Current Release Procedures

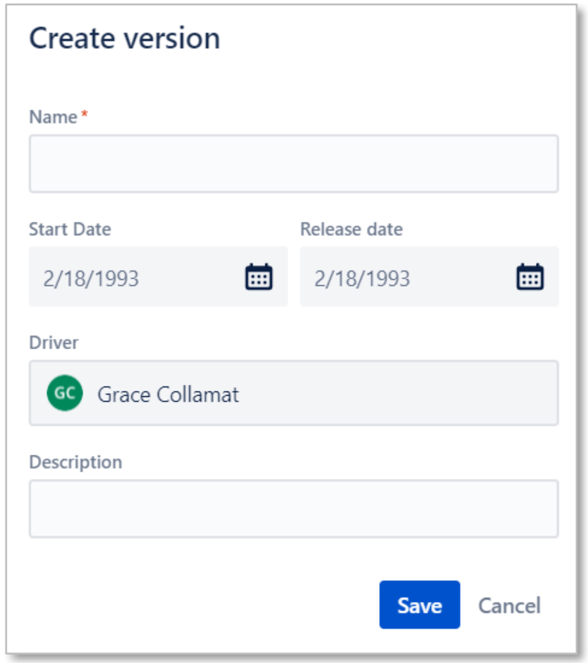
August 23, 2023

Outline

I. Create a version in Jira

You must have project administrator permissions to create or edit a version.

1. From your project's sidebar, select **Releases**.
2. From the releases page, Select **Create version**.



Create version

Name *

Start Date: 2/18/1993

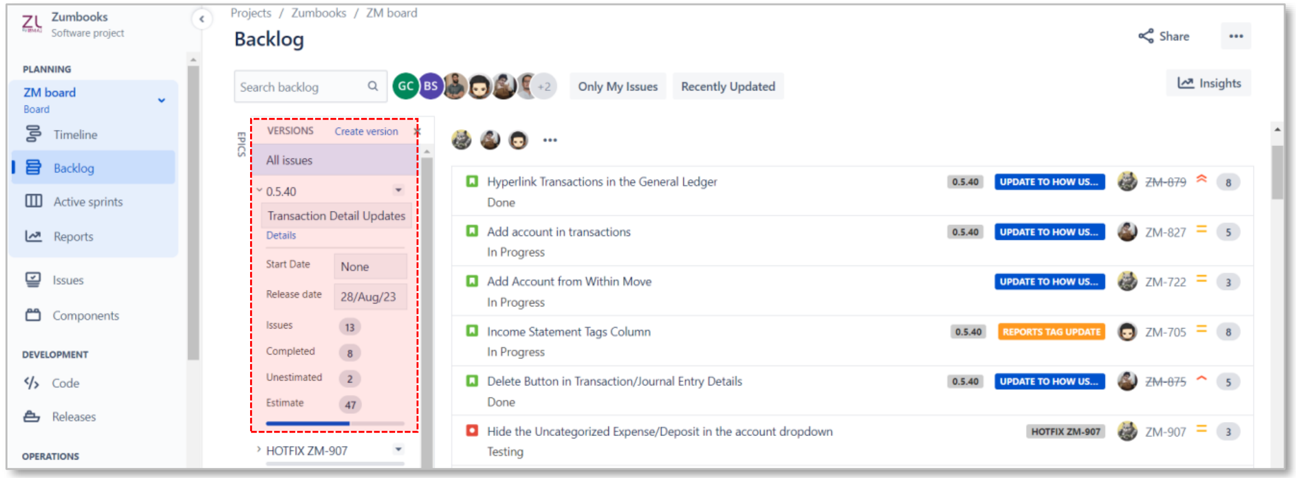
Release date: 2/18/1993

Driver: GC Grace Collamat

Description

Save | Cancel

3. Add issues to a **version.**
   - Navigate to the project **Backlog.**
   - Open the **Versions** panel on the left.
   - Drag issues to add into a version.



