Linköping University | IDA
Master thesis | Computer Science
Autumn term 2016 | LITH-EX-A--16/006--SE

Adapting infinite-scroll with the user experience in mind

Jonas Karlsson
Martin Larsson

Supervisor, Rita Kovordanyi
Examiner, Magnus Bång

TABLE OF CONTENTS

TABLE OF TABLES

TABLE OF FIGURES

ABSTRACT

In a world where technology advances, it is important that the human-computer interface follows in the same pace, to maximize the user experience. This means that when the machines become more powerful and the services become more comprehensive, it is important that they still are adapted to the users.

In the web browsing world, a concept called infinite-scroll has become popular when trying to maximise the user experience. This technique uses JavaScript to dynamically load more content to a webpage whilst the user is scrolling without ever reloading the page, to achieve a more fluid experience. This technique is used widely in social media websites such as Facebook and Twitter among others.

However, this technique have been used very rarely on other types of websites. Why this could be the case and how infinite-scroll performs on an intranet will be described in this report. A quantitative study on how the user behaviour differs between a website with pagination and infinite-scroll has been conducted and analysed by gathering statistics through Google Analytics about how users interacts with the website in question. The study indicates that infinite-scroll can be useful on an intranet when adapted according to usability guidelines.

1 INTRODUCTION

The last couple of year's web development has moved to decrease the waiting time when the user interacts with a web page. This is done to achieve a more enjoyable and fluid experience on the web. The first step was the concept called single-page application (SPA), which means a web application that can be fitted into a single web page and the page itself is not reloaded at any time. Together with this another concept was created, which is called infinite-scroll. Infinite-scroll is used to be able to scroll through large amount of content instead of clicking to the next page to view more. The main benefits of this is faster browsing [1], and that it is touch-friendly [2], [3]. Infinite-scroll is however not adapted in every web page and some problems have been identified which could be the reason why. According to Google trends [4] the interest in infinite-scroll has been great since it really accelerated in 2011, it reached its peak during 2013 but have decreased a little since then. It may be explained by the fact that infinite-scroll isn't for every website or that the peak of interest was just reached. With the knowledge from this thesis, some indication why this has happened may be found.

This thesis is written at Valtech for a customer of theirs. Valtech specialize in digital projects concerning everything from strategy, concepts, design, development and optimization. Valtech is a global IT-consulting company with approximately 1600 employees. The company was founded 1993 in France, nowadays the company also operates in Denmark, Sweden, Great Britain, Deutschland, USA, South Korea and India. The name Valtech comes from the motto; "VALue through TECHnology"[5], [6]. Valtech develops many websites, where the decision to choose between pagination or infinite-scroll needs to be made. Staff at Valtech has pointed out some disagreements between developers, customers and users regarding the use of infinite-scroll.

In this thesis, a quantitative study of the user behaviour on an intranet of a health care company will be conducted regarding which pagination technique to choose, pagination or infinite-scroll. Three versions of this site will be evaluated simultaneously. These versions include; the current implementation of the intranet which is an infinite-scrolling website with a load more button, a version with pagination and finally, a specially designed version with infinite-scroll adapted to this specific intranet.

The intranet has a feed with news articles about what is going on in the company. This feed is presented in the middle of the start page, and will be the main focus for the evaluation. One of the goals with the website in question is finding specific information fast, for example, to get information about an event. Another goal is to get a good overview of a bunch of information, in other words, to scroll through the news feed and get an idea about what has happened today. This makes the site partly goal-driven, by finding specific information, and another part not so goal-driven when scrolling through the feed.

1.1 MOTIVATION

Today there are some confusions whether infinite-scroll or traditional pagination is the best alternative when developing certain websites [7]–[9]. Can infinite-scroll be adapted and optimized to fit every type of website or is infinite-scroll just not for everyone. The anticipation is that this thesis can be used as support for developers and designers when choosing pagination or infinite-scroll for similar websites. The anticipation is that this report can be used to get ideas of possible improvements or tweaks to infinite-scroll.

1.2 AIM

This thesis will compare three versions of the same website with different pagination techniques. This is done by analysing statistics from Google Analytics regarding how users interact with the website. These statistics will be compared to the goals defined by the owner of the website, and can then be used as a decision basis when deciding which pagination technique to use. The goals with the website are the basis for our research questions. This thesis will conduct if infinite-scroll can meet these objectives better than traditional pagination and if it is possible to make a specially designed version of infinite-scroll for an intranet that works better than the other versions.

1.3 RESEARCH QUESTIONS

- Is it possible to get users to view more content in the same timespan on the website with the right implementation of infinite-scroll, compared to pagination?
    - How could loading and unloading of content be implemented to serve this purpose?
    - Will a working back-button help the user to navigate faster in the feed?
- Is it possible to get users to find specific information in less time with the right implementation of infinite-scroll, compared to pagination?
    - Can an indicator of where the user is in the feed and where it ends help the user to find specific information?
    - Is it possible to find specific information faster with the ability to bookmark content?
    - Can functionality to remember previously seen content help the user to find specific content faster?

1.4 LIMITATIONS

This thesis will conclude if an adapted version of infinite-scroll can meet the objectives above better than the old version of infinite-scroll and traditional pagination on this specific website. Several changes and updates were made in the adapted version and this thesis will not evaluate each update individually, but instead determine how an adapted version compares to the other versions by using statistics from Google Analytics.

2 THEORY

This chapter enlighten the strengths and weaknesses of infinite-scroll, how it is implemented and when it should be used. It also becomes clear which areas of improvement exists for infinite-scroll.

2.1 BACKGROUND

When there is more content than what fits in a page, there are two solutions that are common for solving this. There is pagination, where you view a part of the results on the current page and then the user clicks on a next button to see a new page with another part of the results. The other technique is called infinite-scroll and has become quite popular on the latest time [4]. In this thesis, infinite-scroll is defined as a technique that dynamically loads new content on to the page when the user is scrolling through the page. This is achieved via a "load more"-button or continuously whilst the user is scrolling down the page [8].

Infinite-scroll works well on websites which have a lot of content with a flat structure, where each piece of content has the same potential to be equally valuable as the other, e.g. social media [8]. Earlier studies shows that this technique is good for time-killing activities because the users tends to stay longer on a page with infinite-scroll [10]. However, infinite-scroll seems to lead to a more passive type of browsing with fewer clicks [7] which is not optimal for goal-oriented activities such as at an e-commerce site for example. Also when using infinite-scroll, users can feel overwhelmed by the endless stream of content which can make them less intrigued to engage with the website [9].

Although this technique have been used somewhat widely a few problems have been identified by several sources [2], [3], [8], [9]:

- When the user is looking for the perfect search result and keep scrolling through an endless sea of irrelevant information. Chances are that the best result will appear in the beginning, but because of the infinite-scroll, the user gets tempted to continue scrolling. When the user have given up, or stopped scrolling, it is hard to find a previously found result through all the other loaded content.
- The fact that the scrollbar does not work as intended. It looks like you are always near the bottom and the user can get tempted to keep scrolling a little bit more.
- By combining the two points above, the user may feel exhausted because he/she keeps scrolling but there is no end.
- When clicking a link in an infinite scrolling page and then returning from the link, it often causes the user's position to get lost and the user may feel disoriented.
- It is easy to get the feeling that the user is missing out on valuable information because there is always more content to load and there is always a little bit left on the scrollbar.
- Infinite-scroll can lead to fewer clicks.
- No ability to skip results.
- The footer is unreachable in a traditional infinite-scroll implementation.

2.2 PROPOSED SOLUTIONS FROM RESEARCH

During the pre-study phase of this thesis, the goals with the website were examined to find out which of the problems above is related to the website's goals. These problems are defined as Areas Of Improvement (AOI) and will be implemented. These suggested solutions is gathered through research and represents the available solutions. These solutions will be evaluated and the best fit will be chosen.

2.2.1 LOADING CONTENT

The main part of infinite-scroll is that content is loaded continuously while the user scrolls down a page. When the loading of new content should be performed has been solved in some different ways.

One common solution is to load more content when the user is near the end of the page and is used by many popular social networks like Facebook, Twitter and Flickr. One guideline indicates that the new content should be loaded when 100-500px of the page is left [11].

Another similar solution is to load more content when the user has reached the bottom of the page, in other words, when 0px is left of the page. Together with this solution a waiting time and a loading bar can be implemented to visualize the loading and give the user some time to stop and take a breath. The aliases Pathfinder and Janpaepke have created two proof of concepts illustrating this. [12], [13]

Another solution can be what is implemented on the website under investigation. This website has a current solution which consist of a "load more" button that loads more content when the user clicks the button. This button appears typically at the bottom of the page.

2.2.2 UNLOADING CONTENT

When loading endless amount of content, at some point some content needs be unloaded. Otherwise the browser will eventually run out of RAM and crash. Trunal Bhanse describes some possible solutions that LinkedIn used [14], these are described below.

Images is one of the elements that are using the most memory, by unloading images a lot of memory can be released. The proposed solution is to replace the image on the website with a very small image, when the image is not visible on the screen.

Another solution is that, for the elements that are not visible on the screen, the *visibility* property in CSS can be set to *hidden* to release some memory. In other words, pages will be hidden when they are out of the screen.

This problem can also be solved by removing the elements that are not visible on the screen. But in these cases the element is often replaced by an empty stub with same dimensions to prevent the scrollbar from breaking.

