Task-based usability testing and performance measurement over the internet

Stefan Eng

DIVISION OF ERGONOMICS AND AEROSOL TECHNOLOGY | DEPARTMENT OF DESIGN SCIENCES
FACULTY OF ENGINEERING LTH | LUND UNIVERSITY
2020

MASTER THESIS

MASSIVE ENTERTAINMENT | A **UBISOFT** STUDIO

Task-based usability testing and performance measurements over the internet

Stefan Eng

LUND
UNIVERSITY

Task-based usability testing and performance measurements over the internet

Published by

Department of Design Sciences
Faculty of Engineering LTH, Lund University
P.O. Box 118, SE-221 00 Lund, Sweden

Subject: (MAMM10), Interaction Design
Division: Ergonomics and Aerosol Technology
Supervisor: Günter Alce
Company supervisor and company contact: Patrick O'Casey
Examiner: Johanna Persson

Abstract

User-centered design and usability engineering are two concept that both roughly date back to to the 1990s, and both of which have had a large impact on the practices of software-engineering and the field of software-design. How much effort is required to take these concepts and apply them, without much prior experience, to todays interconnected and geographically distributed working environment?

Not much actually, by leveraging Python and Flask, this report demonstrates that it is possible for a software-developer with little prior knowledge, to build and deploy the functional foundation of an web-based usability-testing-platform, accessible through any browser, that allowed 81 participants to run a total of 698 task-based test-scenarios over the span of five days.

Additionally, by conducting some initial interviews about current usability issues. Using this information as the basis for the aforementioned test-scenarios, you now have the building blocks for a online-accessible usability-testing-platform providing insight into real-world usability problems, ready for use in any modern software industry, such as game development.

To Amanda, my lovely fiancé. Thank you for putting up with me dragging my feet behind me for way to long and keeping the hope up ♥

Acknowledgments

I would like to extend a big thank you to my supervisor at the institution, Günter Alce together with my company contact and supervisor at Massive Entertainment, Patric O'Casey. Both of you have provided me with guidance when it has been needed, and you have both shown tremendous patience with me on the occasions I have been out deep in the weeds, thank you.

Malmö, June, 2020

Stefan Eng

Contents

Chapter 1

Introduction

It has been established that the output of a team benefits from information sharing[1, 2] , and that such vital sharing tends to break down when it is needed the most[1].

This effect is something that has been observed first hand by one of the managers at Massive. Working with a team under pressure, nearing a deadline, the software used for planing and information distribution was abandoned, in favor for post-it-notes on a whiteboard. While acknowledging that it is a method that works for smaller teams, the manager observed that it put a big damper on the bandwidth available for information sharing among the team-members.

Asking some of the employees why they opted to not use the software in the scenario mentioned above, they answered that they felt it got in their way, hindering them from doing their work, a clear usability issue.

1.1 MASSIVE
MASSIVE ENTERTAINMENT | A UBISOFT STUDIO

Massive Entertainment is a world-leading game-development-studio, founded by Martin Walfisz in 1997 and located in Malmö, Sweden.

Since their first release, Ground Control in June of 2000, the company has produced a string of critically acclaimed games, and has been part of both Vivendi Universal Games and Activation Blizzard. As of 2008 they are a permanent part of Ubisoft, a video-games company with its headquarter based in Montreuil, France with associated studios all over the world.

After becoming a part of the Ubisoft family, Massive managed to break Ubisofts record for most copies sold in 24 hours with the release of Tom Clancy's The Division on the 8'th of March 2016, which then set another record for having the biggest first week ever for a new

game franchise. At the time of writing Massive has more than 650 employees working at their Malmö studio and has announced that their next big project is related to James Cameron's Avatar.

1.2 The Swedish game-development sector

Since 2006, Dataspelsbranchen, a Swedish trade association for video game companies[3], has released a yearly report called 'Spelutvecklarindex' where they gather and publish information related to the growth of Swedish game development companies. According to their most resent publication, 'Spelutvecklarindex 2019'[4], the domestic Swedish game-development sector continues to grow steadily.

In 2018, according to the report, the total revenue of the sector grew with 33%, totalling 1.87 billion EUR, with the number jobs provided also increasing with 14%, totalling 5 320[4, p. 12] full-time positions, at 384[4, p. 38] active game-development companies in Sweden.

With its 650-plus employees, Massive enters the statistics as the 4'th largest game-development studio in Sweden[4, p. 21], making them a big potential influence on how the sectors develops in the future, both in terms of their employees as well as the players of their games.

1.3 Global game-development and usability

According to the Entertainment Software Association, or ESA, witch is the trade association for the video game industry in the United States the game-development is on a steady rise there to. In their latest report, titled '2019 Essential Facts'[5] they write that 65% of American adults play video games[5, p. 4] , and information at their official homepage states that from 2017 to 2018 the industry grew with 18%, reaching a record-high $43.4 billion dollars in revenue[6].

As digital sales reaches 83% of the total purchase volume in America[5, p. 20] game-development is becoming even more global and distributed. This trend affects not only the sale of the finished games, but also the development of said games, with 31 Swedish studious employing a total of 2 604 persons abroad[4, p. 27].

In this type of large, competitive and global market, the author thinks that usability could provide an interesting edge.

In 'Usability and the bottom line'[7] the author Donahue argues that, while few people disagree with the notion that usability engineering benefits the end user, it is harder to convince people of the benefits for companies and its employees.

And even though the piece does not focus on game-development directly, the general game-development procedures overlaps all the examples mentioned; IT system (backend-systems and servers for games), e-commerce (the digital sale of the game to consumers) and shrink-wrapped software (the actual distribution of the game binaries).

This makes game-development an excellent candidate benefiting from almost all of the listed benefits of incorporating large-scale usability in a company, but most fittingly;

- Reduced development and maintenance
- Lower support cost
- Improved productivity and efficiency
- Reduced training costs
- Reduced documentation costs

It is also worth mentioning a specific quirk that gives the gaming-industry a slight upper hand compared to the rest of the software industry in regards to usability-testing. Even though it might be used slightly incorrectly[8] in comparison to its often cited source[9], the usage of 'thinking aloud protocol' where "subjects verbalize their thoughts at the time that they [surface]"[9, p. 60] is widely used, prompting Nielsen to state "thinking aloud may be the single most valuable usability engineering method"[10, p. 195]

Thanks to the current trend of streaming games to a invisible public, a company with a popular game being streamed will have high-definition videos of players playing their game, while providing commentary, available, for free. And while not providing the same insight and fidelity as a correctly performed in-person thinking-aloud-session, this material could be used to introduce the concept of thinking-aloud to a company, or as training material for new usability engineers[11].

1.4 Goals

1. Create a web based platform for basic interface usability testing.
2. The platform should be able to introduce and administer a group of potentially geographically distributed individuals to task-based usability-testing.
3. Develop the platform by collecting feedback and user-statistics and using the data to inform where and how to make changes in order to improve the platform.

Chapter 2

Background theory

Though there exists many definitions of the term usability, the author finds the following one from the ISO Standard 9241-11:2018[12], section 3.1 - Terms and definitions to be one of the more concise and direct ones.

> **usability**
> extent to which a system, product or service can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use

In other words, for something to be usable, it has to help its user to reach a goal, not only in an effective and efficient way, but it should also be a satisfactory process to get there.

2.1 User-centered design

In 1986 Donald A. Norman contributed to the book *User centered system design : new perspectives on human-computer interaction*[13] with, among other things, a chapter titled Cognitive Engineering[13, p. 31]. Said chapter begins with the following prologue, quoted in part below:

> *Cognitive Engineering*, a term invented to reflect the enterprise I find myself engaged in: neither Cognitive Psychology, nor Cognitive Science, nor Human Factors. It is a type of applied Cognitive Science, trying to apply what is known from science to the design and construction of machines …there are fundamental difficulties in understanding and using most complex devices. So the goal of Cognitive Engineering is to come to understand the issues, to show how to make better choices when they exist, and to show what the tradeoffs are …

He then continues[13, p. 32]:

> …Overall, I have two major goals:
>
> 1. To understand the fundamental principles behind human action and performance that are relevant for the development of engineering principles of design.
>
> 2. To devise systems that are pleasant to use–the goal is neither efficiency nor ease nor power, although these are all to be desired, but rather systems that are pleasant, even fun …

Making note of the second goal, which could just as well have been an expansion of the satisfactory aspect in the above mentioned usability definition. The text then continues with *Analysis of Task Complexity*, *The Gulfs of Execution and Evaluation* and *Stages of User Activities*, to name a few topics, and ends with a section titled *Prescriptions for Design Principles*.

