Refinement Notes

In order to make it possible to implement as much as possible within the remaining time of the internship, the work has been simplified and split into epics and smaller issues. Below is a wireframe from version7 as well as some notes from the feedback received during the review that affects the implementation.

Graphical user interface, application

Description automatically generated

Image . V7 that was approved during review meeting.



Image Example of a column headers to be implemented by the student.

# Refinement Session 1

* Smart Table
  + Create date column with filter (use correct name)
  + Feedback Status column with sorting option
  + Kebab menu per row (with manage goals & collections and delete options)
  + Make “+Add Evidence” a ghost button (not primary)
  + Create Linked Goals column with a filter option (drop down with a list)
  + Redo the Linked Collections column with a filter option (drop down with a list)
  + Add tooltip of evidence Type icons to indicate the type in text on hover
  + Alternate colours between rows to improve readability of large amount of evidence

The refinement is completed using Zenhub, a tool that helps keep track of tickets. It is convenient due to its integration with GitHub. Below is an image of the first refinement board in progress.

A screenshot of a computer

Description automatically generated with medium confidence

Image . First refinement session results in Zenhub.

# Refinement Session 2

A screenshot of a computer

Description automatically generated with medium confidence

Image . Updated evidence wireframe to be implemented.

A screenshot of a computer

Description automatically generated with medium confidence

Image . Progress made on Evidence Page by the time the second refinement took place.

2. EPIC: Dashboard

- Add a new feature flag for the dashboard

- Define the shape of the response object (that API will return)

- Register in API documentation

- Define the path for the request

- Create an endpoint on the request path that provides the response object with test data inside

- Create Evidence Counter (only total evidence) Frontend

- Make a query for the total evidence of the user and pass it to the response object

- Plan a session with Elise to improve the look of the dashboard and make it more AntD

- Linked to Collections Counter Frontend

- Make a query for the evidence linked to collections and pass it to the response object

- Evidence Type counters Frontend

- Make a query for each evidence type and pass it to the response object

- Feedback Status Counters Frontend

- Make a query for the evidence per feedback status and pass it to the response object

- Orphaned Evidence Counter Frontend

- Make a query for the orphaned evidence and pass it to the response object

- Evidence with No Description Counter Frontend

- Make a query for the evidence with no description and pass it to the response object

- Evidence without comments Counter Frontend

- Make a query for the evidence without comments and pass it to the response object

- Check how long the request is taking? Optimize.

- Make it responsive! And accessible?

# Refinement Session 3

Reorganized EPICs into the following:

* Convert table - change what we had as a list into the table view
* Improve table - add new things and improvements to the table
* Dashboard – separate feature flag with all dashboard things

Convert table:

* Manage Collections for an Evidence

Improve table:

* Display Goals that are Linked per Evidence
* Add filtering on type, collection, feedback, goals and date
* Add sorting

## Best case scenario:

EPIC: Interactive Dashobard

- Show only Evidence that is Linked to Collections in the table

- Show only Evidence that is Linked to Goals in the table

- Filter table per Evidence Type

- Filter table per Feedback Status

- Show only Orphaned Evidence in the table

- Show only Evidence with No Description in the table

- Show only Evidence without comments in the table