Another way to handle this is to make the infinite amount of content finite instead. When scrolling on e.g. Facebook you will eventually reach an end, at that point no more content is loaded. Therefore they don't need to unload content.

2.2.3 BOOKMARKING AND MAKING THE BACK-BUTTON WORK AS INTENDED

As mentioned earlier, there are problems when having an infinite scrolling page with bookmarking somewhere in the feed. If the user press an article and enters a new page the earlier position is forgotten. That means, when the user press the back-button he/she is commonly returned to the start of the feed.

To handle this, Discourse and Appspot have come up with a solution that updates the URL address for each article that scrolls past the top using the HTML5 History API [15], [16]. This means that the user's position is always saved and if the tab is reopened or the back-button is pressed, the position is always saved. This does also mean that every article and every change is saved in the browser history. To determine if this is good or bad is up to the user.

There are some other solutions discussed in the forum called Quora, these solutions include, adding a special URL with the position to the history every time the user leaves the page. This will make the back-button work as normal, but the user will not be able to bookmark or send a link to a specific place in the feed. Another possible solution is using *localStorage* or cookies to remember where the user left off [17].

Websites like Pinterest and Twitter uses a different approach. Instead of opening articles in a new page, they open it in a modal. This means, that the user never leaves the page, which means that they don't have to use the back button, however, the bookmarking, and linking by copying the URL does not work.

Another workaround that is quite common on infinite-scroll websites (such as 9gag) nowadays are opening new pages in new tabs. This means that the page is forced to be opened in a new tab. In that way the user does not need the back-button. However this does not solve the bookmarking or linking issue.

2.2.3.1 AN INDICATOR OF WHERE THE USER IS IN THE FEED AND WHERE IT ENDS

A solution to this is using some kind of page notation like Discourse and Appspot. This means that for every, or a certain number of articles, the page, or article count is updated. Then somewhere on the website there is a view

of how many pages/articles that are left and the user can skip to a certain page, the top or the bottom. On Discourse there is even a progress bar to show the user how much is left. [15], [16]

The sausage.js framework provides an interesting take on what can be done with the scrollbar on an infinite scrolling page. It takes place on the far right. Where it divides the page in little chunks. Every chunk corresponds to a page view. These chunks are clickable and give some indications of where the user is in the feed [18].

2.2.4 Functionality to remember previously seen content that can be worth seeing again

Kristine Schachinger enlightens this problem in her article. She says that it is hard to find where you are and where recently clicked content is located. To resolve this, clear reference points (for example article or page numbers like Discourse and Appspot) are needed, or like Pinterest where the user can pin certain posts and make them appear on a different view [19].

2.3 Frameworks and plugins for infinite-scroll

There exist some frameworks and jQuery plugins addressing this technique. Most of them work basically in the same way. The most common solution is to listen for a scroll event. For each scroll event, compare the user's position with how much data that is already loaded. If a user is below a specific number of pixels on a page, a function is called for loading new content. How often content is loaded and how much to load each time differs between website preferences [11].

2.4 Usability

Nigel Bevan is defining usability as the object to achieve quality of use. In other words, can the user achieve the intended task with this product? This relates usability to the objectives with the site, which is of great importance in this thesis. He also states that the difference with usability to other types of design is the focus on the human issues. Usability requirements should be stated in terms of effectiveness, efficiency and satisfaction [20]. To achieve good usability Jeff Johnson's design rules will be followed, that often describe goals rather than actions. This makes them very general and broadly applicable. He states these 12 design guidelines to follow [21]:

- We perceive what we expect
- Our vision is optimized to see structure
- We seek and use visual structure
- Reading is unnatural
- Our colour vision is limited
- Our peripheral vision is poor
- Our attention is limited; our memory is imperfect
- Limits on attention shape thought and action
- Recognition is easy; recall is hard
- Learning from experience and performing learned actions are easy; problem solving and calculation are hard
- Many factors affect learning
- We have time requirements

2.5 Evaluation

The implementations in this thesis will be evaluated using the following techniques. A quantitative study will be made where statistics regarding the user behaviour will be gathered through A/B testing via Google Analytics. A survey will also be held to find out the thoughts of the users as well as their behaviour.

2.5.1 Quantitative study

Suphat Sukamolson is defining quantitative research as:

> *"Quantitative research is about explaining phenomena by collecting quantitative data, which are analysed by mathematically based methods."*

He also states four main type of research questions that is well suited for quantitative research where hypothesis is one of them [22], when you want to explain something. This thesis consist of the hypothesis that solving the AOI can make the site to meet its goals better.

This thesis will contain a quantitative study using A/B testing together with Google Analytics to track user behaviour.

2.5.2 A/B testing

A/B testing is a basic testing technique that compares two (or more) versions (A and B) with a metric that decides the success of a version, and then the most successful one can be chosen. The versions should be tested simultaneously, were different users get different versions. A/B testing have been proven to be a good choice by case studies, when implementing new features and through the whole implementation phase [23].

2.5.3 Google analytics

Google Analytics is a free analytics tool that collects data which can be used to track user behaviour on websites. It can also be used to measure sales and conversions. Google Analytics collects data about how the user interacts with the site, how they arrive and when they leave and much more. This is achieved by tagging a visit on a page as a page view, sending events for user-interactions on the site (for example clicking a button). Many metrics exists to be able to measure the user behaviour from the collected data, for example how many page views a site has, and how long time do the user spend on the site, from which page of the site is the user leaving. Google Analytics then provides an interface to view and filter this data, which is called reports [24], [25].

Google Analytics is proven to be a good tool to identify usability problems on a site. 13 metrics have been proposed that can be used to measure the usability of a site [26]. Some of these are domain specific and not all can be used in this case, which metrics are chosen and an explanation to these metrics will be presented in Figure 2 in the results section of the report.

2.5.4 Survey

The goal with the survey is to add another type of data, to find out what the user thinks about the improvements to the AOI. The user behaviour is collected via Google Analytics and the users' opinion is collected through a survey. The focus in the report is on the analysis of Google Analytics with the survey to be used as a complement.

3 METHOD

The work in this thesis was divided in to three phases. A pre-study phase where the problems and improvements about infinite-scroll where found and ideas how to improve the infinite-scroll concept. The implementation phase describes how the improvements were implemented. The last phase was the evaluation phase which describes how improvements were evaluated to answer the purpose of this thesis.

3.1 PRE-STUDY

By having a discussion with the supervisor for this project and another co-worker, it was obvious that the developers were unsure if pagination or infinite-scroll was preferred in certain types of websites. After identifying that there are uncertainties with this, a website which was suitable for research was identified. An interview with the product owner of the website was held to find out the overall goals with the website. The goals with the website were to have an effective communication channel and to spread and find information quicker. The hypothesis is that these goals can be improved by making it easier to get an overview of the feed, and by giving the user a chance to find specific information in less time. These goals are the main focus of the report. A measurement plan was created to identify the goals with the evaluation and is found Figure 2 in the results section. This will be used to know how the evaluation should be performed when using Google Analytics and what to measure. The measurement plan consists of six parts:

- **Business Objective:** Describes on an abstract level what the goal with the implementation is.
- **Strategy:** Describes the parts of the business objective in more detail.
- **Tactic:** Describes how to achieve the goals of the strategy
- **Key performance indicator (KPI):** Describes what will be measured to determine if the strategies and business objective is met
- **Segments:** Describes how to segment the data, not used in this implementation because all the users have similar environments and only desktop users are relevant.
- **Targets:** Set up targets for each KPI to determine what results are expected. This is not relevant in the report because each version will be compared to the other versions. [27]

3.1.1 GOOGLE ANALYTICS

For the evaluation, a segment for each version was defined in Google Analytics. Then a dashboard was created containing a widget for each KPI defined in the measurement plan (as seen in Figure 2), except for KPI 5, 6, 11. These were defined as special segments filtering out the data which does not contain a specific sequence of events.

3.2 IMPLEMENTATION

When the pre-study was finished the implementation-phase took place. The test environment already had an implementation of infinite-scroll. A solution with pagination was implemented and another version was created with an adapted infinite-scroll. On the version with the adapted infinite-scroll, improvements to the following areas were applied, called Areas Of Improvement (AOI):

- Loading and unloading content
- Bookmarking and make the back-button work
- An indicator of where the user is in the feed and where it ends
- Functionality to remember previously seen content that can be worth seeing again

The solutions to these bullets above was gathered from articles, blogs, etc. If the source is not strong enough, other (more general UX) literature was used as a compliment to the suggested solutions. When all suggested improvements were made to the adapted version the three versions was compared with each other using A/B testing and then by evaluating the data about user behaviour from Google Analytics.

3.2.1 SYSTEM OVERVIEW

The current website is an MVC backend implemented in ASP.NET with a frontend consisting of HTML, CSS and JavaScript. A content management system called EPiServer is implemented to enable administrative users to

add new posts to the news feed which is placed in the middle of the start page. This news feed is the main focus of the report.