Zumbooks
Software project

PLANNING
ZM board
Board
Timeline
Backlog
Active sprints
Reports
Issues
Components

DEVELOPMENT
Code
Releases

OPERATIONS

Projects / Zumbooks / ZM board

Backlog

Search backlog | Only My Issues | Recently Updated | Share | Insights

VERSIONS | Create version

All issues

0.5.40
Transaction Detail Updates
Details

Start Date	None
Release date	28/Aug/23
Issues	13
Completed	8
Unestimated	2
Estimate	47

HOTFIX ZM-907

Hyperlink Transactions in the General Ledger Done	0.5.40	UPDATE TO HOW US...	~~ZM-879~~	8
Add account in transactions In Progress	0.5.40	UPDATE TO HOW US...	ZM-827	5
Add Account from Within Move In Progress		UPDATE TO HOW US...	ZM-722	3
Income Statement Tags Column In Progress	0.5.40	REPORTS TAG UPDATE	ZM-705	8
Delete Button in Transaction/Journal Entry Details Done	0.5.40	UPDATE TO HOW US...	~~ZM-875~~	5
Hide the Uncategorized Expense/Deposit in the account dropdown Testing		HOTFIX ZM-907	ZM-907	3

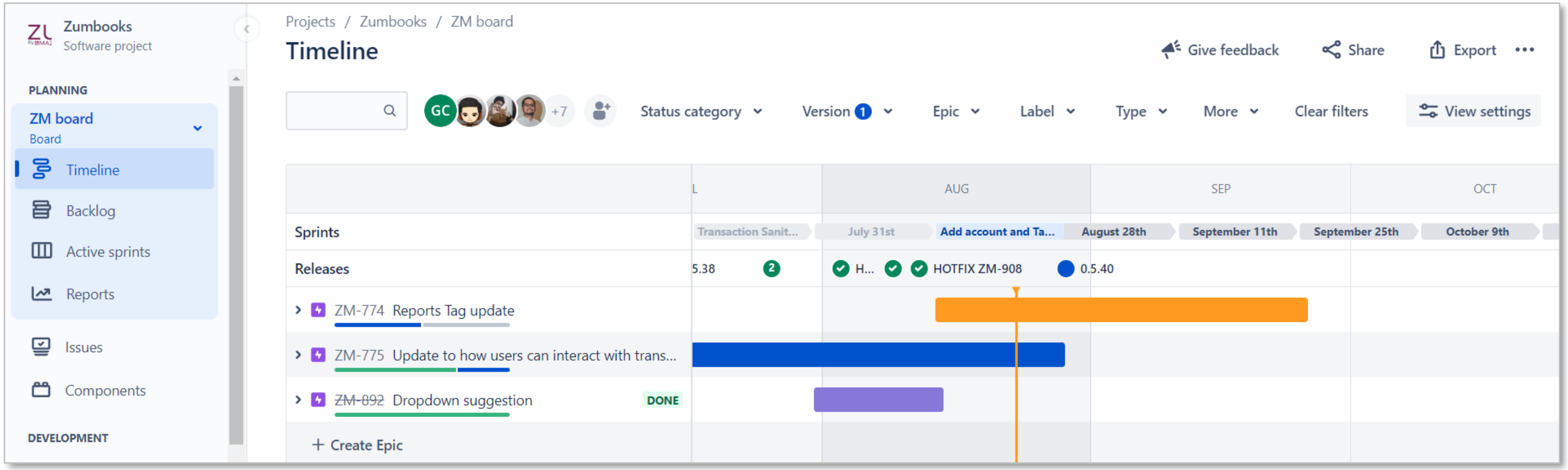
Key Points:
- Versions will serve as a ticket monitoring.
- Version names are typically numeric.
- Bug fixes and features can be put together within the same version.
- Create a separate version for hot fixes.

II. Track the progress of a version

We can monitor the progress of our versions in the **Timeline** view in Jira Software.

