\resources and configured in app\theme\theme-styles.css.

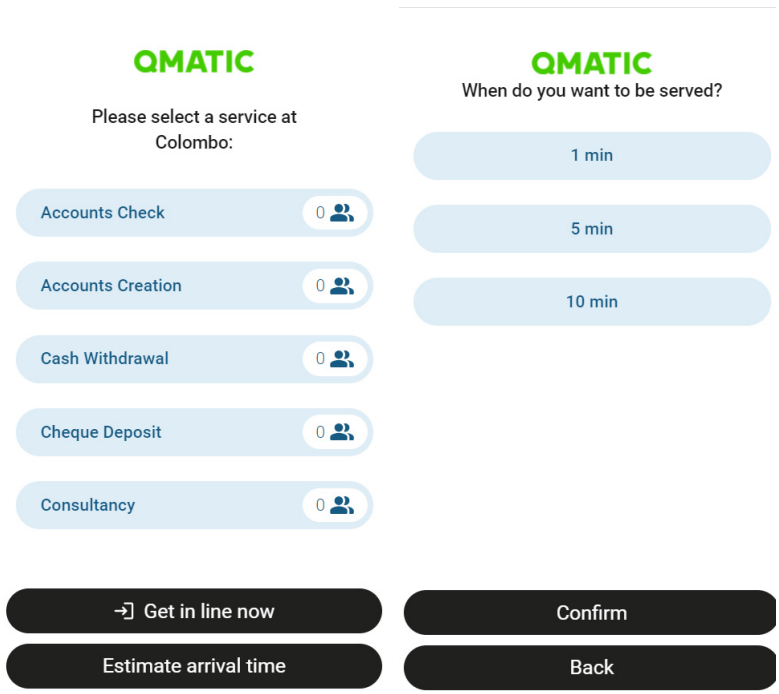
`custom_text` - If you don't want to use an image as logo, you can just enter the company name here as plain text. The text is inserted right after "Powered by" (which can be edited from the language file).

If you add both a custom text and enable the logo, the logo setting overrides the custom text.

Enable visit delay

In Mobile Ticket, there is a possibility to let customers get in line remotely, estimate an arrival time, and then get called on their estimated arrival time.

How it works is that the customer can choose to estimate an arrival time in Mobile Ticket. Their waiting time will start counting, but they will never be called before the selected time has passed. The customer will see a countdown timer with how much time there is left and can choose to extend the estimated arrival time even further.



To turn on this feature, set the following parameter to 'enable':

```
"delay_visit": {
    "value": {
      "availability": {
        "value": "enable",
```

Also, define which should be the available time slots. Enter comma-separated values in minutes.

```
"time_slot": {
  "value": [],
```

Configure appointment check-in

1. A user with the same credentials as the mobile user in System Administration needs to be created, if not done already. The user needs to have access to all branches and be assigned to a role with the Appointment and Connector Entrypoint access modules.
2. Configure the following settings in config.json:
   - **`appointment_late`** - the time in minutes a customer is allowed to arrive late.
   - **`appointment_early`** - the time in minutes a customer is allowed to arrive early
   - **`timeFormat`** - time format for showing the appointment time in the check-in page.
   - **`dateFormat`** - date format for showing the appointment date.

Configure open hours

It is possible to configure during which open hours the possibility to take a mobile ticket should be available.

Opening hours can be configured in two ways. Note that you cannot combine these options.

- The first option is to control the opening hours for different services by using the Button Scheduler application and unit type.
- The second option is to control the opening hours for all branches by setting open and close times in config.json.

Configure open hours per service

To configure the open hours per service, you need to install and configure the Button Scheduler application and a unit type for Mobile Ticket.

1. Add buttonschedule.war into Orchestra custdeploy folder and configure schedules and other settings in the application. Please contact Professional Services if you need further assistance.
2. In Orchestra, upload the unit type buttonschedule.utt (version 7.0.3.09 or later). in System Administration and add it to the equipment profile.

3. In Orchestra, add the Connector ServicePoint access module to a new or an existing role that is assigned to the mobile user. This user needs to be created if not done already, and needs to have the same credentials as the mobile user in System Parameters and access to all branches
4. In **config.json**, set the parameter **`branch_schedule`** to 'enable'.