3.2.2 CONTENT LOADING

All content cannot be loaded at the same time, because the loading of the webpage will not end since the idea is that the amount of content is infinite. Another problem is that it will take more time to load and render more content. Therefore the content needs to be loaded dynamically as the user scrolls down the page. So the content is grouped into sections of a given section size. In other words the website has a representation of a page consisting of five posts. JQuery is used to detect if the user is scrolling, and compares the scrollbar position with the height of the website. When the scrollbar is at the bottom, the scrollbar position is equal to the height of the website. From this, a condition can be created so that when the user is near the end, new content is automatically loaded. This is decided from a given number of pixels to the bottom. According to suggested solutions a recommendation is somewhere between 100-500 pixels [11]. To make a decision about which value to choose, an investigation on how different values affects the performance and usability was performed. Another decision needed to be made regarding how many posts should be loaded at a time, this was done by performance testing the function that loads more posts. This was achieved by creating a JavaScript that made 100 Ajax request and measured the average time doing one loading. Since the measured time per load didn't differ that much, 100 requests gives a good average value. This was done on both a wireless and a wired internet connection since a performance dropdown was noticed when scrolling using a wireless connection. The conclusion from these tests is that the pixel number when new content should be loaded was chosen to be 300 pixels, and 10 new posts should be loaded every time.

When the content loading was concluded the unloading issue was investigated. First some data about the webpage performance was gathered. From this data a decision could be made if unloading data is necessary or not. This was done by investigating how much memory the page uses, both in the worst case and in an average case. The memory usage was measured in Internet explorer, with the Memory tool found in the F12 Developer tool. A snapshot was taken when no content was loaded, and another was taken when all content (200 articles) were loaded. The second snapshot shows how much the memory usage increased when all content is loaded. Internet Explorer was chosen because 90% of the users on the site uses this browser, also Internet Explorer provides a good tool to measure the memory usage. The conclusion from this test was that unloading is redundant and will not be handled in this thesis.

Because of the use of automatic content loading the user will never reach the footer of the webpage. In this case, the website have important information in the footer, so it needs to be available. A solution consisting of a sticky footer that appears at the bottom of the browser was chosen.

3.2.3 BOOKMARKING AND MAKING THE BACK-BUTTON WORK

The chosen method for this, was updating the URL dynamically using the HTML History API. This can be done by detecting each scroll event and for every time an article gets scrolled out of the view, update the URL. However, this worked fine in the Google Chrome browser, because it fires a *$(window).scroll* event for each tick made by the mouse wheel. But the Firefox and Internet Explorer browser fires a lot more events for each scroll tick on the mouse. This makes the application feel slow and unresponsive when running a function that takes some time because each function has to complete before the next can be run.

Therefore, the solution was to make a function which listens for when the user has stopped scrolling. This can look something like this:

```
$(window).scroll(function() {
    clearTimeout($.data(this, 'scrollTimer'));
    $.data(this, 'scrollTimer', setTimeout(function() {
        UpdateUrl();
    }, 250));
});
```

This means that a delay had to be set before the URL updates. In this case it was set to 250ms. In that time the user won't be able to do any action before the URL is updated. Also this provides a scroll flow equal to the normal flow, because the URL update function is not called as often.

For each update, the JavaScript loops through all posts and determines which post are visible at the top. Then that id is taken and pushed to the URL using *history.pushState.* To determine how much more elements that needs to be loaded, two other variables is used to detect how many posts where loaded before and after the previously mentioned article. When the site is called with the URL arguments, it is detected in the controller. There, instead of loading the standard amount of articles, it loads all the articles that appears before said article and those that were loaded after. A constant determining how many posts to load as maximum has also been implemented.

3.2.4 Functionality to remember previously seen content that can be worth seeing again

The idea is that the user should be able to make a reference point to an article so that the user can remember previously seen content. The solution consists of a star icon that is located within the content, in this case to the right of the article title. By pressing this star a reference point is created to the content, and is toggled to be able to remove reference points. This reference point appears next to the scrollbar, in the implemented progress bar, as a small block that is clickable. By clicking this the user will be scrolled to the marked article. This is solved by saving the id of the star icon in the block reference point, and when that block is clicked, JavaScript will scroll to this id, which is the position of the star.

However, when the user refreshes the page or clicks on an article and then goes back, using the back-button, these reference points will be lost. To solve this, the reference points are saved in a cookie, which is loaded every time the page is loaded. So the saved reference points will not be lost before the cookie expires. This created another problem though, a reference point could be saved belonging to content that is not loaded yet. That means, scrolling to that marked article cannot be done until the article is loaded. This was solved by loading more content while automatically scrolling down the page, until the position of the marked star icon was reached. In more detail the functionality consist of JavaScript that checks if the star icon exists on the page, by using the jQuery selector of the star icon id. If not, the article has not been loaded and more content needs to be loaded. That means, more content will be loaded while JavaScript is scrolling down the page. This repeats until the star icon is within the page.

3.2.5 An indicator of where the user is in the feed and where it ends

To have an indicator of where the user is in the feed and where it ends a progress bar was implemented. This progress bar shows how much content the user has scrolled through, as a percentage of the total amount of content. With this information the user can see where the feed ends and where the user is right now.

This progress appears next to the scrollbar (as seen in Figure 3). The progress is updated every time the user scrolls the page. The progress is updated by a calculation of how many pixels has been scrolled divided by an assumption of how many total pixels the page will have if all content is loaded. This assumption is made from an assumed pixel height of an article and the total number of articles. From this calculation a percentage of the progress is created, this corresponds to the height of the progress level in the progress bar.

3.2.6 Pagination

One traditional pagination view is created to be used when testing. With this view, all infinite-scroll code was left out. So instead of the load more button in the original implementation a menu to select page is inserted (as seen in Figure 1). The single arrows returns the next or the previous page, and the double arrow returns the first or the last page. The underline indicates the current page.

Figure 1: Pagination

This menu was created from JavaScript that generates html code that consist of a link with a destination of a JavaScript function that makes an Ajax request of the specified page. The specified page is also added to the URL by using the *history.pushState* function, with a state object consisting of the page number. When the back-button is pressed, the history state is popped and the previous page can be displayed. That means that the back-button is working and the ability to bookmark a page exists, which is expected functionality in a traditional pagination view.

3.2.7 A/B TESTING

The ability to switch between three versions and perform A/B testing was implemented using Google Analytics. This means creating three different URL for the three versions. In this case it was separated by adding /version/[version number] as a path to the URL. Then an experiment was created in Google Analytics, which basically is another word for A/B testing. In the experiment the three versions are setup as variations. When the experiment is setup a code snippet in JavaScript is created that should be added to the original version only. With this code snippet, Google Analytics handles redirecting to the different versions. This includes handling a user in such a way that the user is redirected randomly to one of the three versions and when the user have been assigned a variation, the user sticks with it throughout the whole experiment. This gives the most equal distribution because all users are divided equally instead of for example if a certain department would test a version, the results would have been based on that department specifically instead of being based on users with all sorts of jobs and interests. The user sticks with the their version to prevent unusual behaviour when for example, trying to switch version.

3.3 EVALUATION

To evaluate the implementation, A/B testing was performed. The website has around seven thousand sessions a day, with around four thousand unique users. During the testing period each visitor was randomly directed to one of our versions of the site, and information from Google Analytics was gathered. This information was used to analyse the user behaviour. From this it was concluded which solution is preferred.

The main parameters that was used to answer the research questions is how many articles that are viewed, and if more articles are viewed in the same time with a specific implementation. Also, to answer the second research question, the session duration will be divided on the number of articles clicked. This means that if users engage more with the site and reads more articles in less time, it is fair to assume they are finding information faster.

There are also some other parameters that are not a direct answer to the research questions, but they are interesting, both for the company which owns the site and also because they could be a side-effect of infinite-scroll. These are the time spent on the site and the number of articles clicked per session. Because even if the research questions are fulfilled with some solution, and the optimal version is found, there could be side effects that are unwanted.

3.3.1 SURVEY

A survey was conducted to find out what the users thoughts about the adapted infinite-scroll were. Only data from the users that viewed the version with the adapted infinite-scroll was collected. The survey was performed in Swedish with these questions, translated into English:

- What do you think about the fact that articles are loaded automatically when you scroll?
- What do you think about the fact that the back-button takes you to the location you were when you clicked the article?
- What do you think about the indicator that shows how many articles you have loaded and how many is left?
- What do you think about the ability to mark an article with a star to have a reference point to take you back?
- What do you think about the footer that sticks to the bottom of the page?
- Do you prefer this version over the original one?

These questions were asked to be rated from very dissatisfied to very satisfied in a scale 1-5, and an option that the user cannot take a stand. One open question with a big text field was also asked, if the user wanted to add anything else.

This survey was added to the thesis to get the users' opinions about the changes. Because of the fact that a few different features were added instead of one, it is interesting to see what the user thinks about each feature and how these features collaborate with each other. These survey questions are used as a complement to the statistics gathered from Google Analytics. If, for example, one feature makes the users very unsatisfied, this could explain why the average number of articles clicked in that version is very low. These ratings could be used in future work to decide which features are of higher priority.

4 RESULTS

The outcome and the results from the pre-study is data from the interview, a measurement plan to be used for the evaluation phase and which problems to focus on. The results from the implementation phase contains an overview of the implementation that visualizes the changes made to the site. The evaluation results consist of data from Google Analytics about the user behaviour and the results from the survey performed.

4.1 PRE-STUDY

This section contains the results of the pre-study. These results are what the whole thesis is based on. It contains the goals with the site from the product owner's point of view. It also contains some identified problems that were found in the research phase and which of these problems this thesis will focus on. Finally, this section contains a measurement plan, describing how the potential improvements will be measured.

