Webhook Configuration

MDCMS Webhooks can be configured for each integrated Git Repository on the Git Origin server whenever events of the following type occur:

Push - commits are pushed to a branch in the repository - this can trigger Continuous Integration within MDCMS and/or dynamically update external references for system objects.

Pull-Request Created - a pull request is created - this keeps MDCMS aware of Pull Requests that are created.

Pull-Request Updated - a pull request is updated - this keeps MDCMS aware of Pull Requests that are modified, merged or closed.

The http(s) target of the webhook is the same MDCMS http API server that is used for MDOpen. If the origin server resides outside of your network, you will need to allow inbound http traffic from the origin server to the MDCMS API Server on your development partition. The URL format for the webhook is:

```
https://<your-mdcms-host>/<mdcms-instance>/<server-type>/webhook
```

Examples:

- Azure DevOps: `https://devsys.mycompany.com/mdcms/azure/git/webhook`
- Bitbucket Cloud: `https://devsys.mycompany.com/mdcms/bitbucket-cloud/webhook`
- Bitbucket Data Center: `https://devsys.mycompany.com/mdcms/bitbucket/webhook`
- GitHub: `https://devsys.mycompany.com/mdcms/github/webhook`
- GitLab: `https://devsys.mycompany.com/mdcms/gitlab/webhook`

Network Recommendations: - proxy-forward the inbound traffic for the URL resource to the internal API Server URL if the API Server is not directly accessible from the origin server - allowlist the origin server IP addresses on the firewall of the API Server host, if possible, to restrict access to only the origin server(s)

Register Webhooks from MDOpen

If the repository has an API Server ID configured, webhook registration can be done directly from MDOpen for Azure DevOps, Bitbucket, GitHub, and GitLab:

1. In MDOpen, open the **Git Repositories** view.
2. Select option **Webhooks** for the relevant Repository definition.
3. A row will be displayed for each supported Event Type based on the Git Server Type of the repository. If the row already displays a URL, date, time and user, then MDOpen has previously registered the webhook for that event type. If the URL is blank, then the webhook has not been registered for that event type.
4. Click on an Event Type to create or delete/recreate the webhook for that event type. If the registration was successful, the URL date, time and user will appear for the row. If not successful, click the **View Logs** button to view the REST API log for the API Server to troubleshoot the error. The most common errors are related to network connectivity or permissions of the user account on the Git server.

If routing through a proxy, the hostname and/or port of the webhook URL will likely need to be edited on the Git server after registration from MDOpen, to ensure the Git server can reach the webhook URL.

Manual Webhook Registration

Use the steps below if an API Server ID is not defined for the repository or if the user's credentials isn't scoped to include webhook management.

Azure DevOps

1. Log into Azure DevOps with a user who has administrative rights for the project.
2. Go to **Project Settings** for the project.
3. Click **Service hooks**.
4. Click **+** to add a subscription.
5. Select **Web Hooks** from the list and click **Next**.
6. Select the **Code pushed** trigger event.
7. Select the specific or **(Any)** repository, branch, and group member, then click **Next**.
8. Enter the MDCMS webhook URL (using the `/azure/git/webhook` path).
9. Leave optional settings at their default values.
10. Click **Finish**.
11. Repeat steps 4-10 for the **Pull request created** and **Pull request updated** trigger events.

Bitbucket

1. Log into Bitbucket with a user who has administrative rights.
2. Go to **Settings** for the repository.

3. Expand Workflow in the Repository Settings menu and click **Webhooks**.
4. Click **Add webhook**.
5. Enter a descriptive title (e.g., `MDCMS`).
6. Enter the MDCMS webhook URL (using the `/bitbucket/webhook` path).
7. Enable **Request history collection** to assist with troubleshooting during initial setup.
8. Ensure **Repository push**, **Pull request Created**, **Pull request Merged** and **Pull request Declined** is selected under Triggers.
9. Click **Save**.

GitHub

1. Log into GitHub with a user who has administrative rights.
2. Click on the **Settings** tab for the repository.
3. Click **Webhooks** under Code and automation in the Settings menu.
4. Click **Add webhook**.
5. Enter the MDCMS webhook URL (using the `/github/webhook` path) as the Payload URL.
6. Select the radio button **Let me select individual events**.
7. Select **Pushes** and **Pull requests**.
8. Click **Add webhook**.

GitLab

1. Log into GitLab with a user who has administrative rights.
2. For the given project, hover over **Settings** in the sidebar and click **Webhooks**.
3. Enter the MDCMS webhook URL (using the `/gitlab/webhook` path).
4. Ensure **Push events** and **Merge request events** is selected under Triggers.
5. Click **Add webhook**.