ⓘ When this is enabled, all values from **`branch_open_hours`** are ignored.

Configure open hours for all branches

The open hours for all branches in your system that uses Mobile Ticket are configured in the **config.json** file, in the **`branch_open_hours`** parameter.

ⓘ In the config.json file, there is a distinction between the actual opening hours, for example `open_hours.week_day1`, and the time when the this should be displayed in Mobile Ticket, `display_from` and `display_to`.

Example:

The actual store might not open their doors until 08.00 but they can open up the possibility to take tickets in Mobile Ticket 10 min before the store opens. In the same way, they might want to close the possibility to take tickets 10 min before the store closes.

This would mean that the opening hours displayed to the user would be for example 08.00-18.00, whereas if someone accesses Mobile Ticket at 07:50 it would still be possible to take a ticket, if the `display_from` parameter has been configured to 07.50.

If the time values for a day are left empty, "Closed" will be displayed for that day. This text can be changed in the language property file.

ⓘ If you want Mobile Ticket to appear as closed at, for example, 14:00, you need to set the close time to 13:59.

ⓘ The open hours functionality assumes that all branches as well as all devices used to access Mobile Ticket have the same time zone. If you need a more advanced opening hour functionality, please contact Professional Services.

ⓘ If a user has the branch or service page open, and the branch close time is reached, the user will still be able to create visits.

Localization

The language used depends on the language used in your browser and this also has effect on if the distance to the branches is shown in metrics or imperial (Only if unit of distance setting in config.json is set to 'auto', see "Configure unit of distance" on page 18)

It is possible to support several languages, in parallel, in the same installation. Also, both left-to-right and right-to-left are supported.

Both Left-to-right (LTR) and Right-to-left (RTL) mode are supported by Mobile Ticket. The user interface is mirrored when using Right-to-left mode.

ⓘ The en.json file must never be removed from the system.

Translate Mobile Ticket UI

Translating the static texts in Mobile Ticket is done by translating the texts in MobileTicket\src\app\locale\en.json file.

1. Copy the en.json file to the same folder.
2. Rename the copied file to your language code, for example fr.json for French.
3. Translate all the phrases in the file.

ⓘ Make sure that your translated file contains correct JSON format. For example, there may not be a comma (,) at the end of the last translated phrase in each block of the *.json file, since this will cause the translation to not work properly.

ⓘ If a phrase for some reason is missing in your translated json file, the fall-back language is English.

ⓘ If you are using the Chrome browser, you need to reboot your computer after changing the locale, for the settings to be applied properly.

Translate service names

ⓘ To be able to have translated service names in Mobile Ticket, the service names needs to be translated in General Admin application.

1. In config.json, set service_translation to enabled.
2. In the file application.yml for the API Gateway installation, add the following lines under routes:

```
translations:
```

```
path: /MobileTicket/translations/*
url: ${orchestra.central.url}/calendar-backend/api/v1
```

Change logotype

To change logotype, name the image you want to use qmLogo.png and place it in the resources folder.

Enable HTTPS / SSL

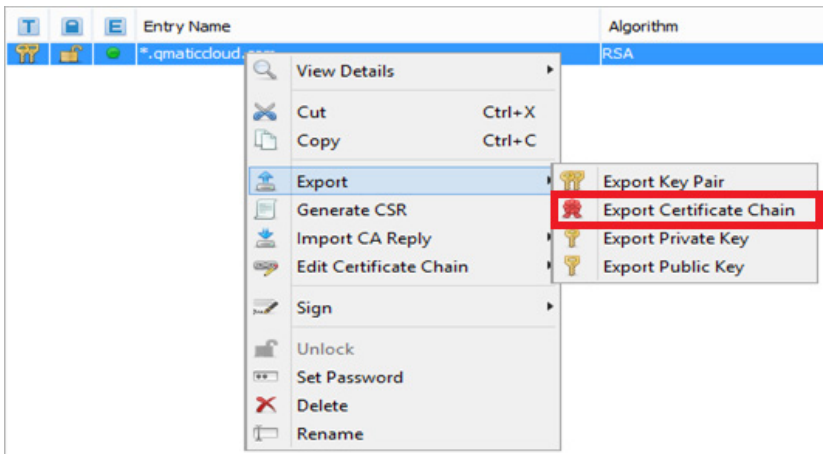
Exporting a certificate file and private key file from a keystore

ⓘ Since Mobile Ticket uses X509 certificate format, any certificate that is created should be according to the X509 format.