4.1.1 GOALS WITH THE SITE

The overall goals with the website was identified with the product owner to be the following:

1. Contribute to a strengthened community between co-workers
2. Facilitate the everyday work for employees
3. Effective communication channel
4. Spread and find information
5. Sharing competences
6. Find information via a mobile phone

4.1.2 IDENTIFIED PROBLEMS

As described in the background section of the theory chapter a few problems have been identified, which is summarized as:

- The user gets tempted to continue scrolling, and having problem finding previously seen content.
- The scrollbar does not work as intended.
- The user may feel exhausted because he/she keeps scrolling but there is no end.
- The users position is lost when returning to the feed, when refreshing the page or using the back button.
- It is easy to get the feeling that the user is missing out on valuable information because there is always more content to load and there is always a little bit left on the scrollbar.
- Infinite-scroll can lead to fewer clicks.
- No ability to skip results.
- The footer is unreachable.

4.1.3 MEASUREMENT PLAN

After identifying the goals of the website and some common issues with infinite-scroll a measurement plan was developed. The business objective is the overall goals with the website, in this case make the staff efficiently find and share information. Each strategy is a more concrete part of the business objective and are easier to measure. The tactics are identified as how to achieve the strategies. These tactics defines what parts of the website that needs change or improvement to reach the desired effect. The KPI: s are measured to evaluate a tactic.

Each strategy and each tactic are identified as a potential source of improvement and the goal is that they will both contribute to the objective. They can be evaluated both individually and together to determine if one is more important than the other or if they both are needed.

These are the strategies and tactics:

- **Strategy 1:** If the user can see more information, he/she can get a quicker overview and therefore browse the website more efficiently.
    - **Tactic 1:** By making the loading of new content automatic, it is possible to assume that users will see more data in less time and therefore get a quicker overview.
    - **Tactic 2:** By making the back-button to work as intended the users are returned to a place in the feed they expect and can from there easier continue browsing instead of first finding the old position and continuing from there.
- **Strategy 2:** When having a good overview it is important that it is possible to find specific information fast, otherwise the strategy could have the opposite effect.
    - **Tactic 3:** By having an indicator of where the user is in the feed, the users can focus more on specific content instead of searching for a view of how much information there is in total.
    - **Tactic 4:** By remembering previously seen content, it is easier for users to find this information again instead of searching for it once more.

Figure 2: Measurement plan

4.1.4 Areas of improvement

With regard to the identified problems with infinite-scroll and the goals of the site, these areas showed improvement potential and therefore they were chosen to focus on:

- Loading and unloading content
- Bookmarking and make the back-button work
- An indicator of where the user is in the feed and where it ends
- Functionality to remember previously seen content that can be worth seeing again

4.2 Implementation

This section contains an overview of the implementation showing what adaptions were made in this project.

4.2.1 Overview of the implementation

The news feed is located in the middle of the website and each post consists of a short description and a link to an article. As seen in Figure 3, a star is available at the top right of every article, this star can be checked. When a star is checked a mark will be created in the progress bar to the right, illustrated by the arrows. When this mark in the progress bar is clicked, the user will be scrolled to the marked article that corresponds to that mark. If a star is unchecked this mark will be removed. The progress bar is located to the right, next to the scrollbar. This displays the progress level, how much of the total content that have been scrolled through. The footer is located at the bottom as a traditional footer, but the new footer sticks on to the bottom of the browser. It will be faded in when the user have scrolled on the page, and will fade out if the user is near the top of the page.

Figure 3: Overview of the implementation

4.3 Evaluation

The evaluation was made by performing an A/B test on the website with the three different versions and data about how the users interacted with the site was collected with Google Analytics. A survey was used as a complement to collect the users' opinions about the adapted infinite-scroll version.

4.3.1 Statistics

The KPI's was used to measure how good the tactics performs, according to Figure 2. This is the main part of the evaluation.

Total Sessions. Because of the fact that users will keep their version throughout the whole test, the total amount of sessions can differ somewhat between the versions. This can be important to keep in mind, therefore Table 1 shows how many sessions the different versions received during the test.

	Original	Adapted	Pagination
Total sessions	10823	15414	15453

Table 1: Total sessions

KPI 1. Amount of extra articles loaded per session / session duration. Table 2 shows statistics from sessions with an occurring "load-more" – event. A "load-more" – event fires every time the page loads more articles. In the original version this happens when the load more button is pressed. In the adapted version, this happens when the user have scrolled near the bottom. Finally, this event occurs when changing page in the paginated version. *Event* in Table 2 below is defined as when a call occurs to load more articles.

	Original	Adapted	Pagination
Total amount of extra articles loaded	815	12200	2935
Total amount of extra articles loaded per Session	0,075	0,791	0,190
Sessions with Event	27	386	141
Sessions with Event / Total Sessions	0,002	0,025	0,009
Amount of extra articles loaded / Sessions with Event	30,2	31,6	20,8
Avg. Session Duration	00:01:26	00:08:07	00:08:01
Avg. Sessions with Event duration	00:07:16	00:13:21	00:18:48
Extra articles loaded per second	0,000872	0,00162	0,000395
Extra articles loaded per second (Sessions with event)	0,0693	0,0395	0,0184

Table 2: KPI 1

KPI 2. Session duration. Table 3 shows the average session duration for all sessions. This is important to keep track of when evaluating other KPI's because the goals with the thesis is to make the users more effective. Because of that, shorter session duration is better in this case. A session is defined as a group of interactions that take place within a given timespan. The session has a timeout set to 30 min by default, which means that the session ends after a user has been inactive in 30 min. Not counting the inactive minutes in the Session duration though [28].

	Original	Adapted	Pagination
Avg. Session duration	00:01:26	00:08:07	00:08:01

Table 3: KPI 2

KPI 3. Number of articles pressed per session. By counting the article clicks it can be determined how active the user is on the different variations of the website.

	Original	Adapted	Pagination
Total Events	482	1252	1282
Sessions with Events	359	628	664
Total Events / Sessions with Events	1,34	1,99	1,93
Total Events / Total Sessions	0,0445	0,0812	0,0830

Table 4: KPI 3

KPI 4. Number of Back-button clicks. By counting the number of back-button clicks, it can be determined how the user reacts on where it is taken on the page when using the back-button (as seen in Table 5).

	Original	Adapted	Pagination
Total Events	685	1 001	1 614
Sessions with Events	170	160	236
Total Events / Sessions with Events	4,03	6,26	6,84
Total Events / Total Sessions	0,0633	0,0649	0,104

Table 5: KPI 4

KPI 5. Sessions where a user exits the page immediately after a back-button click. By identifying these sequences it can be determined if the user likes the way the back-button works or if the user just quits browsing because there position in the feed is lost, or not.

	Original	Adapted	Pagination
Total Sessions with sequence	1560	402	1563
Total Sessions with sequence / Total Sessions	0,144	0,0261	0,101

Table 6: KPI 5

KPI 6. Bounce Rate. Bounce Rate is defined as the percentage of single-page sessions (i.e. sessions in which the person left your site from the entrance page without interacting with the page) [29]. This is used to determine how the user likes the first impression of the homepage. If it is appealing and invites the user to interact with it or if it is the opposite.

	Original	Adapted	Pagination
Bounce Rate	9,14%	1,55%	1,66%

Table 7: KPI 6

KPI 7. How often the star button in the top right corner of an article is pressed. This is used to identify how often the user marks an article and from there it can be determined if the feature is worth implementing or not on other websites.

	Original	Adapted	Pagination
Total Events	0	3	0
Sessions with Event	0	2	0
Total Events / Total Sessions	0	0	0

Table 8: KPI 7

KPI 8. How often the mark in the progress bar is pressed. This is used to identify how often the user uses the marking feature and from this it can be determined if the feature is worth implementing or not on other websites.

	Original	Adapted	Pagination
Total Events	0	0	0
Sessions with Event	0	0	0
Total Events / Total Sessions	0	0	0

Table 9: KPI 8

KPI 9. How often a marked article is pressed. This is used to determine if the marking feature is used and how often marked articles are clicked comparing to unmarked.

	Original	Adapted	Pagination
Total clicks on a marked article	0	0	0
Total clicks on an unmarked article	482	1252	1282
Total clicks on a marked article / Total Sessions	0	0	0
Total clicks on an unmarked article / Total Sessions	0,0445	0,0812	0,0830

Table 10: KPI 9

KPI 10. Sessions where a marked article is pressed immediately after the mark in the progress bar is clicked. In Table 11 it is possible to see how often the user uses the marking feature as it was intended by the developers.

	Original	Adapted	Pagination
Total Sessions with sequence	0	0	0
Total Sessions with sequence / Total Sessions	0	0	0

Table 11: KPI 10

4.3.2 SURVEY

There were 13 users that answered the survey while around 1000 users was exposed to the adapted infinite-scrolling version. That response rate is too low for drawing any conclusions. An acceptable response rate would be around 25 %.

5 DISCUSSION

This sections contains a discussion of the results, method and the work in a wider context. The results discussion will mainly be about the data from the evaluation. The methods discussion should bring motivations for the chosen methods and improvements that was implemented to the AOI.

There has been a few similar studies in this field, [10], [30]. These studies have compared pagination and infinite-scroll on a blog or a social network where the goal is to keep the users on the page for as long as possible, exposing him/her for as much data and advertisement as possible.