This last section expands on what purposed functions and accomplishments should come as an result of developing Cognitive Engineering further. Considering the time the text was written, due to the lack of available information on what fosters good interactions between people and machines, there are no guidance given for the specific details of the design. There are however, guiding prescriptions for how the design process might proceed[13, p. 59-61], each with its own heading and describing paragraph, headings replicated below:

- Create a science of user-centered design
- Take interface design seriously as an independent and important problem
- Separate the design of the interface from the design of the system
- Do user-centered system design: Start with the needs of the user

It is the last heading and accompanying paragraph that is of special interest in this context since it contains the following definition of what user-centered design should strive for[13, p. 59-61]:

> user-centered design emphasizes that the purpose of the system is to serve the user, not to use a specific technology, not to be an elegant piece of programming. The needs of the users should dominate the design of the inter- face, and the needs of the interface should dominate the design of the rest of the system.

A more recent touch-stone for user-centric design comes from the 2003 paper titled 'Key principles for user-centred systems design'[14]. Here, the authors argue that user-centered system design (UCSD) lacks an agreed upon definition, and that the concept as a whole suffers for it. Building on earlier defined concepts from literature combined with their own research and software development experience, they present the following definition of UCSD[14, p. 401]:

> User-centred system design (UCSD) is a process focusing on usability throughout the entire development process and further throughout the system life cycle. …based on the following [12] key principles:

Headings of the key principles[14] are represented below:

- *User focus – the goals of the activity, the work domain or context of use, the users' goals, tasks and needs should early guide the development*
- *Active user involvement – representative users should actively participate, early and continuously throughout the entire development process and throughout the system lifecycle*
- *Evolutionary systems development – they systems development should be both iterative and incremental*
- *Simple design representations – the design must be represented in such ways that it can be easily understood by users and all other stakeholders*
- *Prototyping – early and continuously, prototypes should be used to visualize and evaluate ideas and design solutions in cooperation with the end users*
- *Evaluate use in context – baselined usability goals and design criteria should control the development*
- *Explicit and conscious design activities – the development process should contain dedicated design activities*
- *A professional attitude – the development process should be performed by effective multidisciplinary teams*
- *Usability champion – usability expert should be involved early and continuously throughout the development lifecycle*
- *Holistic design – all aspects that influence the future use situation should be developed in parallel*
- *Process customization – the UCSD process must be specified, adapted and/or implemented locally in each organization*
- *A user-centered attitude should always be established*

2.2 Usability testing

In 1993, Jacob Nielsen wrote a book titled *Usability engineering*[10], where he, among other things, builds on and clarifies previous work surrounding usability testing. In the preface he states the goal of the book as follows:

> The main goal of the book is to provide concrete advice and methods that can be systematically employed to ensure a high degree of usability in the final user interface. To arrive at the perfect user interface, one also needs genius, a stroke of inspiration, and plain old luck. Even the most gifted designers, however, would be pressing their luck too far if they were to ignore systematic usability engineering methods.

This quote, from the introduction to chapter 6, *Usability Testing*[10, p. 165] does a good job exemplifying one of the core tenants underpinning both usability testing and user centered design.

> User testing with real users is the most fundamental usability method and is in some sense irreplaceable, since it provides direct information about how people use computers and what their exact problems are with the concrete interface being tested.

Continuing the chapter, Nielsen introduces the following concepts, probably familiar to anyone working with usability testing: *Test Gaols and Test Plans*, *Getting Test Users*, *Choosing Experiments*, *Test Tasks*, *Stages of a Test*, *Performance Measurement*, *Thinking Aloud* and *Usability Laboratories*. Looking at more resent literature related to usability testing, it is not hard to identify concepts that are, if not identical, at least similar to the ones mentioned above.

In the 2008 edition of *Handbook of usability testing: how to plan, design and conduct effective tests*[15, p. 19], the authors talk about usability testing being roughly split into two approaches. The first being the more formal one, where tests are conducted as true experiments, with the goal to confirm or refute one or more specific hypothesises. The second testing approach, which book focuses on, is characterized by an iterative cycle of tests and is somewhat less formal, but still rigorous. Core to this type of testing is the iterative testing cycle, intended to gradually expose shortcomings in the design of the system under test and continuously re-shape it based on collected data and evaluations from the performed tests.

It is this second iterative type of usability testing that is the basis for any usability testing conducted as part of this report. Also, if not specified otherwise, any further mentions of *usability testing* refer to this type of iterative approach.

Chapter 3

Development process and technology stack

The development process outline was inspired by the 12 key-principles of user-centered system design listed earlier, especially the following ones: *user focus*, *active user involvement*, *evolutionary systems development*, *simple design representations*, and *prototyping*.

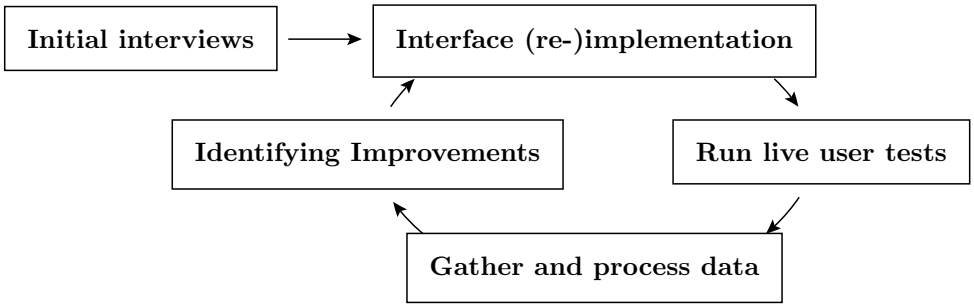


Figure 3.1: Concept, development, testing and improvement cycle.

3.1 Brainstorming-sessions and interviews

In order to figure out the initial development approach and goal, two brainstorming-sessions were conducted.

The first session was conducted with members of the team at Massive, trying to figure out something that could en up being beneficial to the team and possibly the studio at large. The reasoning was that by focusing on something with a tangential benefit clearly defined from the start, the buy-in from the team would increase while also making it possible to perform usability-tests among team-members, since they would then be a part of the group of potential end-user.

In the second session the ideas and suggestions brought up during the first session were discussed with the supervisor, figuring out possible academic applications and approaches.

One item that got a lot of attention in both sessions was the tendency of information-sharing breaking down when it is needed the most. On the team side, the discussions centered on the negative impact on team performance and general well-being of the team members. The academic part of the discussion revolved mainly around how this negative impact could be mitigated or eliminated by improving the design and usability of the underlying communication systems.

Initially the idea was to incrementally perform alterations to the existing design of the communication system used, and then perform usability testing in order to determine which, if any of the alterations would reduce the friction of use during high-stress situation. But after an initial investigation revealed that doing these kind of alteration on the current system would be very hard, the idea was scrapped.

Discussing alternatives, it was decided that creating a web-based application where it would be possible to conduct task-based usability-tests with tasks based on current managerial pain-points would strike a good balance between academic application and real-world benefits for the team.

In order to deicide what type of mock-assignments the tests should be constructed from, three in-person interviews were conducted with managers of the team. The managers were asked about what they would like to see implemented in order to make their day-to-day work easier, producing the following ideas:

- An easy way to see if a co-worker is assigned more work than they have available hours.
- Calendar overview where it is possible to determine if there are hot-spots where lots of results need to be produced at the same time.
- A concise way to identify if there are critical tasks that, if delayed, would delay other tasks that depend on it.
- The possibility to identify a group or teams strengths and assign task types accordingly.

3.2 Lo-fi prototypes

The prototype designs were created with pointers from Krugs *Don't make me think, revisited : a common sense approach to web usability*[16] in mind, mainly, "Break up pages into clearly defined areas" and "Make it obvious what's clickable". The second point also reflects the concept and importance of '*affordance*', a term used by Donald Norman, in the oft-cited design classic *The design of everyday things*[17].

In the context of the book, affordance refers to *"the relationship between a physical object and a person (or for that matter, any interacting agent, whether animal or human, or even machines and robots). An affordance is a relationship between the properties of an object and the capabilities of the agent that determine just how the object could possibly be used"* [17, p. 11]

And while the book focuses on design in the physical reality, the concepts are still easily

adaptable to a web-environment by identifying digital objects and try to figure out how to promote the right affordances, or as Krug puts it, making it obvious what it should be used for.

Keeping these principles in mind, together with suggestions gathered from the manager-interviews, four simple paper prototypes were created, shown below. The main difference being the placement of the main interface elements. The first group having them on top and the second one the left side. Lager versions of interface prototypes can be found as figures 8.2-8.4 the appendices section.