ⓘ We recommend using a tool such as Keystore Explorer, http://keystore-explorer.org/, for managing your keystores and certificates.

The procedure below describes how to export an X509 certificate from a p12 keystore file. This certificate can then be used with Mobile Ticket.

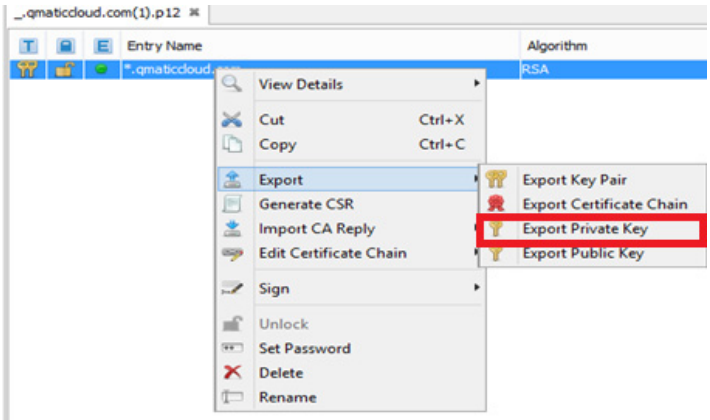
1. Open the keystore file in Keystore Explorer, right-click and select *Export Certificate Chain*, as in the example below:



2. Select *X.509* in the popup window and keep *PEM* checked. Change the file name to *server.crt* and click *Export*.
3. Copy the exported *server.crt* file to the *sslcert* folder in your Mobile Ticket installation.

The procedure below describes how to export a private key from a p12 keystore file.This private key can then be used with Mobile Ticket.

1. Open the keystore file in Keystore Explorer, right-click and select *Export Private Key*, as in the example below:



2. In the popup, select *OpenSSL* and click *OK*.
3. In the following popup window, deselect the *Encrypt* option and keep *PEM* checked. Change the file name to *server.key* and click *Export*.
4. Finally, copy the *server.key* file to the *sslcert* folder in your Mobile Ticket installation.

HTTPS / SSL Configuration - API Gateway

If you want to run Mobile Ticket against API Gateway in HTTPS mode, change the *server* section in the *application.yml* file, so that the *ssl*, *key-store*, *key-store-password*, and *key-password* parameters are uncommented, as in the following example:

```
server:
  # define server port for gateway
  port: 9090
  ssl:
    key-store: classpath:keystore.jks
    key-store-password: secret
    key-password: password
```

Make sure that you are using a keystore with a correct password here!

Also, make sure that you have configured the *proxy-config.json* file correctly, see below section.

Security and HTTPS configuration node.js / Mobile Ticket

Edit the **proxy-config.json** file in the MobileTicket folder. The following parameters need to be configured, if you want to use SSL / HTTPS.

- **`local_webserver_ssl_port`** - Local webserver HTTPS port.
- **`support_ssl`** - Whether or not server should support SSL. For this, you need a valid public key and a certificate in the *sslcert* folder. For more information, see “Exporting a certificate file and private key file from a keystore” on page 27.
- **`gateway_has_certificate`** - Whether or not API Gateway is configured to run on HTTPS.
- **`gateway_certificate_is_valid`** - Whether or not the configured SSL certificate in the API Gateway is a valid one.

  ⓘ This setting is only active if the `gateway_has_certificate` parameter is set to ‘true’. This must be set to ‘false’ if the API Gateway is configured with a self signed certificate.

- **`tls_version`** - You only need to configure this if you are using another TLS version than the latest. Otherwise leave empty.
- **`cipher_set`** - Indicates cipher set to use in the server. Enter the value like this: ['ECDHE-RSA-AES128-SHA256', 'DHE-RSA-AES128-SHA256', 'AES128-GCM-SHA256']. Keep it empty if you don’t want to change the cipher set.
- **`hsts_expire_time`** - Here you can set the HSTS (HTTP Strict-Transport-Security) expire time.