This study have focused on the opposite, how infinite-scroll can be used on an intranet of a healthcare company where the goal is to provide more efficient web browsing. Here, a study has been made comparing three types of websites; one paginated website, one adapted infinite scrolling website and one infinite scrolling website with a load more button.

The focus of this thesis was to make the web browsing as efficient as possible. Therefore, a few improvements were made to the adapted infinite-scrolling website. However, the intention was not to test each improvement individually them to see which one was worth the most, regarding user behaviour and implementation cost. However, the purpose of this thesis was to see if an adapted version of infinite-scroll could perform better than the other versions. This was confirmed with regards to how many articles were loaded per session duration and how many articles were clicked per session duration.

The topic in this study has only been scratched a bit in this report. Recommendations for future work is to investigate what improvements can be made to all three of these pagination techniques and how each improvement performs individually. Also, what techniques are best suited for different kinds of websites. Finally, a study about the users' thoughts regarding these three different pagination techniques could be a useful addition.

5.1 PRE-STUDY

This section provides information and motivations on why certain strategies where chosen in the pre-study phase.

5.1.1 GOOGLE ANALYTICS

Google Analytics is a widely known application for gathering statistics from the web and are by far the biggest tool in this area. It is used by many companies to analyse websites. Using Google Analytics was an easy call because it can gather all the data that was going to be measured and it is easy to implement. When it comes to analysing the data it is possible to apply filters, make sequences, segments and much more. It is also a free to use web tool.

5.1.2 MEASUREMENT PLAN

The measurement plan was developed to specify what to measure. With Google Analytics it easy to get lost in all features and all statistics. By defining exactly what needs to be measured and specifying it in the measurement plan it is easier to gather statistics in its purest form, for example if data should be analysed from all sessions or just the ones containing a certain event.

5.2 IMPLEMENTATION

This section provides motivations for the implementation and contains discussion about which techniques was chosen and why. How they did pane out and what alternative methods there are.

5.2.1 CONTENT LOADING

The chosen solution for loading new content was to load when the user is scrolling near the end. The motivation was mainly based on the goals which was to find specific information faster, and view more information in less time. The other alternatives gives a longer waiting time for the new content to appear for the user. So the assumption was made that the most important thing for the user concerning content loading in this case is to minimize the waiting time for the new content to appear on the screen.

5.2.1.1 AJAX LOADING PERFORMANCE

Ajax is used to update the feed, and to know how many posts should be loaded every request a test was performed. The Ajax loading performance was measured both with wired and wireless internet connection. Based on the average request time a time for loading 10, 50 and 200 posts were calculated. 10 corresponds to show one page and 50 corresponds to show a couple of pages and 200 corresponds to showing a lot of pages and more than 200 is assumed to be unnatural.

The test was also performed on another feed, and the data from these tests shows differences between the two. The other feed will however not be handled in this report. So this data will only serve the purpose to show that differences can occur depending on the content management system. The feed under investigation is using EPiServer as content management system, the other is using a custom implementation of Entity framework.

As seen in Figure 4-Figure 7 the tests shows that the wireless connection was nearly twice as slow as the wired. The Ajax request is a lot faster on the feed using EPiServer compared to the other feed, in the best case 15 times faster. The measured time also differed very much and a lot of spikes were detected when using a wireless connection. In this case a spike means that the Ajax request takes much longer time than usually, for example the average request time is about half a second but in certain cases the request time could be around two seconds. This makes it four times slower, therefore the spikes drastically affects the performance.

To summarize, when using Entity framework, the Ajax request time is nearly the same when requesting 1 post or 100 posts. This is not the case when using EPiServer, because 100 posts takes longer time to request than 1 post.

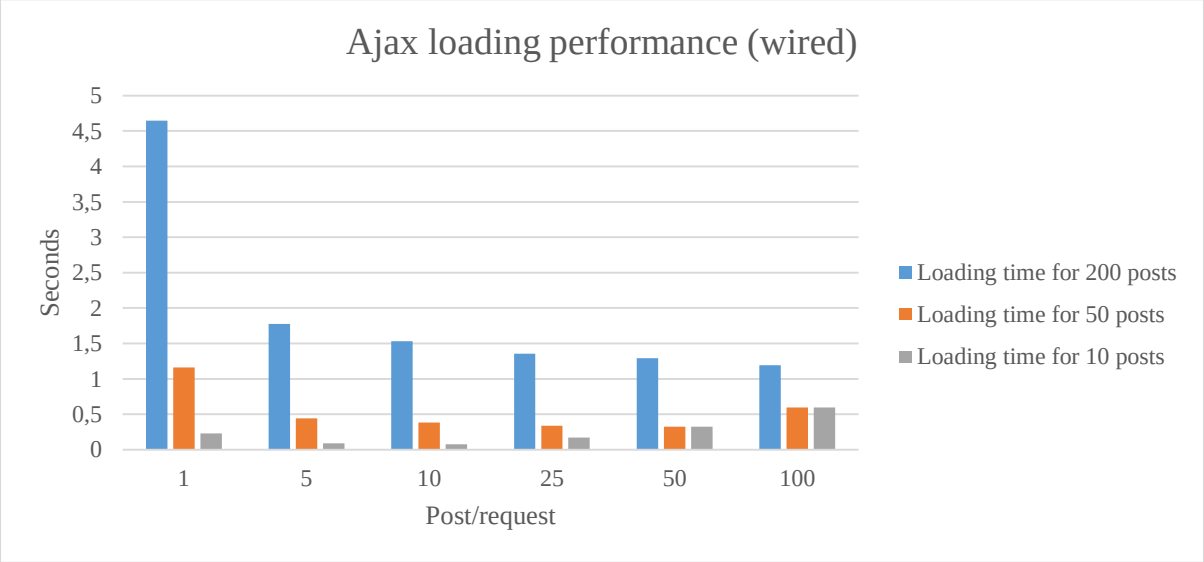

Figure 4: Ajax loading performance for feed using EPiServer (wired)

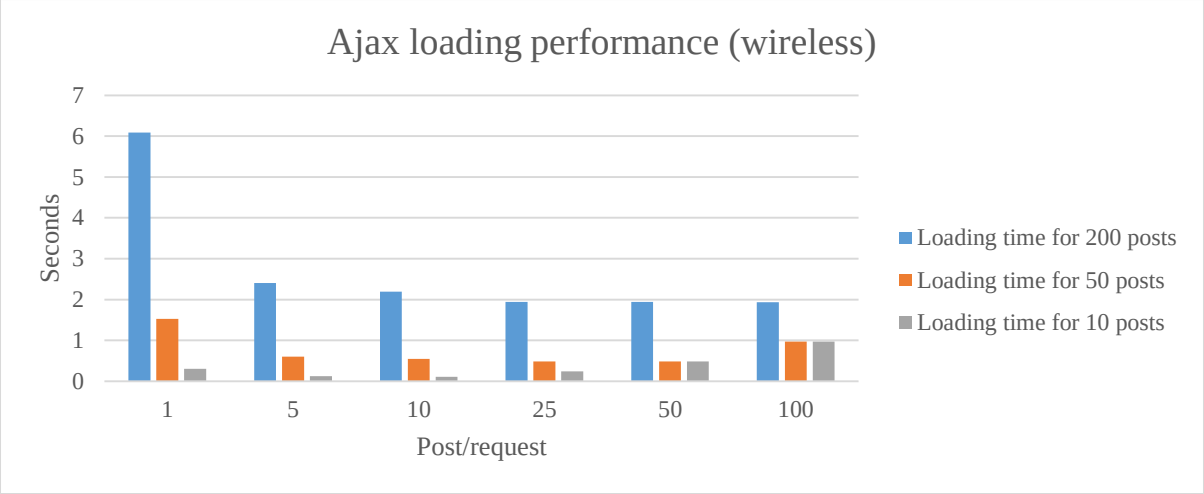

Figure 5: Ajax loading performance for feed using EPiServer (wireless)

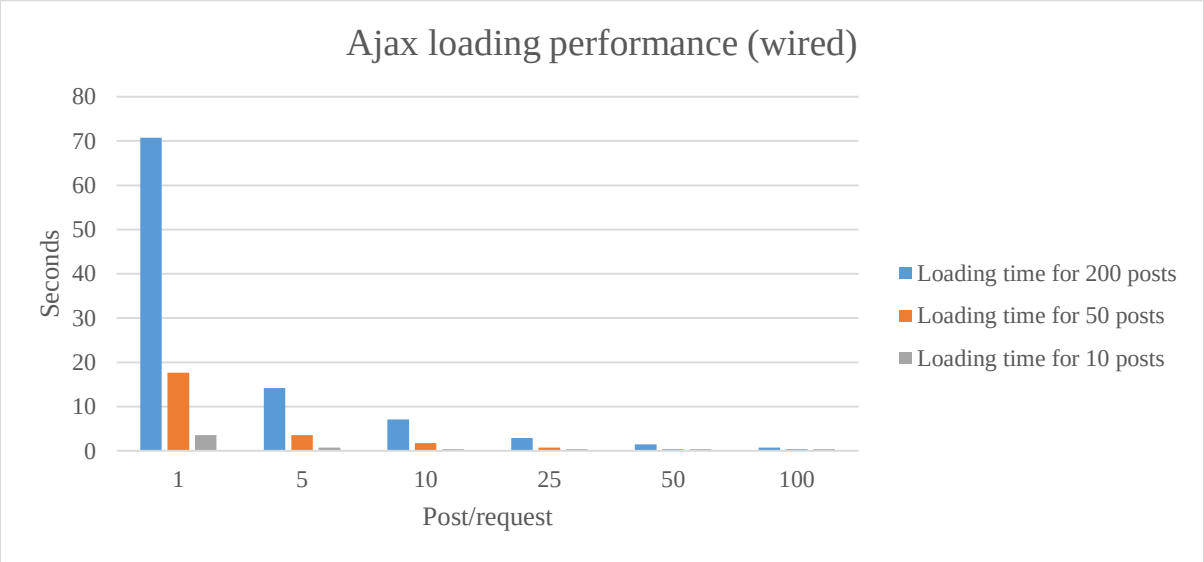

Figure 6: Ajax loading performance for feed using Entity framework (wired)