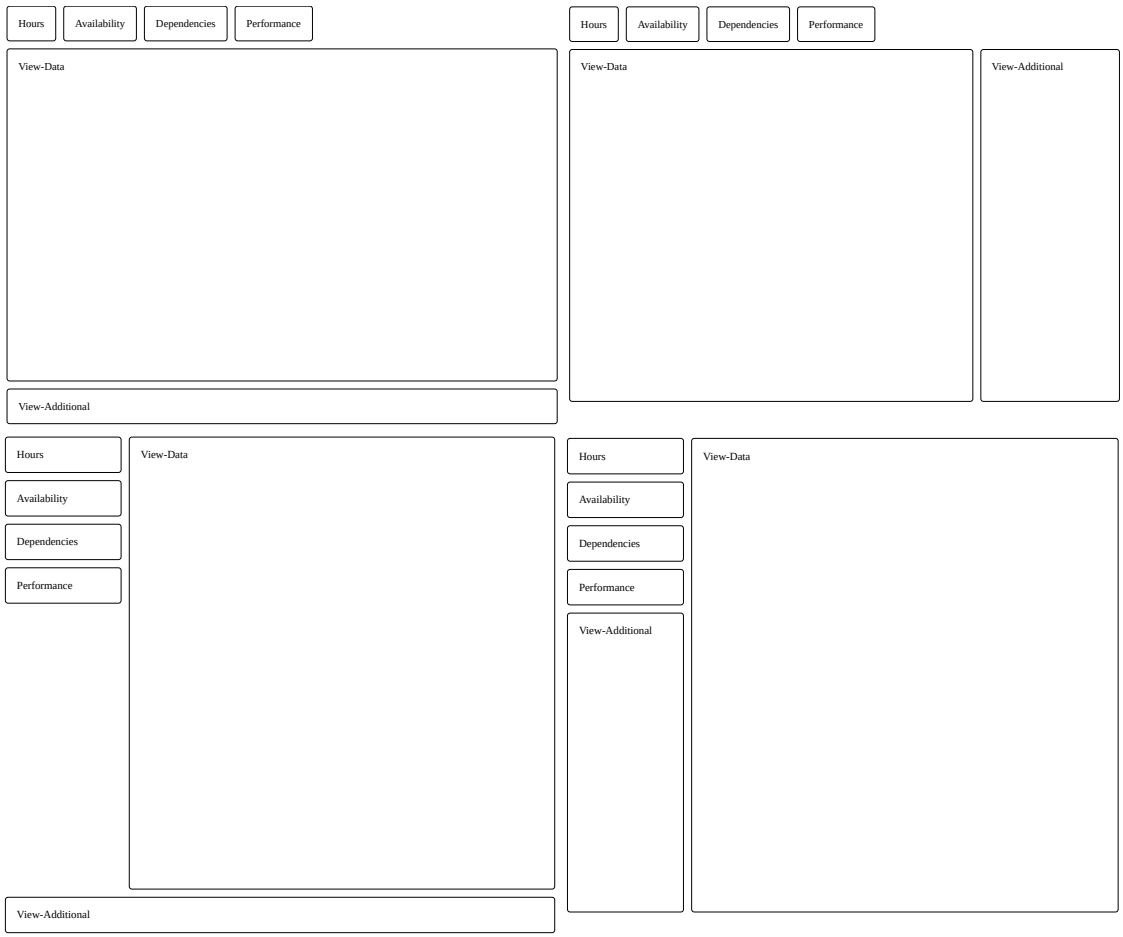


Figure 3.2: Interface drafts 1.1, 1.2, 1.3 and 1.4

In order to determine which one should be used as the basis for the high-fidelity prototype, they were all printed and made ready for evaluation, described below.

3.2.1 Interview structure for prototype selection

The specific purpose of these interviews was to determine which of the presented ui-mockups was perceived to be the most usable.

The interviews were conducted in-person, with five team members, on the premises of Massive. The interview started with a description of what was happening; an evaluation of four different mock-up interfaces to determine which one was the most suitable for further testing.

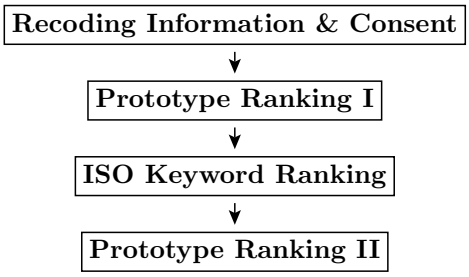


Figure 3.3: Overall structure of prototype selection interviews.

After confirming that there were no immediate questions, the interviewee was presented with the printed mockups in the same arrangement as shown in figure 3.2. They were then asked to evaluate, out loud, the interfaces in any order they wished, and to ask if there was anything unclear about the assumed functionality.

When the participant felt they were done and any questions they had about the interface were sorted out, they were asked to arrange the prototypes from best- to least-suited for the upcoming test purpose. Again, the participant was asked to voice their thought process out aloud as they did the sorting.

After confirming that the interviewee was satisfied with their ordering, a set of ten flash-cards were introduced. These cards represent the combined key-words-attributes from ISO 9241-11[12] and ISO 9241-12[18], concerning usability and presentation of information respectively.

Keyword definitions presented below:

Effectiveness
Usability is measured by the extend to which the intended goals of use of the overall system are achieved.

Efficiency
The resources that have to be expended to achieved the indented goals.

Satisfaction
The extent to which the user finds the overall system acceptable.

Clarity
The information content is conveyed quickly and accurately.

Discriminability
The displayed information can be distinguished accurately.

Conciseness
Users are not overloaded with extraneous information.

Consistency
A unique design, conformity with user's expectation.

Detectability
The user's attention is directed towards information required.

Legibility
Information is easy to read.

Comprehensibility
The maning is clearly understandable, unambiguous, interpretable, and recognizable.

Where the top three terms are defined in ISO 9241-11 with the remainder coming from ISO 9241-12.

In the next task, each participant was asked to order the keywords in order of how important they thought that specific quality was for a well-functioning, user interfacing software. There were no restrictions in regards to the number of times a participant could ask about the definition or clarification of each term.

When the participant acknowledged that they were finished with their selection, they were asked to do one final task, to re-evaluate their previously selected ranking of the prototype interfaces, changing the order if they felt another one was more appropriate after being exposed to the ISO definitions.

As for the final result of these interviews, all participants sans one, chose 1.4 as being the most usable in the initial ranking. And after the ISO definitions section everyones answered that 1.4 was the most suitable interface setup.

3.3 Technology Stack

Since this is development-based project, this section will provide insight in why the different technology choices were made, and if any other choices were considered at the time.

3.3.1 Python

The backbone of the testing-platform is written in Python[19], a general purpose programming language that is steadily becoming more and more popular among developers according to the 2019 installment of the annual developer survey[20], conducted by popular programming questions and answers site, StackOverflow[21].

In addition to the author having prior experience with Python, and the language being used heavily throughout Massive, the language is chosen due to its interesting relation to both data-analysis and web-development. When JetBrains[22], creators of PyCharm[23], a Python coding environment, asked 24 000 Python developers: “What do you use Python for?”[24], allowing for multiple selections, 59% answered data analysis and 51% web development.

Python is also the home of SciPy, “a Python-based ecosystem of open-source software for mathematics, science and engineering”[25], that has become “a de facto standard for leveraging scientific algorithms in Python”[26], making the language a good fit for gathering and processing data from studies in various ways.

3.3.2 Flask

Since one of the goals of this project is to facilitate usability-test in a geographically distributed team, presumably, over the internet, there needs to be web-component added to the mix. Here the final choice comes down down to Flask[27] or Django[28], two popular Python web-frameworks. Django facilitates an established structure and more features defined out-of-the-box, also known as "batteries included", while Flask encourages a more build-it-as-you-need-it approach.

In the end Flask was chosen because it better supports the chosen iterative development process. Additionally, it is hard to know if the project-structure enforced by Django is a good fit or not until the development has been progressing for a while, which is not ideal when development resources are scarce.

3.3.3 SVG – dynamic tasks, scaling and sharing

Scalable Vector Graphics (SVG) is, as described on the W3C-homepage, "*a markup language for describing two-dimensional graphics applications and images, [and is] …supported by all modern browsers for desktops and mobiles.*"[29] Where W3C stands for the World Wide Web Consortium, an international community that work together to develop web standards[30].

While this choice might seem a bit odd for a reader that is somewhat versed in web-development, since it would undoubtedly be easier to visualize the user-interface using static images such as .png or .jpg, both common image standards used on the internet.

However, there are specific advantages in using this markup language for this specific application. First, it is supported on a wide array of browsers and hardware, mitigating the loss of control of the underlying testing-hardware somewhat, more on this in section 4.2.

Secondly, as stated in the name, this is a vector based graphics format. This means that the underlaying primitives are made up by points and vectors instead of pixels storing color values. This approach enables the interface scale very well, both to larger and smaller formats, without any loss of quality. The usability of such a scaled interface of course assumes that the underlying interface design is done in such a way that it will still be usable at the new size, but there is no need to take concerns about loss of quality into account.