Security precautions

Prevent Mobile Ticket from being opened in another device

As a security precaution, you may want to limit the possibility to open Mobile Ticket in another device than the one it was created in. To turn this on, set **`block_other_browsers`** in config.json to 'enable'.

Enable one-time password functionality

For security reasons, you may want to send a one-time password (OTP) to all users who want to create a mobile ticket to confirm that they are a unique user.

The flow is like this: The user is asked to enter phone number > A PIN number is sent to the user > The user enters PIN.

To enable this feature

1. Configure API Gateway by adding sms_api mapping to application.yml

```
sms_api:
path: /rest/notification/**
url: ${orchestra.central.url}/notification/
```

2. In Orchestra, a mobile user with same credentials as in System Parameters and access to all branches needs to be created, if not done already. Add the NotificationConnectorUser role to the mobile user.
3. This feature uses a database, so you need to set up the database configuration, if not done already. See "Database configuration" on page 33
4. Go to src/app/config/config.json and set **`otp_service`** to 'enable'.
5. Restart the server by running `npm start.`

Prevent replay attacks

If you want to prevent the possibility to create multiple tickets using replay requests, follow these steps:

1. This feature uses a database, so you need to set up the database configuration, if not done already. See "Database configuration" on page 33
2. Set `create_ticket_token` in src/app/config/config.json to 'enable'.
3. Restart the server by running `npm start.`

Install Mobile Ticket as a service

In production environment you may want to install Mobile Ticket as a service.

Windows

For Windows, follow these steps:

1. After running `npm start` and checking that Mobile Ticket works, exit the command prompt.
2. Go to bin folder (cd <mobileticket_installation_directory>/bin) and run **install-mt-service.bat** file (you may need to run it as administrator). This will create a Windows service with service name “Qmatic Mobile Ticket”.
3. To uninstall, run **uninstall-mt-service.bat**. It should then be removed from Windows service list.

Linux - CentOS 7

For Linux, follow these steps:

1. After running `npm start` and checking that Mobile Ticket works, exit the command prompt.
2. Install the pm2 package by executing

   ```
   sudo npm install -g pm2
   ```

   (https://pm2.keymetrics.io/docs/usage/quick-start/)

3. Execute

   ```
   pm2 start /<mobileticket_installation_directory>/server.js
    --name="mobile-ticket"
   ```

4. Execute

   ```
   pm2 startup
   ```

5. To install, execute

   ```
   npm install -g pm2 && pm2 update
   ```

   It should now be installed as a service in CentOS. You can execute `pm2 list` to verify.
6. To uninstall the service, execute “pm2 delete “mobile-ticket”.

Upgrade

This section explains how upgrade to a newer version of Mobile Ticket can be done without resetting the current configuration data.

To upgrade Mobile Ticket using the upgrade helper, follow these steps:

1. Go to the “upgrade-helper” folder inside the source folder. (e.g: >cd <*installation_path*>\mobile-ticket-1.9.2\upgrade-helper)
2. Execute npm install to install all dependencies (one time execution).
3. Execute below cmd to upgrade (replace source and destination paths accordingly):

   Windows:- >gulp upgrade --src <*installation_path*>\mobile-ticket-1.10.0 --dest <*installation_path*>\mobile-ticket-1.9.2

   Linux:- >gulp upgrade --src <*installation_path*>/mobile-ticket-1.10.0 --dest <*installation_path*>/mobile-ticket-1.9.2

All files will be replaced except below list of files/folders

- ./proxy-config.json
- ./mt-service/src/config/config.json
- ./src/app/config/config.json
- /src/app/locale/en.json
- /src/app/resources

Above files and folders will be compared and if there are any new items it will be added into the respective destination.

ⓘ Note that **theme-styles.css** file will not be automatically upgraded. If there are any new changes, they will be documented in release notes.

You can also “upgrade” by installing a newer version of the software. In this case, any configuration and customization performed must most likely be performed manually again.

Database configuration

Some features in Mobile Ticket requires a database and configuration related to the database. The database should be a Mongo Database and it can be any of following:

- On-premise database connection (Local Database)
- Database hosted in a cloud (e.g. AWS)
- Global cloud database service for MongoDB (e.g. Atlas)
- Docker container for MongoDB
  Note: If you want to run the Mongo instance on a docker container, please see “Set up MongoDB on docker” on page 33.