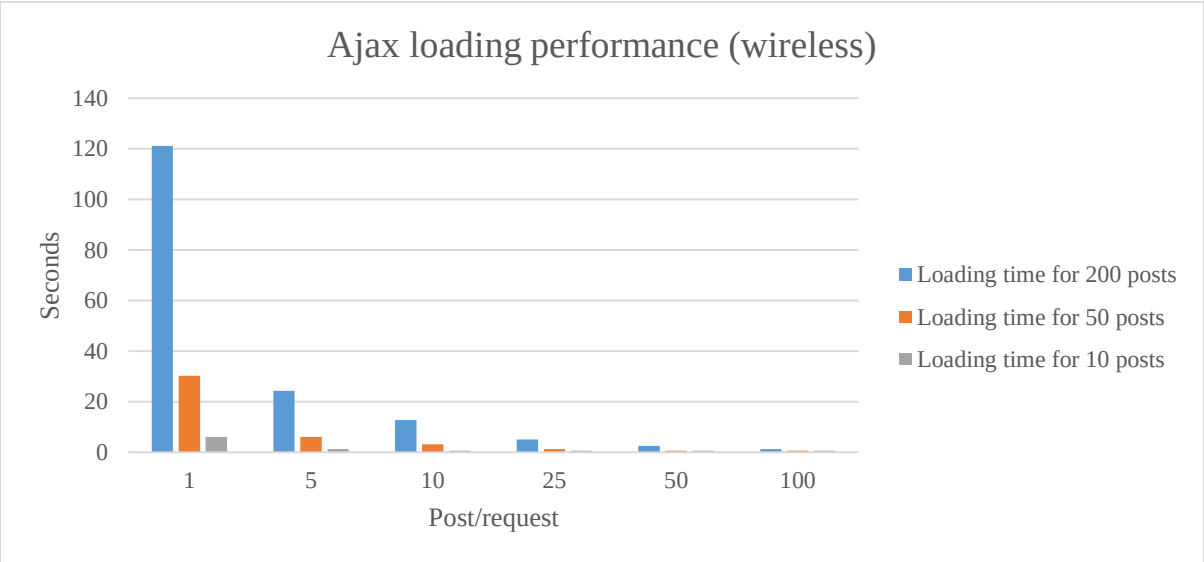

Figure 7: Ajax loading performance for feed using Entity framework (wireless)

The results showed that loading a few posts or many posts every request matters. So different post/request values will affect the performance of the site, so the best value should be selected. One idea of choosing the post/request value is to choose the value that gives the shortest time for the average loaded posts. However no information is available about how many posts is loaded in the average case, so the value cannot be selected from that. So the value will be chosen from reasoning. When loading a lot of articles, the time difference will have a smaller impact since the time spent on the page is longer. Therefore the case when loading a few article posts should be prioritized. From that assumption the value should be chosen that loads 10 posts fastest, and that is 10 post/request, and that selected value only loads 200 post about a half second slower than the fastest way to load 200 posts.

As a note from the result, there could be a case where there is a small difference between loading many and a few posts in the average request time, so regarding, only Ajax performance, loading more posts every request is desirable. This was the case with the other feed using Entity framework, this feed will however not be handled in this work so this is only seen as an observation.

5.2.1.2 PIXELS TO BOTTOM

A value needs to be chosen how near the end new content should be loaded, i.e. many pixels to bottom. This should be chosen so that the user don't need wait for new content to appear. To find a value which minimizes the waiting time for the user, a script was made, which scrolls continuously in a constant scroll speed that simulates a user scrolling. This script also detects if the scrollbar reaches the end of the page, which counts as the user is getting stalled and waiting for new content to be loaded. A test was performed to find the value that doesn't get the user stalled when scrolling, using this script. The script ran with different pixel values, starting from 100 to 250 pixels with 50 pixels between. If the seamless loading is stalled, it counts like a fail, otherwise a success. The test page has 118 posts and the scroll speed was set to 500 pixels/second. Every pixel value is tested 10 times, to prevent inconsistent data. The chosen value is also tested in another browser 10 times, to verify it. This test was performed with wired internet connection on both feeds and the results are seen in Table 12 and Table 13. When using wireless connection, all tests failed up to 600 pixels.

Pixels to bottom	Succeeded	Fails
100	10	0
150	10	0
200	10	0
250	10	0

Table 12: Pixels to bottom, EPiServer

Pixels to bottom	Succeeded	Fails
100	0	10
150	0	10
200	9	1
250	10	0

Table 13: Pixels to bottom, Entity Framework

The results from the test shows that the recommendation of 100-500 pixels [11] works seamless on this specific site using wired internet connection. But when using wireless connection no pixel values succeeded. However, no conclusions can be drawn because all users will have different connection speed and there is no known average scroll speed in that clientele.

5.2.1.3 UNLOADING CONTENT

When determining if unloading content is necessary, information about the user environment is needed. All of the users' computer setup is almost the same, which is conducted from the product owner and confirmed from google analytics. They run Citrix 6.5 with Windows 7 and the majority uses Internet Explorer 11, the other uses chrome. The RAM memory and CPU differs, but taken from windows 7 requirements they must have at least 1GB of RAM and 1 GHz CPU [31].

A test was made that showed that the memory usage increased with 3.5mb when 90 articles where loaded. Which means approximately 0.04mb/article. The latest content available is 4 months old, with this information they

produce 22.5 articles/month, and 270 articles/year. 270 articles gives you a memory usage of 10.8mb. If the assumption is made that no one scrolls through one year of articles, the maximum memory usage is 10.8mb. When having 1 000mb of RAM memory, a conclusion can be drawn that 10.8mb is not much, and will not affect the performance of the site.

The test also showed that the page is using 984 document object model (DOM) elements when 90 articles where loaded. This can be approximated to 11 elements/article. With the information above of 270 articles/year gives a use of 2970 elements/year. It is hard to tell if this is good or bad, some rule of thumb indicates that 10 000 is an absolute max. Though, when the test was performed and 90 articles was showed, no indication was found that the webpage was performing badly.

The conclusion from the test was that the site will not perform badly due to not unloading content in the closest years unless the behaviour and the amount of articles produced or how they are presented changes drastically. This is because the RAM memory and DOM element usage is low and the site consists of text mainly, with 150x150 pixel images.

5.2.2 Bookmarking and making the back-button work

Being able to bookmark a specific place in the feed solves three problems with the nature of infinite-scroll. The first, and most enlighten is when clicking on an article and then pressing the back-button will make the previous position to get lost and the user will have to start over from the top of the feed. By solving this, in the way described in the method section, automatically makes two other features available to infinite-scroll which previously only have been able on paginated websites. These are, being able to bookmark a specific section of a feed and being able to send a link to a specific place of the feed and not only the start.

Solving this can be done in various ways. The solution that was chosen was using the HTML5 history API. This means that the URL is updated every time an article is scrolled past. Another solution that would work almost like this, but on elder versions than HTML5 is updating the hash on the site. The problem with the hash implementation is that the information after the hash is never sent to the server, which means that all the other content has to be loaded from JavaScript.

This solution was chosen because it solves all of the three problems defined earlier, the back-button issue and the bookmarking and linking issue. The reason why the HTML5 approach was chosen was because the loading time for the page would decrease by sending all the data from the back-end at once instead of sending a little bit of the data as usual and then using Ajax for getting the rest of it.

In this solution the URL is updated with three extra parameters, these are called *content*, *before* and *after*. The *content* variable determines which article are at the top of the visible area and the other two determines how much content there is before and after said article. This is saved because when using the back-button the user will return to a view that looks like the one the user left with the same articles in the same places. A constant variable is used to determine the maximum posts that can be loaded to prevent malicious attempts by increasing the parameters in the URL. Also by using these parameters, it is easy to discover and notify the user if new data can be loaded by scrolling to the top.

However, this means that the URL changes every time the user scrolls past an article. This could potentially be a distraction for the user. This does also mean that the user could by mistake change the URL and therefore should the implementation take care of such changes.

Also, if the user scrolls past an article and decides to bookmark the page. The page will be bookmarked in the current section. This could be both positive and negative, depending on the user. But a potential problem is if the user is not aware of the feature and always gets to the middle of a feed and not knowing why.