The third point is closely related to the second point. Since all the graphical information of the page will be displayed using this vector format, making a high-fidelity sharable snapshot of the current state of the page is as easy as printing the page to a pdf through the standard print-dialog in the browser viewing the page. At the time of writing, this works best in Googels popular web-browser, Chrome, and is the method used to produce all figures visualizing the platform in the following sections, feel free to zoom in on any of them if you are reading this in a digital format.

Lastly, since the format is text-based, it is possible to use any programming language that can handle text, Python in this case, to dynamically generate interfaces wholesale or to apply modifications, such as randomization or parameters-adjustment on already existing base-interfaces on the fly.

3.4 Hi-fi prototype

In order to properly begin the design of the hi-fi prototype the general test-flow for each participant is defined, using the high-level description of a test method[15, p.78] as inspiration with modifications due to the remote nature of the tests.

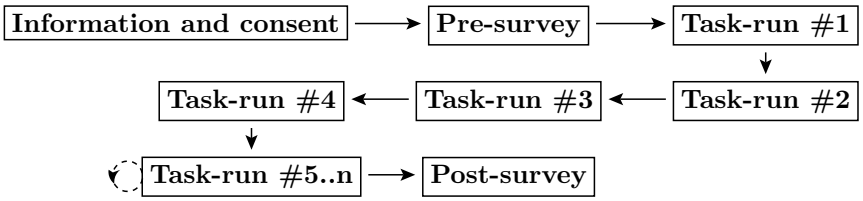


Figure 3.4: Illustrated flow for overall test-session.

The first major difference is the overall lack of time-limits. Since these tests are performed online, at a device of the participants choice at a, presumably, fitting time for them there is really no resource-restrictions in play as for how long a individual test-session could be. There is however a restriction centered around the minimum allowed tests performed, since in order to measure performance and the effects of repeated tests there needs to be a minimum number of tests performed by each participant.

Other than this, the overall test-flow follows the basic outline, beginning with a nondisclosure and privacy statement (Information and consent), followed by a combined background and experience questionnaire (Pre-survey). Then in order to complete the test-flow, a participants completes at least five tasks of their choosing, more if they want, ending with a post-test debriefing (Post-survey). Again, with the main difference being the absence of an active moderator, due to the online nature of the tests.

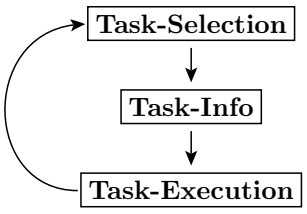


Figure 3.5: Illustrated flow for individual task-runs.

As for the flow of each individual tasks, it is split into three distinct phases; *Task selection*, *Task-Info* and *Task-Execution*. Each of the sections final design, purpose and implementation is described in greater detail in later sections below.

What is worth noting here is that, as with many of the other steps, the participant is free to spend any amount of time on the first and second section, moving between them freely. It is when moving to the last stage, execution that the interface becomes more restrictive, locking the participant into the selected task, and measuring the time to the first submitted answer regardless if it is correct or not.

3.4.1 Basis and purpose of the test-tasks

The test-tasks are derived from the items listed at the end of section 3.1 at page 17. This list contains a selected subset of the answers to the interview question of what tooling is currently missing that could make the day-to-day work for interviewed managers easier.

Using these examples of what could feasibly be implemented to add real-world value to current day-to-day activities gives a concrete foundation to base the test-task design on. However, since all of the select suggestions could be implemented as a self-contained tool or service in their own right, implementing each of them fully, with correct behaviour, is outside the scope of this report.

Due to the limited time and development resources mentioned above, each suggestion will be abstracted to its simplest goal that still contains the core function of the original suggestion. With the overarching goal being measurements of the time-to-completion performance of the participants, the main focus of the tasks will be to define a clear completion criteria in order to make it possible to precisely measure the elapsed time.

3.4.2 General information, consent and initial survey

When accessing the test the participant is greeted by a information and consent screen, detailing the goals of the test and how the information generated by their activity will be used. The page explains that this is a usability study, and though there will be information collected, anything that will be publicised will be aggregations and include no personally identifiable information. The full information and consent page can be found in figure 8.1.

Participant that accept and submit the consent form are then navigated to the main landing page. This page acts as a hub for the duration of the test session, providing access to all other parts of testing application. On the first visit the main view of the landing page is occupied by the initial survey, shown below, with a larger version available in figure 8.6 in the appendices.

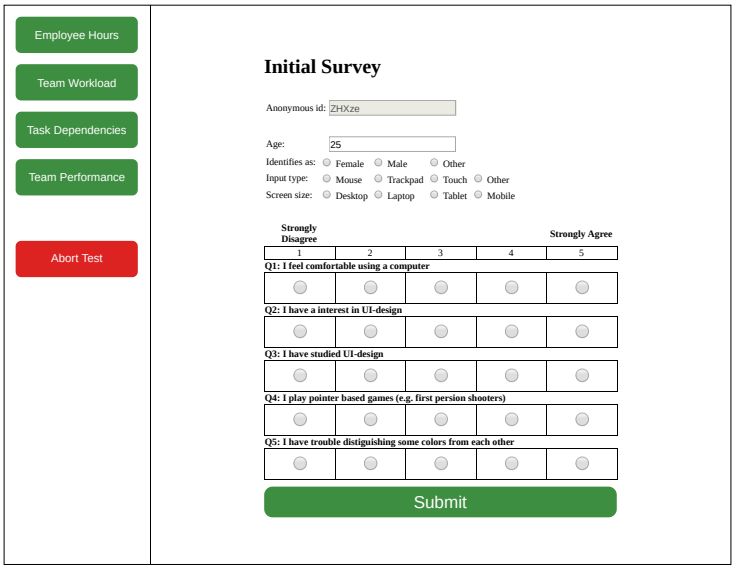


Figure 3.6: Capture of initial survey page.

While it is possible for the participant to interact with the buttons on the left side before the initial survey is submitted, pressing any of the buttons except 'Abort Test' will only display the same survey-page.

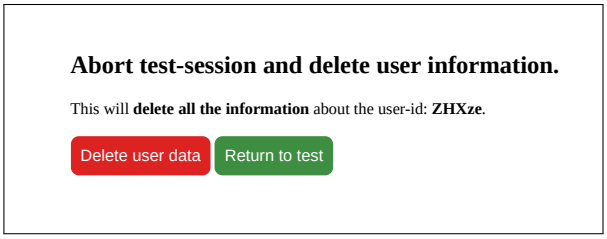


Figure 3.7: Capture of confirmation page for aborting the test.

The dedicated 'Abort Test' button is available in all the different views of the application, in case the participant no longer wants to participate for any reason. Pressing this button displays a page explaining that continuing will scrub any activity related to their current anonymous id from the database and abort the current test.

While the choices for the input method, except the last one, corresponds directly to a concrete input method used in human-computer interactions, the screen-sizes are somewhat ambiguous. This was an active choice in order to not alienate participants by forcing them to specify an actual measurement for the screen size. The assumed values for the screen-sizes ranges for the given choices are: *Desktop* >17", *Laptop* 12"-17", *Tablet* 7"-12" and *Mobile* <7".

3.4.3 Landing page

After the participant has submitted the pre-survey, it disappears and is replaced by a few basic statistics about the current test session. The functionality of the left-side buttons is restored and makes it possible to choose any of the following four test types: *Employee Hours*, *Team Workload*, *Task Dependencies* and *Team Performance*.

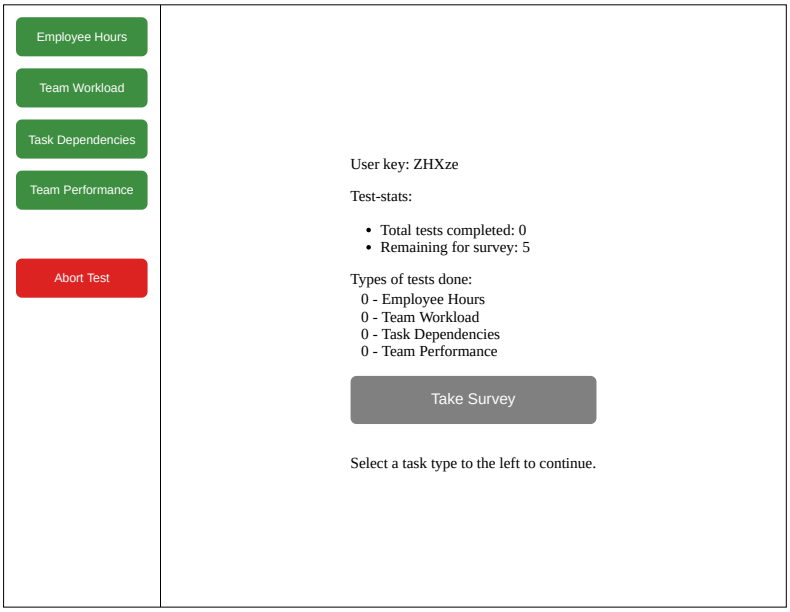


Figure 3.8: Capture of the hub page, post initial survey, default state.