Each release version status can be easily identified based on the release icon:

- A **blue** circle indicates the release is UNRELEASED and has not passed its release date.
- A **red** circle with an exclamation point indicates the release is UNRELEASED and overdue.
- A **green** circle with a checkmark indicates the release has been RELEASED

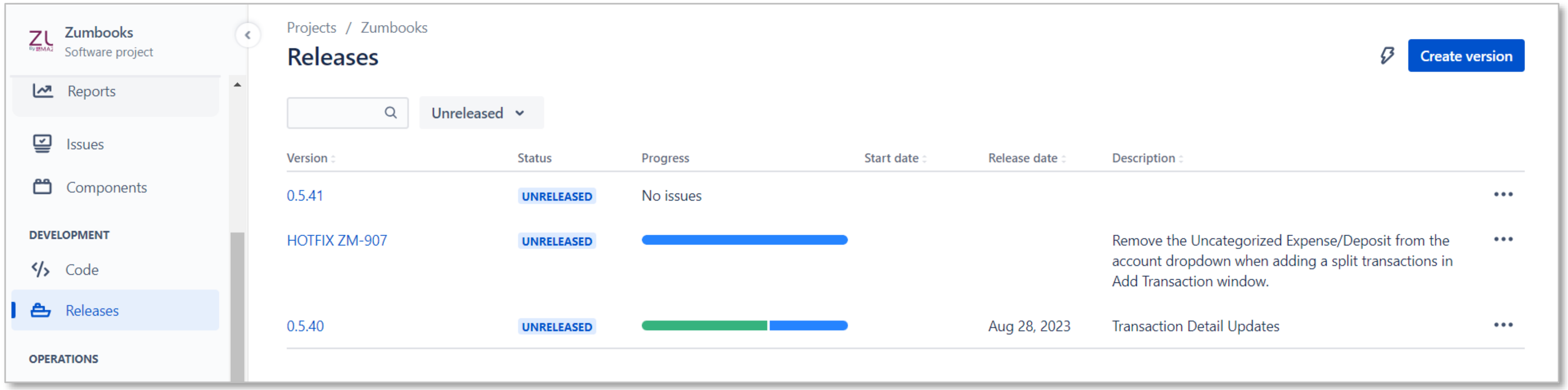


III. Manage release versions

We can easily manage all created release versions through the release hub. This hub provides status updates on our releases and a breakdown of the numbers of issues in each version. In this section we can also add and remove issues within a version.

To navigate to release hub:

- Go to your project, In the project menu, select “**Release**”



Version	Status	Progress	Start date	Release date	Description
0.5.41	UNRELEASED	No issues			
HOTFIX ZM-907	UNRELEASED				Remove the Uncategorized Expense/Deposit from the account dropdown when adding a split transactions in Add Transaction window.
0.5.40	UNRELEASED			Aug 28, 2023	Transaction Detail Updates

To move or remove issues from version:

- Select a version from the list
- Navigate to the "Issues" section.
- Click the kebab menu located next to the issue you wish to move or remove, then select options for moving or removing.

IV. Setup for Local Testing

Zumbooks

- Jira Software
- GlobalProtect
- SmartGit
- Laragon
- php-7.4.19-Win32-vc15-x64
- Node v16
- Postman
- MongoDBCompass
- VS code editor

Client Portal

- Jira Software
- GlobalProtect
- SmartGit
- Laragon
- php-8.1.9-Win32-vs16-x64
- Node v14
- VS code editor

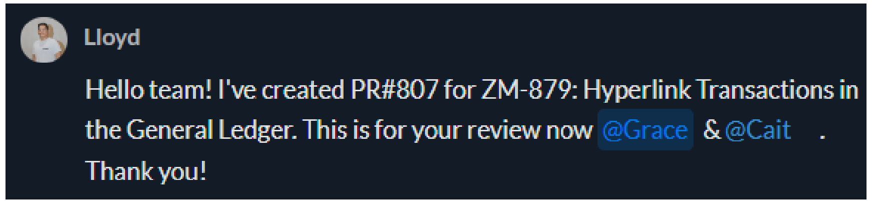
MSA

- Jira Software
- GlobalProtect
- SmartGit
- Laragon
- php-8.1.9-Win32-vs16-x64
- Node v16
- VS code editor