The following settings need to be configured:

- Specify a valid database connection URL to **`db_connection_string`** parameter of mt-service configurations. Configuration file can be found in <*mobileticket-version*>/mt-service/src/config/config.json”.

  Example connection string:
  mongodb://YourUsername:YourPassword@host:port/YourDatabaseName[?options]

- Still in the same file, if you are a Mobile Ticket installer / DevOps engineer who want to configure several Mobile Ticket instances to use the same database (shared database) with a discriminator for each mobile ticket instance, a unique Tenant ID needs to be specified for **`tenant_id`**. (e.g. “b14f3c08”, “client-0001”).

Set up MongoDB on docker

This section describes how to run Mongo instance on a docker container.

1. Pull an image which gives you a mongo DB host. You can use an image called “**mongo**”
   $ docker pull mongo:latest
2. Copy the docker compose file and the DB script to a directory (required files can be found below).
3. Run the docker compose file with **$ docker-compose up -d**

Now you should be able to see the service running. You can check it from docker desktop or **$ docker ps**.

Connection strings

- connect with root user: mongodb://root:adminhe11@127.0.0.1:27017/mt?authSource=admin

- connect with db user: mongodb://mobileticket:nsavip@127.0.0.1:27017/mt?authSource=mt

docker-compose.yml ¨

```yaml
version: '3.8'
services:
    # ... other configs ...
    mongodb:
        image: 'mongo'
        container_name: 'mt-mongo-container'
        environment:
            - MONGO_INITDB_DATABASE=mt # database name
            - MONGO_INITDB_ROOT_USERNAME=root # set your container root username
            - MONGO_INITDB_ROOT_PASSWORD=adminhe11 # set your container root password
        volumes:
            - ./init-mongo.js:/docker-entrypoint-initdb.d/init-mongo.js:ro
            - ./mongo-volume:/data/db
        ports:
            - '27017:27017'
    # ... other configs ...
```

init-mongo.js

```
db.createUser(
    {
        user    :   "mobileticket",
        pwd     :   "nsavip",
        roles   :   [
            {
                role    :   "readWrite",
                db      :   "mt"
            }
        ]
    }
)
```

Troubleshooting

Troubleshooting guide 36
Frequently Asked Questions 37

Troubleshooting guide

If, after following the configuration procedures, your Mobile Ticket setup still does not work as expected, we suggest that you check your configuration by following the steps below. Use a tool such as Advanced REST client or Postman to perform the REST calls.

ⓘ For more information about REST calls that can be used for the Mobile Ticket-scenario, through the API Gateway, please see the API Gateway SDK documentation.

1. Perform the REST call below, to verify that you have a successful connection to Orchestra central (using port 8080)

   ⓘ You need to replace “serviceId” and “branchId” with valid Id numbers from your configuration. Also, you need to use *Basic Authorization* and the *Username* and *Password* for the *mobile* user (set in *Parameters* in the *System Administration* application, in Orchestra).

   /qsystem/mobile/rest/v2/services/serviceId/branches/branchId/ticket/issue/

   If the REST call successfully issues a ticket, you have verified that the connection works.
   If it fails, however, check that Orchestra is configured correctly. For more information, see “Installation and Configuration” on page 15.

2. To verify that your connection through the API Gateway is correctly configured, perform the following trial REST call through API Gateway.

   ⓘ Make sure that you have set the correct authorization, using your *auth-token* and that you replace *[serviceId]* and *[branchId]* with valid Id numbers from your configuration.

   /MobileTicket/services/[serviceId]/branches/[branchId]/ticket/issue

   If the REST call is successful and a ticket is issued, you have verified that the connection works.
   If it fails, however, make sure that you have configured the API Gateway correctly. See “Basic configuration API Gateway” on page 14.

3. Start the *node.js* server, from a command prompt, with the following command:

   npm start

4. Check that the web page and UI works as expected, by entering *https://<your ip-address>:4443* in the browser address field.