The session-statistics include the participants assigned anonymous id, how many test of each type they have completed and how many test they have to complete in order for the post-survey to become available.

As previously stated, requiring five tests before making the post-survey available is the only hard limitation placed on the participants. Other than the test-minimum there is no limit on how many tests they can perform or how long time they take to complete a individual test or the test session as a whole.

3.4.4 Information, fictional context and execution

Before starting any of the tasks, the participant is greeted by a page containing the general information about the task at hand. Each of the pages follows the same general structure, three sections, in the following order *Goal*, *How* and *Summary*.

According to the guidelines for on-screen text[31], a line should be around 55 characters long in order to maximize the ease of reading and comprehension. Any line shorter than 25 character or longer than 100 character is detrimental to the reading experience.

Keeping this in mind, the goal sections should be a short summary that informs the participant what they need to do in order to complete the task successfully.

After reading the how section, the participant should have a basic understanding of how the information related to the test is structured and presented. If there are any specific details about the representation of the task that are deemed essential they should be described in this section.

Finally, the summary section is meant to repeat the information in the how section, but in a different and briefer fashion in an attempt to help with user retention.

Lastly, the participant is told that the same information that has just been presented is available after starting the test, accessible by hovering the cursor over a '?' in the upper left corner.

Since all the task-mechanics boil down to *find the element and press it as fast as possible*, the intent is to give the participant enough concrete information about the test at hand while trying to maintain the implied fictional context of the task. Which means the tone of any instructions or information should be closer to *"select the co-worker with the most X"* rather than *"click the largest box"*.

3.4.5 Employee hours

An easy way to see if a co-worker is assigned more work than they have available hours.

The premise for this task is that the organization generating the data utilizes some form of task assigning system together with a cost-estimation system. In short, there is a set of tasks that need to be done, each with a estimated time-cost, and a number of people that can be scheduled to do said work.

Given there is a limited amount of time that any assignee can spend working, it is possible to over-schedule someone. In this scenario, being over-scheduled is the result of having more

work scheduled to you over a period of time than the total available capacity for the same period. In summary, the goal of this task consists of identifying these cases, in particular, the person that is the most over-scheduled.

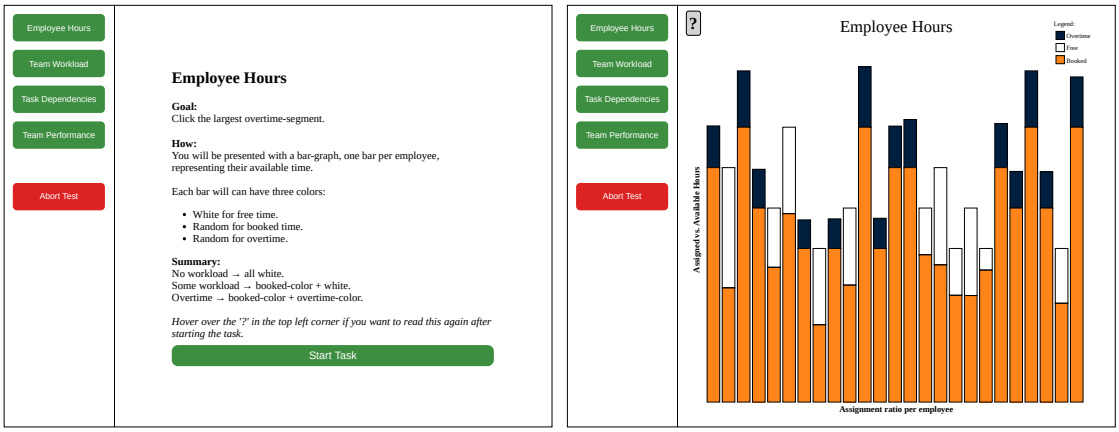


Figure 3.9: Capture of Employee Hours information and task page.

After starting the task, the participant is shown a bar-graph with bars of varying heights and coloring. Each bar shows the relation between available and scheduled work-time for one out of the twenty-five represented people.

Here, the height of the black outline of each bar represents the *available work-capacity*, the varying heights simulating the difference of available hours due to to sick-leave, vacations or different contracts.

The *total scheduled workload* for each person is represented by the height of the colored segments in each bar. If there is capacity left the colored segment will not extend to the top of the bar outline, leaving a white segment representing non-scheduled time. On the other hand, if the amount of scheduled work is greater than the available capacity, the color will extend beyond the original outline and be colored differently for readability.

In summary, the height of a white segments in a bar represent the amount of remaining capacity, while the height of any other-colored segment signifies how *over-scheduled* someone is. Find the largest other-colored segment it and click on it.

3.4.6 Team workload

Calendar overview where it is possible to determine if there are hot-spots where lots of results need to be produced at the same time.

In the following scenario the business or organization has a schedule, Monday to Friday, that stretches twenty-five weeks into the future. The goal is to keep the workload as steady as possible without to many jumps in either direction. Have to little to do and people are underutilized and in worst case, bored. Have to much that needs to be done at the same time and people might burn out. Avoiding both extremes would be preferable.

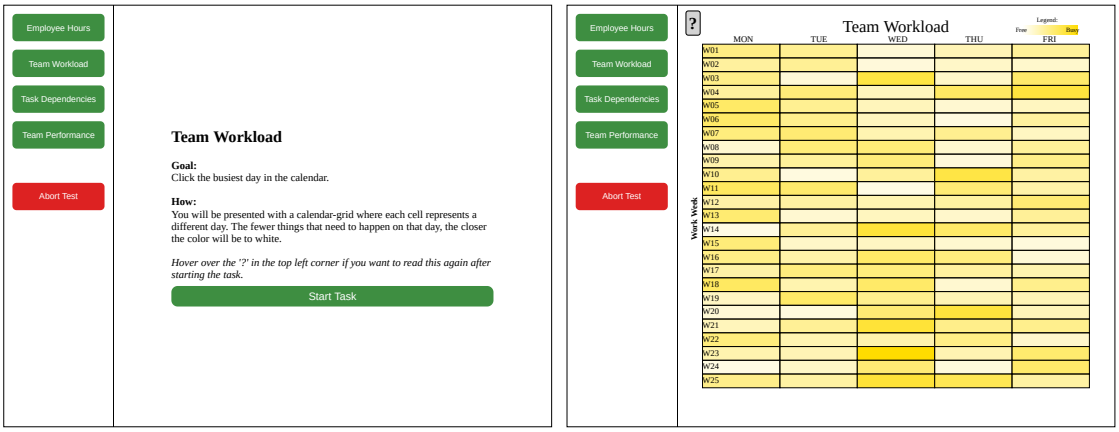


Figure 3.10: Capture of Team Workload information and task page.

Running this test shows the participant a grid of twenty-five rows consisting of five columns each, denoted `MON`, `TUE`, `WED`, `THU` and `FRI`, representing the scheduled twenty-five work-weeks mentioned above. Each of the rectangles are colored with the same color but differ in the saturation. A cell that has zero saturation, or white in this case, signal that no work is scheduled for completion on that particular day. Inversely, the darker a cell is, the more work needs to be completed on that specific day.

Since there needs to be a clearly defined goal for each test, there has to be a choice between bored and burned out people, where the latter seems far worse. Now the problem can be corrected by identifying days where there is an extra high amount of work that needs to be done, and try to re-schedule it to less busy days.

For the participant executing the test, the goal is to find the darkest, or most saturated rectangle and click it as fast as possible.

3.4.7 Task dependencies

A concise way to identify if there are critical tasks that, if delayed, would delay other tasks that depend on it.

The assumption for this test is that the underlying system producing the data has tasks that need to be completed first in order for other tasks to complete in turn, in short, dependencies. This test revolves around answering the question *"which elements are the most crucial for the overall process?"*. The answer to this question can be very useful, as an example, when being forced to cut back on things due to limited resources while trying to affect as few other things as possible.

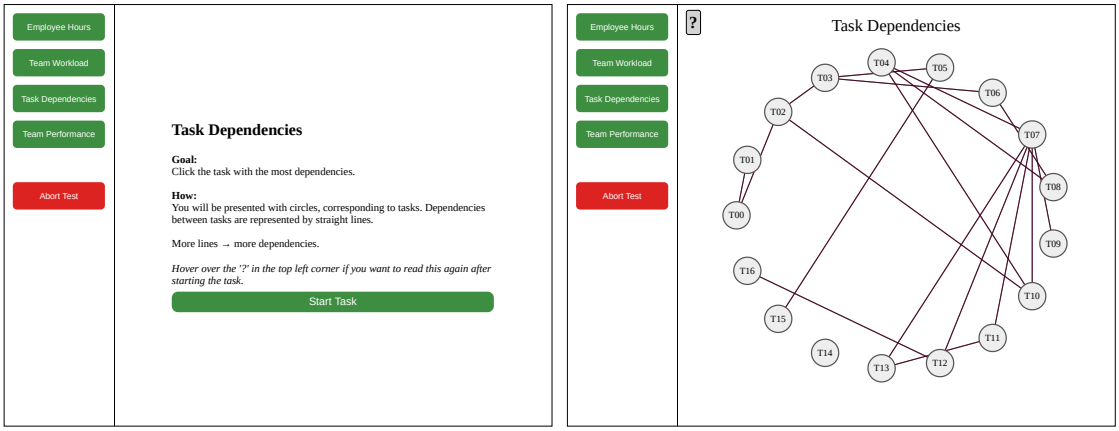


Figure 3.11: Capture of Task Dependencies info- and task-page.