5.2.3 Footer

The footer becomes unavailable with the solution to load new content automatically. It exist some hate over teasing footers that are unavailable [32], it is also quite easy to understand that a footer that is unavailable is

unnecessary. With this in mind the traditional footer should be removed. In this case the information is important and should be provided. The easiest solution is to provide the information somewhere else, as another page reached from the navigation bar for example. This is often a suitable solution since it does not break anything [33]. Another alternative is to use a sticky footer or move it to the sidebars [34], and Lookbook.nu provides an example of such an implementation [35]. A similar solution to this was created, the original footer is appended to the right sidebar at the bottom of the page. So the footer appears to the right of the feed when the user is scrolling. Though, the owner of the site had a strong opinion that a sticky footer with the same dimensions was the best solution, so that was created instead of the footer to the right.

5.2.4 An indicator of where the user is in the feed and where it ends

Some problems with the scrollbar when implementing infinite-scroll have been identified by several sources [8], [9], [19]. Each source claims that the user is tricked by the scrollbar because it always seems like there is just a little more info left on the page. Another claim is that users get exhausted and lost in the page when they have no idea of where they are in the feed.

When choosing an implementation to address this problem, a progress bar felt like an obvious choice for determining how much is read and how much is left. Since users tend to look to the right for information of where they are on the site (using a working scrollbar) the progress bar was placed next to the scrollbar. In that way the progress bar does not attract any extra attention from the user when browsing. But when the user looks at the scrollbar for information, the progress bar is there and provides the info the user needs.

5.2.5 Functionality to remember previously seen content that can be worth seeing again

The chosen solution was to have a reference point to the previously seen content and the ability to mark articles to create these reference points. That is a combination of the two suggested solutions, the pin feature will create a reference point. Opposite to Pinterest where these pinned post will appear on a new view. This did not feel like a desirable feature in this case, since opening an article in a new tab will achieve this and will not add much to the site.

The trick with the solution is how these reference points should appear. The first decision was made to avoid numbers and text, based on that reading is unnatural [21]. So using page numbers as discourse does, was precluded. A solution was created with shapes instead. The first solution was created with the scrollbar in mind, so small blocks was created next to the scrollbar indicating where the pinned article is located according to the scrollbar. So if the scrollbar was moved to that position the article will appear in the top of the screen, if the user clicked the block JavaScript will scroll the user to that position. However, when a progress bar was created to serve a purpose of an indicator where the user are and where the feed ends the two solutions did not collaborate well. So the reference points was adapted to fit the progress bar instead, so the blocks are appended to the progress bar corresponding how much progress is made to see the marked article. The functionality to be able to click on a reference point still takes the user to the pinned article. This may however not be as intuitive as using the scrollbar for reference, but the assumption is made that the feature will be easy to understand and adopt when used a couple of times. Which can be a good assumption according to Jeff Johnson's guidelines [21].

5.3 Evaluation

The evaluation was performed as a quantitative study using A/B testing and Google Analytics to collect data. All of the four thousand users participated in the evaluation, so the data should reflect every user in the system, compared to doing interviews and user test with a small group of the users. Because of that the result should reliable. Problem can though occur if no data is collected, if no user is using a certain feature for example. Then no conclusion can be made about it. Another problem can be that user's opinion is ignored. For example, a feature can perform well according to the user behaviour, even if the user hate the feature. To avoid this problem, a survey was used to collect the users' opinion.

5.4 RESULTS

This section contains a discussion about the results, about what was expected from the pre-study, and what was not and what can be the reason in different cases.

KPI 1. Amount of extra articles loaded per session / session duration. One expected user behaviour for infinite-scroll is that the user is more passive, not clicking on articles and also staying longer on the site [7], [10]. An assumption can be that the user scrolls a lot and therefore is exposed to a lot of content, the amount of extra articles loaded per session is expected to be higher.

As seen in Table 2 it is obvious that the adapted version loads more articles than the other versions, as expected. With about ten times more than the original and about five times more than the paginated version per session.

The adapted version also has a lot more sessions with events than the other ones. But something to note is that, when only counting the sessions that has at least one event, the original version shows more articles in the same time span. This means that users generally are more exposed to articles and view more with the adapted version. So the chance of an article being clicked is greater, because an article that is not shown, cannot be clicked.

Conclusions can be drawn that the adapted version encourages users to get a quick overview of what has happened by showing more articles than the other versions. This KPI proves that for the general user, the adapted version shows more articles per second than the other ones. This answers the first research question.

KPI 2. Session duration. Previous studies shows that users tend to stay longer on the page when using infinite-scroll [10]. For that reason the expected result is that session duration should be lower for the paginated version. The desirable scenario at this site is to work more effectively, to find more information in the same time span. This means that a long session duration is not necessarily good on this site, opposite to many other sites.

The result from Table 3 shows similar session duration for the adapted version and the paginated version, which is a lot longer than for the original version. The case can be that these two new versions feels exciting and users spends more time to explore them. Another reflection can be that the users feel more comfortable using the two new versions and therefore wants to spend more time on them.

KPI 3. Number of articles pressed per session. The expected result is that users clicks less when using infinite-scroll, since previous studies have showed just that. Mainly because an automatic loading of articles attracts the user to scroll more and not click on articles [7]. The desirable case is not to lose article-clicks on a certain solution.

The adapted and paginated versions has nearly the same numbers as seen in Table 4. Based on these numbers the myth is broken that infinite-scroll leads to fewer clicks compared to pagination, however the original version has about half the clicks on articles as the other two. You can be speculating as above that the two newer versions are more exciting and users tends to click and explore the site more on those versions. This means that no clear conclusion can be drawn from this either, though some good indication that infinite-scroll doesn't leads to fewer clicks on articles on this site.

KPI 4. Number of Back-button clicks. The idea with this metric is to see if the users tends to use the back-button more if it works as intended on the adapted versus the original version. No big differences by the two versions were found. Pagination generates more back-button clicks which is expected, to go back a page the user hits the back-button and an event is sent, when going back a page on the other versions the user scrolls up on the page and no event is sent.

KPI 5. Sessions where a user exits the page immediately after a back-button click. This KPI detects if the user uses the back button and then exits the site immediately. This KPI can indicate if the user is satisfied with the place on the site he/she is returned to by using the back-button and can easily continue browsing which contributes to the first research question.

When looking at Table 6 it is clear that the original version has far more occurrences of this event than the other versions. This indicates that the users are unhappy with the place on the site they are returned to. As said earlier

in the report, the users can feel disoriented when returning to the top of the feed instead of returning to the point where they left off. By looking at Table 6 it is clear that this part has great potential for improvement. By improving this part it may be possible to get users to continue browsing instead of being returned to the start of the feed and having to relocate themselves again

KPI 6. Bounce Rate. By looking at the bounce rate (as seen in Table 7) for the different versions it is shown that the rate for the original version are more than five times higher than the other ones. Why the results appears like this is unclear. A theory could be that the original page is as it always have been and the users only go to the page to see if some new articles have appeared on the website. If not, they just close the page without interacting with the website and then a bit later repeats the same behavior. But when a user comes to one of the other versions they have a new look and feel and the website may somehow intrigue the user to interact with the page.

KPI 7, KPI 8, KPI 9, KPI 10. The results (as seen in Table 8-Table 11) show that the star button was clicked three times, which is counted as zero when compared to the total number of sessions. The results also showed that no user clicked a mark in the progress bar, and therefore no one clicked on an article directly after that.

This means that no one used this new function on the site. The new functionality was not explained in advance, so the solution was not so intuitive that the users knew what to do directly. The functionality cannot be implemented based on this result, and should be investigated more thoroughly to see if it is useful for the user. If this functionality have been used more it is possible that they would have contributed to the second research question.

5.4.1 Survey

The response rate was as low as 1.3 %, because of that the results cannot be used to draw any conclusions from. The results would not reflect all the users' opinion, therefore the result is unreliable. The survey was reached through an article in the news feed. So it was the user's responsibility to answer the survey and they received no reward after the survey. The users is also primarily using the intranet at work, which means that they need to use working time to answer the survey. This could maybe explain the low response rate. One alternative to get more answers for the survey could be to send the survey by email to the users. That alternative is though in practice since the survey only is for those who tried the adapted infinite-scroll version and that email would have to go to all users.

The intention with the survey was to get the users' opinions about the new features. Lacking this information means that no rating between the features can be done. Also no connection could be drawn to how these features collaborates with each other. However, the statistics from Google Analytics showed that some of the new features were unused. Presumably the users would have answered that they can't take a stand, since they didn't try them. Though it would have been interesting to see the users' opinions about the automatic loading of content, since it performed well according to Google Analytics. If the users are unsatisfied with it, the decision could be to not implement it even if it performs well in Google Analytics. The data from the survey is more of this interesting fashion and missing this information would not impact the conclusion of the report.

5.4.2 Summary

The data from Google Analytics showed differences about user behaviour between the different versions. Even though most of the new functionality implemented to the adapted version was unused by the users and should be investigated in more detail to find out if they have some effect on the user behaviour. The two additions that had impact on the user behaviour were the automatic loading of content and making the back button work. The impact was that the automatic loading exposes the user to more content and loading more content in the same time span than pagination and the original version. The most significant difference was that the users didn't exit the page immediately after using the back button to the same extent on the adapted infinite-scroll unlike the other versions. Another main disclosure was that the users didn't tend to click less on articles, when comparing the adapted version of the infinite-scroll with pagination, opposed to previous studies.