Frequently Asked Questions

The table below contains some answers to some frequently asked questions:

Question	Answer
Why are my branches not available, in the branch selection list?	In Orchestra, make sure that the check box *Mobile enabled* is checked for all the Branches that you want to be listed.
Why are my services not available, in the service selection list?	In Orchestra, make sure that the check box *Mobile enabled* is checked for all the Services that you want to be listed.
Why can I not create a mobile ticket?	In Orchestra, make sure that the *VisitApp* unit type is installed. Make sure that API Gateway is configured correctly.
Why is my connection to Orchestra/ API Gateway not working properly?	Make sure that you have opened up port 9090, in your firewall, see “Prerequisite - Windows Control Panel” on page 14. Make sure that API Gateway is configured correctly.

Common Use Cases and Walk-through

Use cases 40
End user walk-through 41

Use cases

The following are examples of typical use cases:

- The end-customer goes to the branch and takes a paper ticket with a QR code on it. Information on the ticket and/or in the branch says something like “If you want to use your mobile phone to stand in line, please scan the QR code!”. The end-customer scans the QR code and can then follow the progress of the line on his/her phone.
- The end-customer goes to the branch and sees information, something like “Go to this url/Scan this QR code to take a ticket online!” The end-customer then enters the url/scans the QR code, takes a ticket and can then follow the progress of the line on the phone.
- The end customer goes to the branch, and either enters his/her phone number in a kiosk when selecting service, or gives the phone number to a receptionist/floorwalker. When the visit is created, this triggers a notification with a link to that particular visit in Mobile Ticket, so that the customer can follow the progress online.
- The end customer books an appointment from home and enters contact information. An hour before the appointment he/she gets an appointment reminder with a link to check in for the appointment and follow the progress of the ticket online.

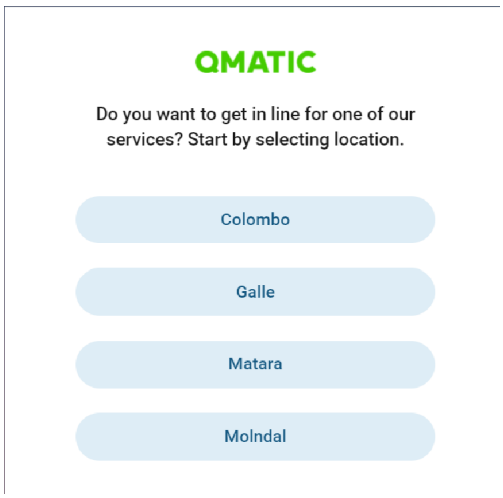
ⓘ For information about how to configure the use of QR codes and SMS notifications, please refer to “URLs for notifications and QR codes” on page 10 and to the chapter about the *Notification Admin* application, in the Orchestra Administrator’s Guide, found on Qmatic World.

Below is a walk-through of the typical flow for the end-user through Mobile Ticket. For each step, there is and example of the GUI and some information that may be useful when demonstrating Mobile Ticket.

End user walk-through

In the following sections, a common flow for Mobile Ticket is described. It starts when the user has opened Mobile Ticket (via URL or QR code) with no services or branches preselected.

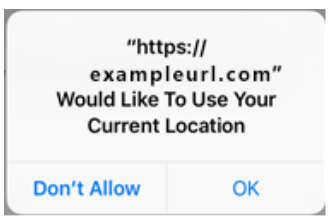
Selecting a branch



First, a branch is selected. Only branches within the defined range are listed. Branches with a defined position are sorted on distance. Branches without a defined position are sorted alphabetically.

If there are no branches within range, you are asked to try again. If there is only one available branch, within range, it will be selected and you will move on to service selection, automatically.

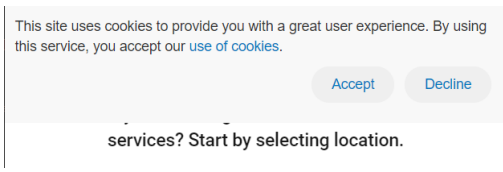
The position is defined by the browser. If using a smart phone, you will get a question that looks something like this:



ⓘ For location detection to be supported, you need to run in HTTPS.