Initializing this test shows the participant seventeen small circles, representing tasks, arranged in a larger circle. Additionally, there are colored lines connecting some circles together, indicating there is a dependency between them, more lines equals more dependencies.

Since the stated interest in knowing how dependent the different moving parts are on each other, knowing which task has the most dependencies is a good start. In terms of what the participant has to do to finish the test, this translates to; find the task with the most connections and click on it as fast as possible.

3.4.8 Team performance

The possibility to identify a group or teams strengths and assign task types accordingly.

Given a working-environment where things are done in parallell it becomes interesting to know if a specific group or individual is especially effective in a certain area. Having this information makes it possible to assign people, if they excel at a specific area, to tasks that match their abilities.

In this test there are five different task-categories, *tickets*, *rnd*, *support*, *features* and *maintentance*.

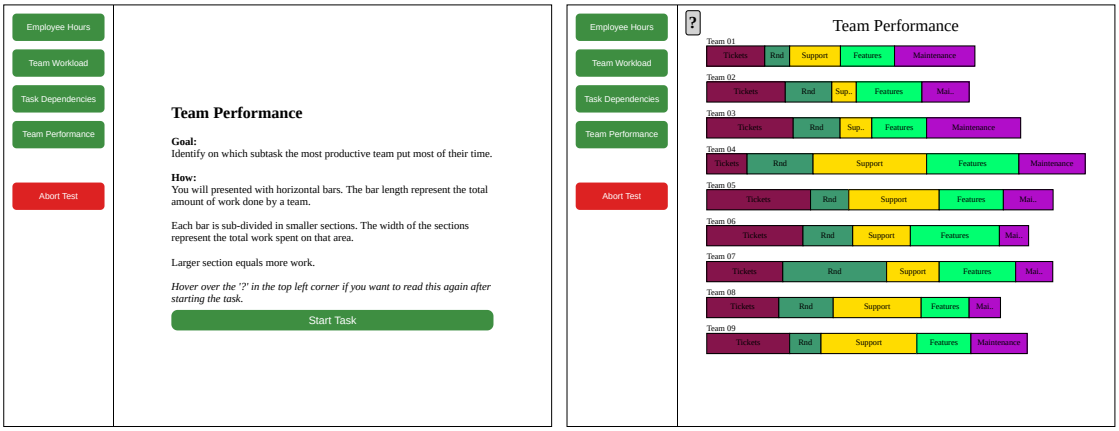


Figure 3.12: Capture of Team Performance info- and task-page.

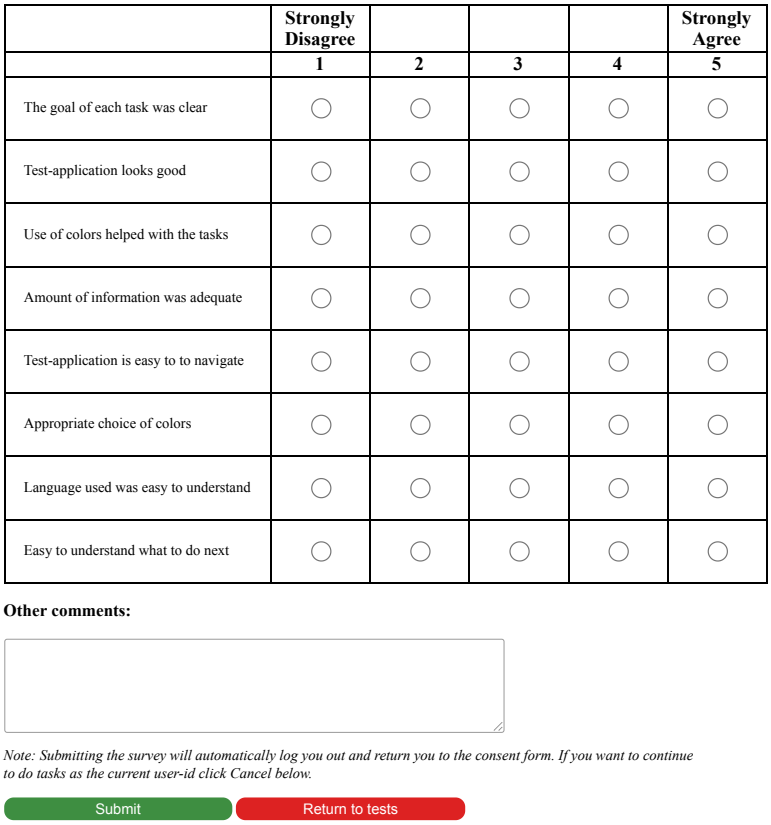
This task presents the participant with bars showcasing the total completed work-load for nine different teams over an unspecified amount of time.

There are two types of information shown, the total amount of work done by each team, represented by the total horizontal length of the bar. And the amount of the total work that went into a specific task-section, represented by the length of each individual sub-section of the overall bar. In order to correctly answer this test, the participant first needs to identify which of the bars is the longest one, and then identify which sub-section is the largest one, and click on that one.

3.4.9 Post survey

As the participant completes five or more tests the 'Take Survey' button becomes active, pressing this button sends the participant to the post-survey page. Other than the survey itself, there is also the option to return to the tests if the button was pressed by accident or if the participant had a change of mind.

The post-survey consist of eight questions and an optional input field for additional comments. These questions are ment to evaluate the participants thoughts about the current iteration of the testing application in order to derive possible improvement for the next implementation iteration.



	Strongly Disagree				Strongly Agree
	1	2	3	4	5
The goal of each task was clear	○	○	○	○	○
Test-application looks good	○	○	○	○	○
Use of colors helped with the tasks	○	○	○	○	○
Amount of information was adequate	○	○	○	○	○
Test-application is easy to to navigate	○	○	○	○	○
Appropriate choice of colors	○	○	○	○	○
Language used was easy to understand	○	○	○	○	○
Easy to understand what to do next	○	○	○	○	○

Other comments:

Note: Submitting the survey will automatically log you out and return you to the consent form. If you want to continue to do tasks as the current user-id click Cancel below.

Submit | Return to tests

Figure 3.13: Capture of the post-survey.

Options that are of interest include, among others, if the goal of each task was clear, if the application was pleasing to look at, if the use of colors help when navigating application, etcetera. For a larger version of the capture, see figure 8.7 in the appendices.

Chapter 4

Evaluation

After the test application was implemented and deployed, a link to the application was shared on the authors personal Facebook-profile together with an explanation of what the purpose of the test was. Three days later, the same link was shared on the Massive internal mailing list.

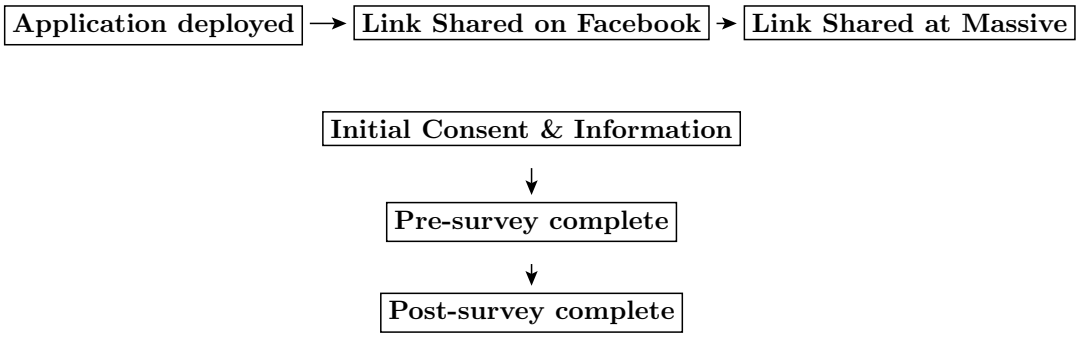


Figure 4.1: Link-distribution and test milestones.

Looking at the test setup and the final result, the participants can be grouped into three categories, depending how far through the test-session they ended up. The test structure has the following milestones:

- Getting past the initial information and consent form.
- Completing the pre-survey.
- Completing the post-survey.