6 CONCLUSIONS

The conclusions of this study is that the users of the adapted infinite-scroll version were more exposed to the articles than the users of the other versions. It has also been shown that more articles were loaded in the same timespan with the adapted version which answers the first research question. This may be because of the automatic loading of articles encourages users to get a good overview fast and this, in combination with the back-button function that made it easier for users to continue browsing because they were returned to the same place in the feed where they left off. Regarding the second research question, the results showed that the adapted and the paginated version performs very similar regarding to how many articles were clicked compared to the session duration. This is probably because the improvements in this area were rarely used by the subjects in this study. In future studies it is interesting to see if this changes when the users knows about these features. The recommendation for an intranet site is to at least implement the back-button functionality together with the automatic loading.

This thesis shows that infinite-scroll can be valuable on a goal-driven site. The infinite-scroll and the paginated version performed similarly in this study. The only area where differences were seen between these two were the amount of articles loaded per session. The general recommendation is to use infinite-scroll, and implement support to retrieve the user's position when returning to the feed.

REFERENCES

[1] R. Baker, “Paging vs. Scrolling: Looking for the Best Way to Present Search Results | Software Usability Research Lab,” 2002. [Online]. Available: http://usabilitynews.org/paging-vs-scrolling-looking-for-the-best-way-to-present-search-results/. [Accessed: 18-May-2015].

[2] A. DiSilvestro, “The Pros and Cons of Infinite Scrolling | Search Engine Journal.” [Online]. Available: http://www.searchenginejournal.com/infinite-scrolling-content-pros-cons/113604/. [Accessed: 18-May-2015].

[3] C. Cousins, “Infinite Scrolling: Pros and Cons | Design Shack,” 2015. [Online]. Available: http://designshack.net/articles/layouts/infinite-scrolling-pros-and-cons/. [Accessed: 18-May-2015].

[4] Google, “Google - Infinite scroll trend.” [Online]. Available: http://www.google.com/trends/explore#q=Infinite%20Scroll. [Accessed: 05-May-2015].

[5] Valtech, “Where Experiences are Engineered - Valtech.” [Online]. Available: https://www.valtech.com/. [Accessed: 18-May-2015].

[6] Valtech AB, “Din digitala partner - Valtech.” [Online]. Available: https://valtech.se/. [Accessed: 18-May-2015].

[7] N. Dan, “Why did infinite scroll fail at Etsy?” [Online]. Available: http://danwin.com/2013/01/infinite-scroll-fail-etsy/. [Accessed: 05-May-2015].

[8] L. Hoa, “Infinite Scrolling is Not for Every Website,” 02-Feb-2014. [Online]. Available: http://www.nngroup.com/articles/infinite-scrolling/. [Accessed: 05-May-2015].

[9] A. Yogev, “Infinite Scrolling: Let’s Get To The Bottom Of This - Smashing Magazine.” [Online]. Available: http://www.smashingmagazine.com/2013/05/03/infinite-scrolling-lets-get-to-the-bottom-of-this/. [Accessed: 05-May-2015].

[10] S. Y. Zhang and L. Liu, “Attention Trade-off between Two Types of User Contributions: Effects of Pinterest-Style Infinite Scroll Layouts on Creating Original Sharing and Appreciating Others’ Sharing,” 2013.

[11] infinite-scroll, “Infinite Scroll | jQuery plugin, Wordpress plugin, interaction design pattern.” [Online]. Available: http://www.infinite-scroll.com/. [Accessed: 19-May-2015].

[12] Pathfinder, “Infinite scroller (smarter),” 2015. [Online]. Available: http://georgep.pathfinder.gr/load_more/. [Accessed: 28-May-2015].

[13] Janpaepke, “Infinite Scrolling - Examples - ScrollMagic,” 2015. [Online]. Available: http://janpaepke.github.io/ScrollMagic/examples/advanced/infinite_scrolling.html. [Accessed: 28-May-2015].

[14] T. Bhanse, “LinkedIn for iPad: 5 techniques for smooth infinite scrolling in HTML5 | LinkedIn Engineering,” 2012. [Online]. Available: http://engineering.linkedin.com/linkedin-ipad-5-techniques-smooth-infinite-scrolling-html5. [Accessed: 19-May-2015].

[15] Discourse, “Try Discourse,” 2015. [Online]. Available: http://try.discourse.org/. [Accessed: 28-May-2015].

[16] Appspot, “Items,” 2015. [Online]. Available: http://scrollsample.appspot.com/items.

[17] Quora, “Has the problem with infinite scroll and bookmarking been solved yet?” [Online]. Available: http://www.quora.com/Has-the-problem-with-infinite-scroll-and-bookmarking-been-solved-yet.

[18] C. Cliff, “sausage.js,” 2015. [Online]. Available: https://christophercliff.com/sausage/. [Accessed: 28-May-2015].

[19] K. Schachinger, “Infinite Scroll: Is It Right For Your Site?” [Online]. Available: http://marketingland.com/infinite-scroll-125183. [Accessed: 27-May-2015].

[20] N. Bevan, “Usability is quality of use,” *Adv. Hum. Factors Ergon.*, vol. 20, pp. 349–349, 1995.

[21] J. Johnson, *Designing with the mind in mind: Simple guide to understanding user interface design rules.* Elsevier, 2010.

[22] S. Sukamolson, “Fundamentals of quantitative research,” *Retrieved On*, vol. 25, 2012.

[23] P. Chopra, “The Ultimate Guide To A/B Testing – Smashing Magazine.” [Online]. Available: http://www.smashingmagazine.com/2010/06/24/the-ultimate-guide-to-a-b-testing/. [Accessed: 07-May-2015].

[24] Google, “Features – Google Analytics.” [Online]. Available: http://www.google.com/analytics/features/. [Accessed: 07-May-2015].

[25] Google, “Google Analytics Feature List – Google Analytics.” [Online]. Available: http://www.google.com/intl/en_uk/analytics/features/index.html. [Accessed: 26-Aug-2015].

[26] L. Hasan, A. Morris, and S. Probets, “Using Google Analytics to evaluate the usability of e-commerce sites,” in *Human centered design*, Springer, 2009, pp. 697–706.

[27] K. Avinash, “Digital Marketing and Measurement Model: Web Analytics,” 2015. [Online]. Available: http://www.kaushik.net/avinash/digital-marketing-and-measurement-model/. [Accessed: 26-Jun-2015].

[28] “How a session is defined in Analytics - Analytics Help.” [Online]. Available: https://support.google.com/analytics/answer/2731565?hl=en. [Accessed: 13-Aug-2015].

[29] “Bounce Rate - Analytics Help.” [Online]. Available: https://support.google.com/analytics/answer/1009409?hl=en. [Accessed: 13-Aug-2015].

[30] J. Campbell, A. Chin, and A. Lee, “The Effects of Pagination and Infinite Scrolling on Leisure Browsing.” [Online]. Available: http://www.julieanncampbell.com/paginationvsscrolling.pdf. [Accessed: 04-May-2015].

[31] Microsoft, “Windows 7 system requirements - Windows Help.” [Online]. Available: http://windows.microsoft.com/en-us/windows7/products/system-requirements. [Accessed: 24-Jun-2015].

[32] M. Wilason, “The ‘Death of the Footer’ – A Case of Infinite Scrolling - UX Motel,” 21-Aug-2013. [Online]. Available: http://ux.walkme.com/the-death-of-the-footer-a-case-of-infinite-scrolling/. [Accessed: 22-Jul-2015].

[33] M. Ciarrocchi, “website design - How can the classic footer elements be integrated into an infinite scrolling layout? - User Experience Stack Exchange,” 05-Aug-2012. [Online]. Available: http://ux.stackexchange.com/questions/12234/how-can-the-classic-footer-elements-be-integrated-into-an-infinite-scrolling-lay/12235#12235. [Accessed: 22-Jul-2015].

[34] A. Smith, “Reach A Definite Logical End Using Infinite Scrolling - Search Engine Journal.” [Online]. Available: http://www.searchenginejournal.com/reach-a-definite-logical-end-using-infinite-scrolling/64008/. [Accessed: 18-May-2015].

[35] Lookbook, “Hot Looks, by Hype + Newness | LOOKBOOK.” [Online]. Available: http://lookbook.nu/. [Accessed: 21-Jul-2015].

GLOSSARY

SPA	Single-page application, a web application that can be fitted into a single web page and the page itself is not reloaded at any time
Intranet	Private website only accessed by the company staff
DOM	Document Object Model Example of DOM element is DIV, HTML, BODY element on a page.
AOI	Areas Of Improvement: • Loading and unloading content • Bookmarking and make the back-button work • An indicator of where the user is in the feed and where it ends • Functionality to remember previously seen content that can be worth seeing again • Functionality to remember where you left off last time (optional) • Skipping results (Optional)
Quantitative study	Explaining phenomena by collecting quantitative data, which are analysed by mathematically based methods.
Footer	Bottom section of a webpage, which can contain information about the webpage. For example copyright and contact information.
Infinite-scroll	A technique that dynamically loads new content on to the page when the user is scrolling through the page.
Experiment	Googles implementation and definition of A/B test
KPI	Key Performance Indicator
EPiServer	System to handle and publish web content, in other words content management system.
Entity framework	Technique for converting data between incompatible type systems in object-oriented programming languages, in other words object-relational mapping
Ajax	JavaScript can make request to the web server without refreshing the page