V. Features and Bug fixes Release Process

1. Code Review

- The ticket assignee will create **Pull Request (PR)**, then submit it for **code review** and **QA testing** (done locally). At this point the team lead will update the ticket status to **'IN REVIEW'.**
- If PR passes the code review, the team lead will then approve it for QA testing and update the ticket status to **'TESTING'.**

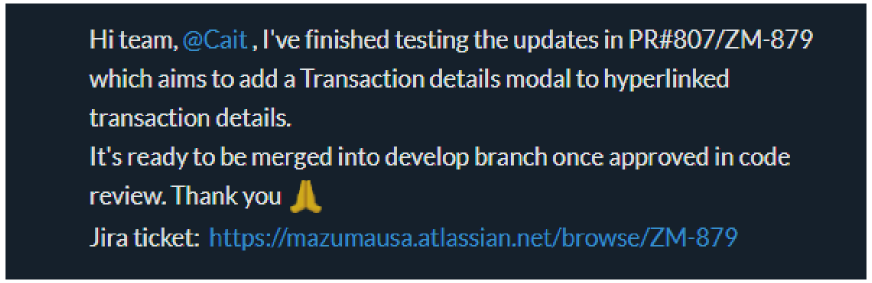


Lloyd

Hello team! I've created PR#807 for ZM-879: Hyperlink Transactions in the General Ledger. This is for your review now @Grace & @Cait .
Thank you!

2. Local Testing

- If bugs emerge during local testing, the tester will *notify the task assignee*, so they can address the issue and make necessary updates to their branch.
- If the PR passes the local testing, the tester will notify the team lead to ***merge*** it into the **Parent branch** in the case of a **Subtask**. But if it's **not a subtask**, the PR will be merged directly to the **develop branch**.



Hi team, @Cait , I've finished testing the updates in PR#807/ZM-879 which aims to add a Transaction details modal to hyperlinked transaction details.
It's ready to be merged into develop branch once approved in code review. Thank you 🙏
Jira ticket: https://mazumausa.atlassian.net/browse/ZM-879

Key Points:
- All subtasks must pass QA testing and be merged into their parent branch, before merging the full user story or parent branch into develop branch.
- A Bug fixes or a non-subtask tickets can be merged directly into develop branch if it passes QA testing.

3. Merge 'develop' branch into Master-Dev-Server

Once the tickets are merged into develop branch, they can now be pushed to the Dev server for further testing and UAT.

Steps:

- Make sure that your are connected to the server (GlobalProtect VPN).
- Start the Laragon, make sure that you have switched to the correct php and node versions.
- Open SmartGit, and ensure that you are in the correct repository.
- Check-out to **develop** branch and click **PULL** button (to pull the updates from the develop remote repository to your local develop branch).
- Check out '**Master-Dev-Server**' and **PULL** updates.
- Right-click '**develop**' branch and choose '**Merge**'. A modal will pop-up, select '**Create-Merge-Commit**'.
- Right-click '**Master-Dev-Server**' and click '**PUSH**' (to push the updates to Dev Server)
- After successfully merging commit, open **Laragon terminal** and execute the command '**npm run dev**', wait until build process is completed.
- After build process, three files will be generated in SmartGit which are ***.css, .json,*** and ***.js*** file.
- Discard the ***.css,*** and ***.json*** file and retain only the ***.js*** file.
- Right-click ***.js** file* and select '**commit'**, then input commit message and click **'Commit**'.
- Finally right-click '**Master-Dev-Server**' and click '**PUSH**'. After successful commit, check if the changes are reflected in the dev server.

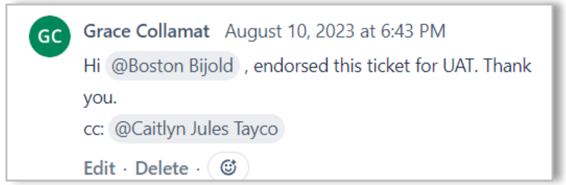
Key Points:
- Make sure that your local branch is always updated.
- Commit message should be PR# or ticket key (for user story).