4.1 Participants

Out of the total of five days the application was live, in the first three days there were 27 participants that got past the first milestone, with 23 (85.2%) of them completing the post-survey. At the third day a link to the application was shared internally at Massive. The final combined result over the five day span was; 101 participants getting past the initial information, 79 (78.2%) answered the initial survey and a total of 74 (73.3%) completing the post-survey.

Since there is no concrete information to work with for users that only passed the first milestone, that group is excluded and the pre-survey and post-survey groups will receive further analysis.

4.1.1 Age, gender-identity and completion times

In the pre-post-survey group, the average age is, 30.5 years, 28 participants (32.9%) identify as female, 54 (63.5%) as male and 3 (3.5%) as other. Looking at the post-survey group, the average age of the participants shifts slightly, 31.1 years, and 24 (32.4%), 47 (63.5%) and 3 (4%) of the participants identify as female, male and other respectively.

Looking at completion times for the post-post-survey group, the average time from registration to finishing the last test-question was 20 minutes and 56 seconds with a median of 4 minutes and 23 seconds. Additionally, the fastest completion time was 37 seconds with the longest completion time clocking in at 14 hours 15 minutes and 35 seconds.

4.1.2 Prerequisites, prior knowledge and education

Other attributes that were deemed interested included; how comfortable the participant felt using a computer, if they hade any general interest in user interface design and if they had studied user interface design in any capacity. They were also questioned on if they regularly practiced precise mouse movements through games and similar activities and finally if they have trouble distinguishing colors from each other. These questions were posed as a personal statement, such as, "*I feel comfortable using a computer*" together with a range of selectable numerical options, ranging from one, *strongly disagree* to five, *strongly agree*.

4.2 Dealing with varying testing hardware

Since the tests were done remotely via the internet, only requiring a device capable running a standard browser, there was no standardized hardware-setup that the tests were executed on. In order to have the possibility to analyze impact of different types of hardware and group different results, the pre-questionnaire included questions about what kind of hardware the participant was using to access the tests.

It was concluded that screen size and input method had the largest probability to affect the result given the design of the tests. The participants were asked to specify their screen size

by selecting one of the following options: *desktop, laptop, tablet* or *mobile*. As for the input method the choices were: *mouse, trackpad, touch* or *other*, see figure 8.6 for a capture.

4.3 Results

This section presents the different data categories extracted from the test-data received after concluding the initial run of the testing-platform. The main focus is put on the individual data-groupings without discussing the data in a larger context, since this is done later.

4.3.1 Pre-questionnaire – itemizations

Before being able to perform the test, each participant has to fill out and submit a pre-questionnaire. This is done in order to get a rough demographical overview of the people participating in this study. Initially they are asked about age, used input device, type of screen category the test were conducted on, and what binary gender, if any, they identify as.

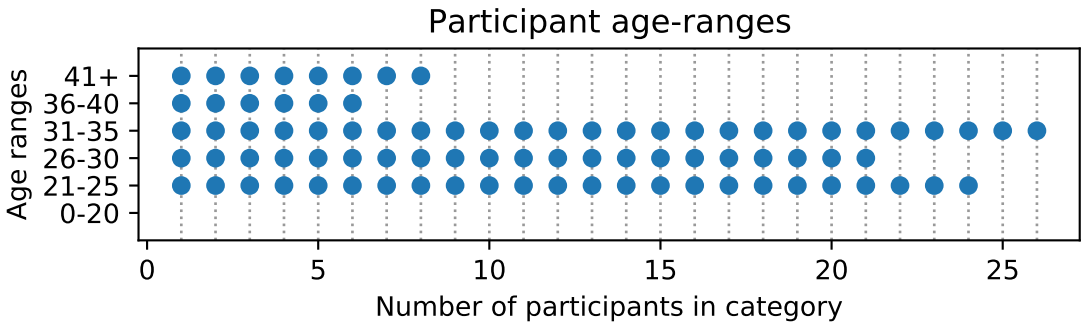


Figure 4.2: Answers for pre-questionnaire age-ranges.

Beginning with age, with each dot representing one answer, no participant was twenty or below, most of the participants were between twenty-one and thirty-five, with fourteen being thirty-six and older. Since a large part of the participants came from MASSIVE, this seems to fit the mean age at the company, which is at thirty-two at the time of this writing.

It's is worth mentioning however, that the most represented single age was twenty-five, which is the default value set for the age input field, more reflection on this specifically in section 5.6.2.

Asking people for what gender they identify as results in the largest group, 52 participants, identify as male. Again, as a large portion of participants came from MASSIVE, this is expected since the gender distribution in businesses related game-development, at the time of writing, is weighted toward a majority of men[4, p.16].

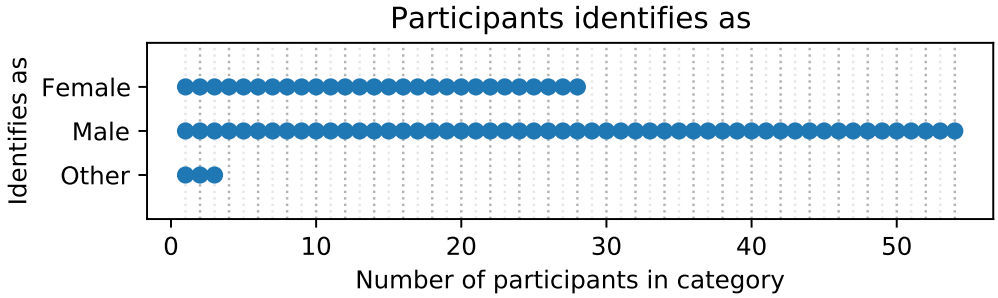


Figure 4.3: Answers for the pre-questionnaire gender identity.

Continuing the trend, since most workstations at MASSIVE consist of a desktop PC where the main peripherals used for input are a mouse and a keyboard with the occasional drawing tablet mixed in, this is the most represented category in the results.

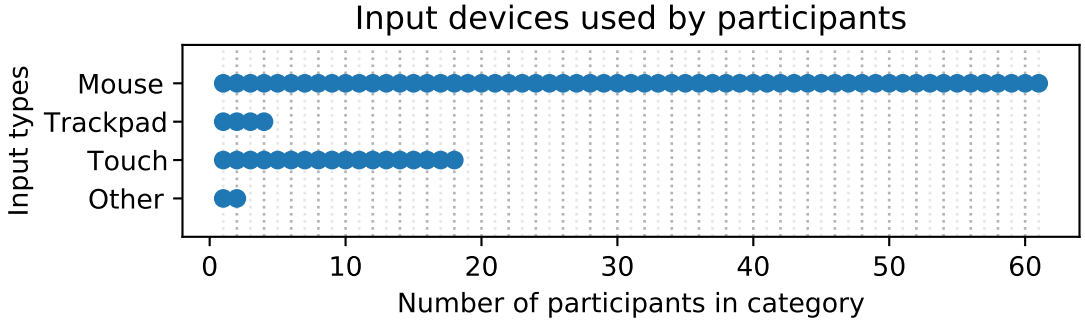


Figure 4.4: Answers for the pre-questionnaire input device.

In general it was assumed that most of the participants were going to use a mouse and keyboard, with trackpads as a probable second place, making the turnout of nearly twenty participants using touch as their main input a bit of a surprise.

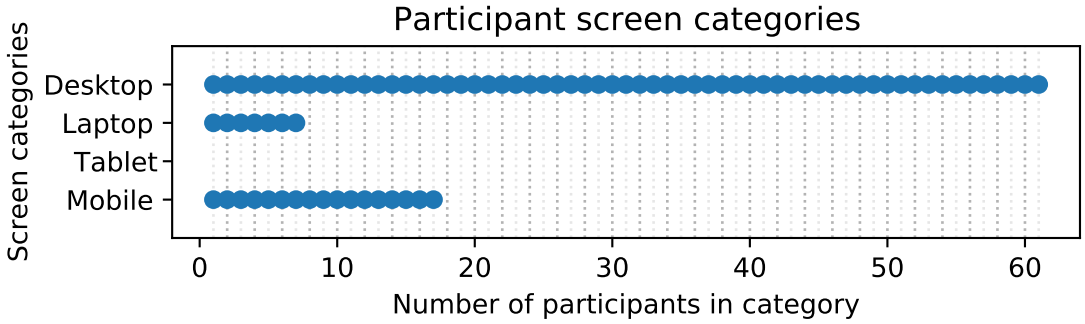


Figure 4.5: Answers for the pre-questionnaire.

Requiring actual screen-sizes was deemed to cumbersome of a task to ask of the participants, especially if using devices other than a somewhat standard desktop monitor. Here the categories roughly equate to; *Desktop*: 18" or larger, *Laptop*: 13"-17", *Tablet*: 11"-12" and *Mobile*: 10" and below.

