Contacts represent website visitors and store marketing-related information about them. To learn more about contacts, see Working with contacts.

To track contacts on websites presented by a separate MVC application, you need to:

1. Install the **Kentico.ContactManagement** NuGet integration package into your MVC project.
2. Enable on-line marketing in the **Settings** application of the related Kentico instance.
3. Assign existing contacts to registered or signed-in users to make contact tracking effective.

> **Default cookie level or consent requirements**
>
> The contact tracking functionality in Kentico EMS only works if the **Default cookie level** setting is set to *Visitor* or *All*, or for visitors who give tracking consent and increase their cookie level. For more information, see Working with consents on MVC sites.

Contact detail updates are batch processed and logged at an interval together with activities. To learn how to change the logging interval, see Configuring activity and contact updates processing.

Obtaining the current contact

The system uses a browser cookie as persistent storage for the current contact. By default, the storage is empty. However, once you ask for the current contact, the system sets the contact's GUID (unique identifier) to every response cookie.

Whenever you need to get the current contact during the implementation of an MVC website, use the asynchronous **GetCurrentContactAsync** method that returns a *ContactInfo* instance. The **GetCurrentContactAsync** method is available in the **ContactTrackingService** class, which is provided by the **Kentico.ContactManagement** integration package. Specify the user name (*string userName*) of the currently signed-in user as the parameter (if there is a signed-in user). If the user name does not exist, the system creates a new contact for the given user name.

```
using System.Web.Mvc;

using CMS.ContactManagement;
using Kentico.ContactManagement;
```

Example

```
            // Gets the current contact
            IContactTrackingService contactService = new ContactTrackingService();
            ContactInfo currentContact = contactService.GetCurrentContactAsync(User.
Identity.Name).Result;
```

Assigning the related contact to registered users

For effective tracking of contacts, you need each registered user to use the same contact for all their visits. Call the **MergeUserContactsAsync** method from the **ContactTrackingService** class to:

- Assign a contact to a user immediately after registration
- Switch a visitor's contact to their user's contact after the visitor signs in

To ensure that every registered user gets their own permanent contact and every signed-in user is assigned their permanent contact:

1. Open your MVC project in Visual Studio.
2. Edit the controllers that process sign-ins and registrations (referred to as *MembershipController* in the example code below).

3. Initialize a **ContactTrackingService** instance in the sign-in and registration controller's constructor. This enables you to work with contacts.

   We recommend using a dependency injection container to initialize service instances. When configuring the lifetime scope for *ContactTrackingService*, create a separate instance for each request.

   ```
   using Kentico.ContactManagement;
   ```

   ```
   public class MembershipController : Controller
   {
       private readonly IContactTrackingService contactTrackingService;

       public MembershipController()
       {
           contactTrackingService = new ContactTrackingService();

           // ...
       }

       // ...
   ```

4. Call the **MergeUserContactsAsync** method of the *ContactTrackingService* instance in the controller's sign-in and registration actions.

   - Call the method after a successful sign-in or registration.
   - The *userName* variable in the code example represents the user name of the visitor that signs in or registers.

     ```
     await contactTrackingService.MergeUserContactsAsync(userName);
     ```

For every registered user, the system permanently assigns their current contact. For every signed-in user, the system merges the current contact with the given user's contact (if the current contact does not belong to a different user).

Merging contacts

In some cases, multiple contacts on your websites may actually represent a single realworld person. Automatic merging allows you to get rid of duplicates by combining several contacts into a single object. The system automatically merges contacts that are associated with the same email address.