4. Dev Testing

- Once updates are deployed to the Dev Server, the Tester will provide progress updates on the 'Mazuma DevOps' channel to ensure that everyone is well-informed.
- Tester will also execute Dev testing to validate that the newly added features are align with the expected output defined in the Acceptance Criteria.
- If any related bugs emerge during this testing phase, the Tester will create a bug ticket and another cycle of ***code review***, ***testing***, and ***merging processes*** will be initiated. This cycle will continue until bugs are resolved. *(Repeat procedure 1 to 4)*

5. USER ACCEPTANCE TESTING (UAT)

- Once Dev Testing phase is done, tickets will be forwarded to Project Manager (PM)/Stakeholders for User Acceptance Testing (UAT).
- The Tester will update the ticket status to "**USER ACCEPTANCE TESTING**" and notify PM/Stakeholders by tagging them in the comment section of the ticket.



GC Grace Collamat August 10, 2023 at 6:43 PM

Hi @Boston Bijold , endorsed this ticket for UAT. Thank you.

cc: @Caitlyn Jules Tayco

- If PM/Stakeholders encounters any bugs during UAT, another bug ticket will be created (if necessary) and another cycle of ***code review, testing***, and ***merging processes*** will be initiated until the tickets are bug free. *(Repeat procedure 1 to 5)*
- If the ticket/PR passes UAT, PM/Stakeholders will update the ticket status to "**DONE**".

Key Points:
- Make sure that the changes reflected in Dev server for each added feature should be in-line to the Acceptance Criteria.
- Monitor the version to check the progress and ticket status.
- Once all the tickets within the version are marked as 'DONE', we can proceed with the Production release.

6. Create Release (SmartGit)

Once all tickets within a Sprint/version are marked as "**DONE**", we can start the release deployment to production.

Steps:

- Connect to mazuma server (GlobalProtect VPN)
- Start the Laragon, make sure that you have switched to the correct php and node versions.
- Open SmartGit, and ensure that you are in the correct repository.
- Check-out to **develop** branch and click **PULL** button (make sure to pull updates from develop branch/ develop branch is updated).
- Click the **'v'** letter in Git Flow. Choose the **Start Release** option. A modal will pop-up, input the **Release Name** then click **Start**.
- Right-click the release then hit **Finish Release.**
- To check if release was successfully created, check the **develop** branch and **master** branch (*their Journal View updates should be equal).*

Key Points:

- Ensure that the release versions in JIRA and in SmartGit have the same name to easily track tickets and releases.
- Develop branch = master branch

7. Push the Release Update to Master-Dev-Server

Once the release has been created, merge the updates from the **‘develop’** branch to **Master-Dev-Server.** To merge develop branch to Master-Dev-Server follow the steps in Release Procedure no. 3.

8. Push the Release Update to Master-production-server

To deploy updates into Production server we have to merge the updates from the ‘**master**’ branch to ‘**Master-Prod-Server**’.

Steps:

- Make sure that your are connected to the server (GlobalProtect VPN).
- Start the Laragon, make sure that you have switched to the correct php and node versions.
- Open SmartGit, and ensure that you are in the correct repository.
- Check-out to **‘master’** branch and click **PULL** button (to make sure that the branch is updated).
- Check out ‘**Master-Prod-Server**’ and **PULL** updates.
- Right-click ‘**master**’ branch and choose ‘**Merge**’. A modal will pop-up, select ‘**Create-Merge-Commit**’.
- Right-click ‘**Master-Prod-Server**’ and click ‘**PUSH**’ (to push the updates to Prod Server)
- After successfully merging commit, open **Laragon terminal** and execute the command ‘**npm run prod**’, wait until build process is completed.
- After build process, three(3) files will be generated in SmartGit which are ***.css, .json,*** and ***.js*** file.
- Select all 3 files (***.css, .json*** and ***.js***), Right-click and select ‘**commit’**, then input commit message and click **‘Commit**’.
- Finally right-click ‘**Master-Prod-Server**’ and click ‘**PUSH**’. After successful commit, check if the changes are reflected in the Prod server.