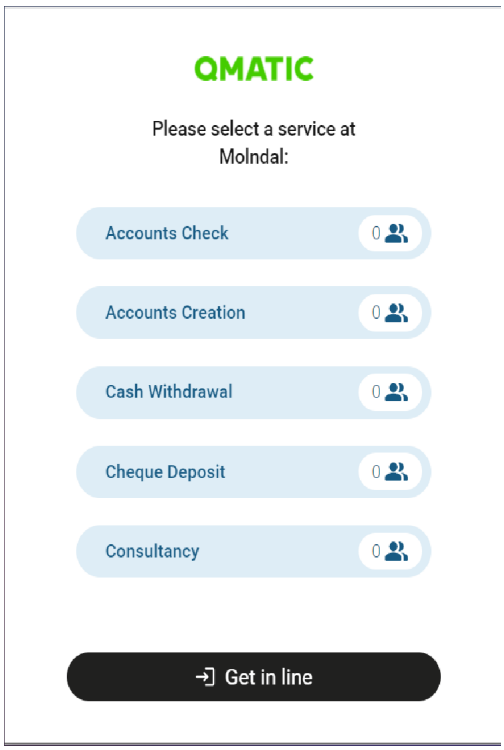
If positioning services are turned off and your position therefore is unknown, all available branches will be listed, in alphabetical order.

Cookies



If active cookie consent is activated, this popup will be visible until the user accepts or declines.

Selecting a service



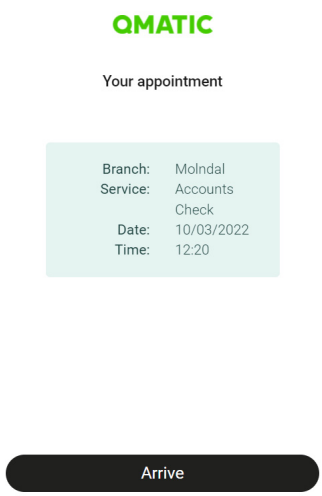
Next, the user selects the wanted service and clicks Get in line. Services are sorted alphabetically and the number of people waiting for each service is displayed.

If only one service is available, it is automatically selected, but you still need to click Get in line to generate a ticket.

After pressing Get in line, user may be prompted to enter their phone number in order to receive a PIN number, if the OTP functionality is enabled.

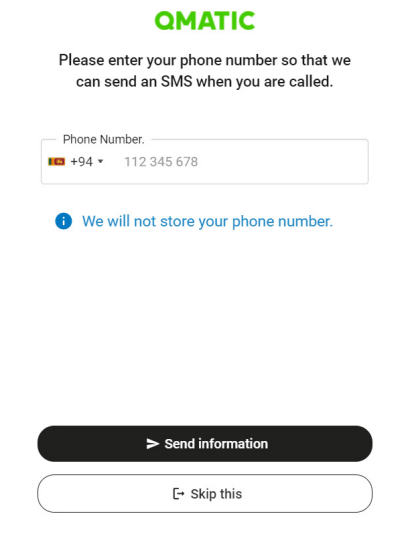
Alternative entry - appointment check in

An alternative flow to open a mobile ticket is if the user uses a link to check in for an appointment. When the user has checked in, the normal visit flow will be followed.

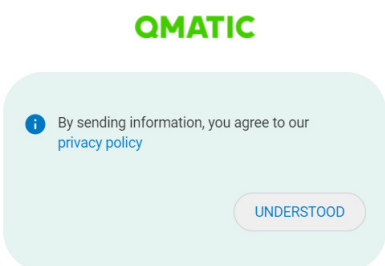


Add phone number

If the functionality to add phone number is enabled, that is displayed before the ticket is shown.

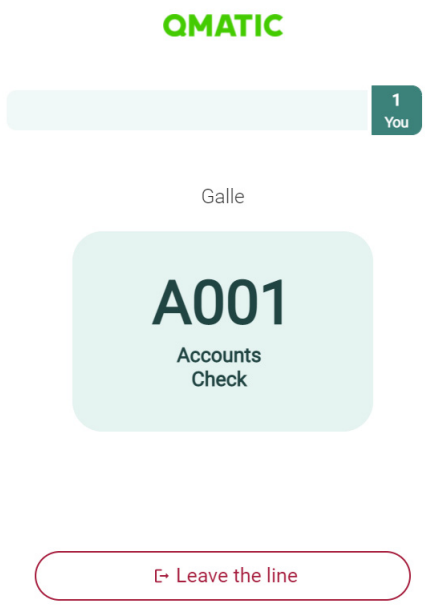


Before the phone number is added, the user has to agree to the privacy policy (if that is enabled)



Waiting to be called

When the ticket has been generated, the ticket number is displayed, together with the branch and service name, and information about queue position.