VI. Hot fixes Release Process

1. Code Review

- The ticket assignee will create **Pull Request (PR)**, then submit it for **code review** and **QA testing** (done locally). The team lead will update the ticket status to **'IN REVIEW'.**
- If PR passes the code review, the team lead will then approve it for QA testing and update the ticket status to **'TESTING'.**

2. Local Testing

- If bugs emerge during local testing, the Tester will *notify the task assignee*, so they can address the issue and make necessary updates to their branch.
- If the PR passes the local testing, the tester will notify the team lead to ***merge*** the HOTFIX ticket to both **master** and **develop branches**.

3. Merge 'develop' branch into Master-Dev-Server

Once the hotfix ticket is merged into develop and master branches, it should be pushed to the Dev server for further testing and UAT. To merge develop branch to Master-Dev-Server follow the steps in *Feature and Bug fixes Release Procedure no. 3.*

4. Merge 'master' branch to 'Master-Prod-Server'

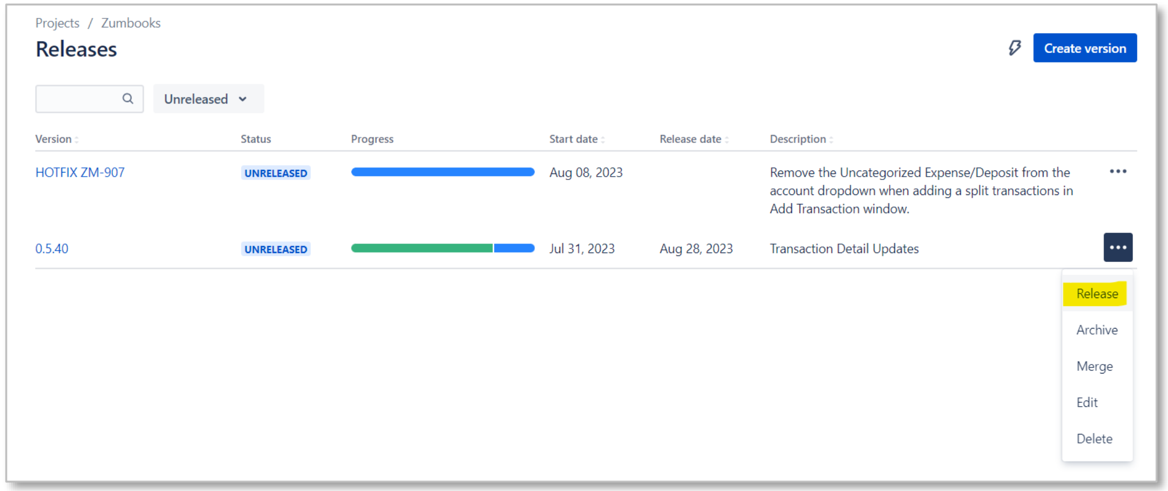
Once the hotfix ticket is marked as "**DONE**", then it is ready to be deploy into Prod Server. To release hotfix into Prod Server please follow the steps in *Feature and Bug fixes Release Procedure no. 8.*

VII. Complete a version

Following the Production deployment using SmartGit and confirming that all the issues within a version are marked as done, have been code reviewed, merged and have successfully pass QA testing and UAT, we are now ready to update our version status as RELEASED.

To complete a version

1. Navigate to your project
2. In the Project menu, select **Release**
3. For the version you want to release, select **Actions** (●●●) and click '**Release**'



Projects / Zumbooks

Releases

Version	Status	Progress	Start date	Release date	Description
HOTFIX ZM-907	UNRELEASED		Aug 08, 2023		Remove the Uncategorized Expense/Deposit from the account dropdown when adding a split transactions in Add Transaction window.
0.5.40	UNRELEASED		Jul 31, 2023	Aug 28, 2023	Transaction Detail Updates