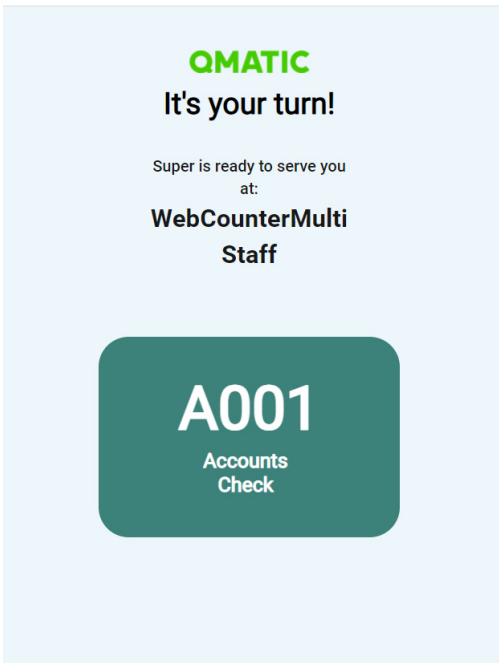
While standing in line, the you can follow the queue progress and place in line as you get closer to being called. There is also an option to Leave the line, which will direct the user to the Thank you page. The same will happen when pressing the back button in the browser.

If the visit is transferred, the information about the new service name is displayed. If the visit is recycled, a message saying “You will be called shortly” is displayed.

Popups and alert/confirm dialogs use the default browser behavior. This means that, for example, if the user clicks *Yes* when asked if he/she wants to leave the line, this value will be cached and the popup will not be displayed again, until the browser cache is cleared. It is also possible to create customized popups, if you want to avoid the browser behavior.

Called to a counter

Once the user is called, the ticket number will be flashing, a sound will be played (only for desktop browsers) and the user will get information about which service point and staff member to go to.



Note that the flashing ticket number and sound only will be played the first time the ticket is called. If Recall is used, no sound etc is played.

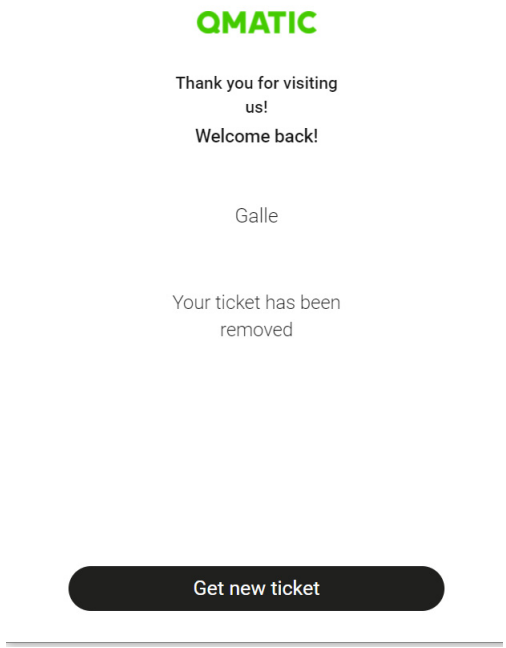
It is also possible to configure the system so that the user is notified by sms when he or she is called.

If Teams Meeting integration is used, a button to open the virtual meeting is visible in the ticket page.

When the visit is over

When the you have been served, and the visit is ended, one of the following things will happen - depending on the configuration.

- A Thank You page will be displayed. From this page, it is also possible to take a new ticket, by clicking *Get new ticket*, if that setting is enabled. This is however not possible if the ticket was opened via a URL.



- You are redirected to a Customer Feedback survey page.

Landscape Mode

For demo purposes, we recommend that you use Mobile Ticket in portrait mode. However, it is of course also possible to use Mobile Ticket in landscape mode.

Q-MATIC AB,
Box 198,
SE-431 23 Mölndal,
Sweden.

VAT No SE556212749701

Phone: +46 31 756 46 00
Email: info@qmatic.com
Web: www.qmatic.com