4.3.2 Pre-questionnaire results

Aside from the physical setup, there are relevant knowledges and experience that could become interesting when analyzing free-form answers and test-results gathered from the participants. As an example, is the free-form feedback different from someone that is interested in usability-design? Does the level of computer literacy or experience with mouse-driven games impact completion times? Etcetera.

In order to evaluate these aspects the second half of the pre-questionnaire included five questions on a five-point Likert scale, questions and answer distribution displayed below:

Q1: I feel comfortable using a computer.

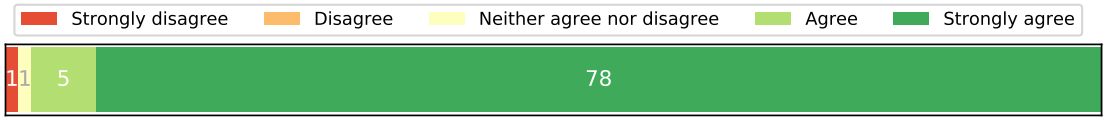


Figure 4.6: Answers for the pre-questionnaire Q1.

The clear majority of participants feel that they are very comfortable using a computer. Since this question is so heavily skewed towards *Strongly agree*, it would be interesting to add an additional *I see myself as an advanced computer user* or similar in order to try to separate out more divisions.

Q2: I have a interest in UI-design.

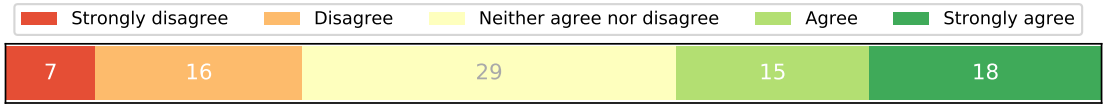


Figure 4.7: Answers for the pre-questionnaire Q2.

With this pretty even spread, it would once again be interesting to follow up on the ends of the spectrum such as *do the seven participants that have a strong disinterest in UI-design just don't like interfaces in general?*

Q3: I have studied UI-design.

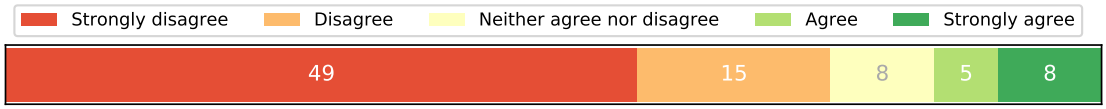


Figure 4.8: Answers for the pre-questionnaire Q3.

In terms of having studied user interface design, the majority of participants have not. Since this is a pure self-assessment, the definition of what *studied* means in this context is up for grabs. A follow up question in regards to what type of study; self-learnt, online-course, university etcetera would be needed clarify this information further.

Q4: I play pointer based games (e.g. first person shooters).

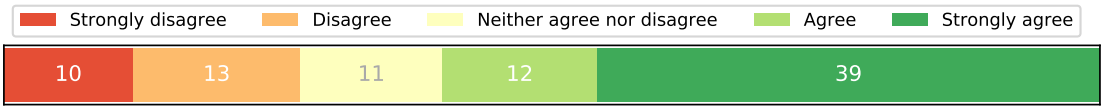


Figure 4.9: Answers for the pre-questionnaire Q4.

A slight majority agree, in some capacity, that they play games where the main form of input is pointer movements. Following this question with one that seeks to answer the underlying reason why disagreeing participants do not interact more with this kind of games could be interesting from a interface design standpoint. Additional questions or follow-up would be needed to determine if the disagreement it is a matter of taste, design faults, available time, or something completely different.

Q5: I have trouble distinguishing some colors from each other.

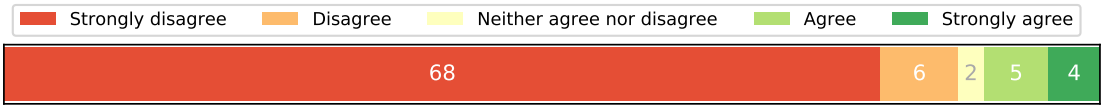


Figure 4.10: Answers for the pre-questionnaire Q5.

All of the available tests incorporates different color pallets to varying degree, which makes it interesting to know if any of the participants have trouble making out the difference between colors.

4.3.3 Launch, participation and overall success ratio

The test-site went live 2020-01-24 with the link initially shared only through the authors personal Facebook. On 2020-01-27 the link was shared on the MASSIVE internal mailing list, boosting the participation significantly. In total, 101 participants moved past the initial information over a five day period.

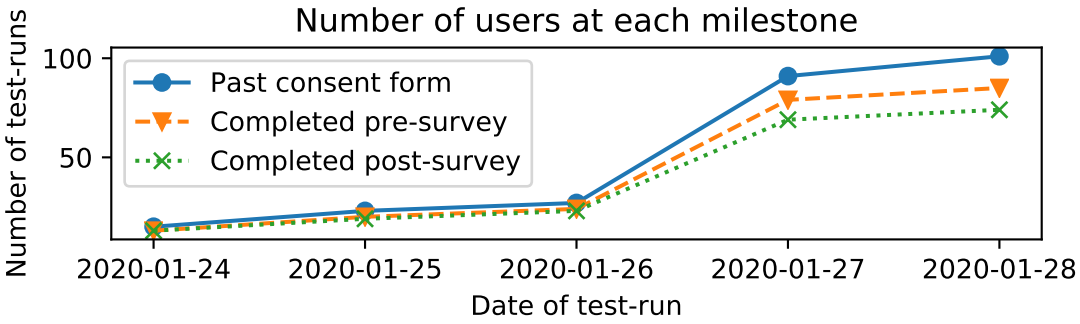


Figure 4.11: Total amount of participants that got past each of the milestones defined in figure 4.1.

In total 698 tests were run, where 537 were answered correctly, 3 never produced an answer, leaving 158 incorrect answers. Looking only on the test runs without any user- or task-correlation, the chance that any given test run produces the correct answer is ~76.5%.

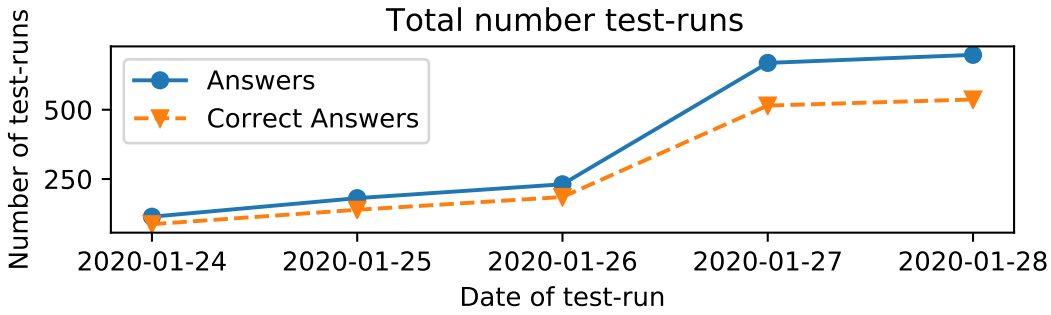


Figure 4.12: Lines representing total amount of test-runs together with the total of runs that produces the correct answer.

4.3.4 Tests per user and defining outliers

The number of recommended test was five, which when completed, allowed the participant to continue to the final survey. However, there was nothing stopping each participant from doing more or less than five. Generating a histogram for the number of performed tests per participant it is possible to determine what total number of tests was most commonly run per participant.

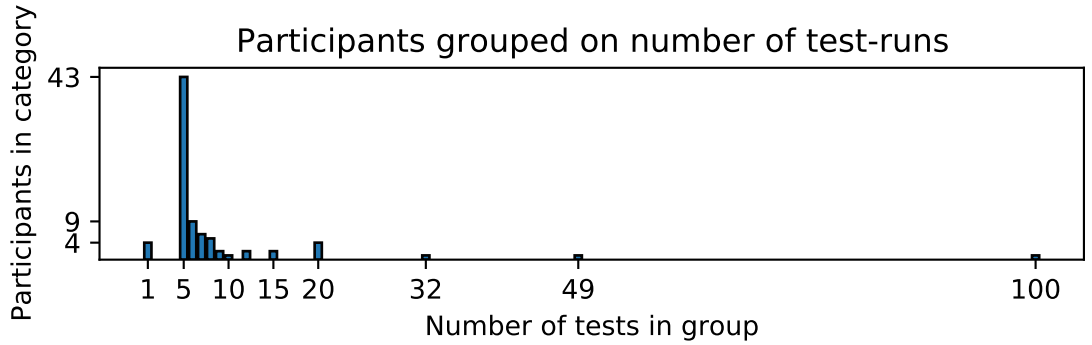


Figure 4.13: Participants grouped on how many test they performed.

In total, of the 101 number of participant that started a session, 81 of them ran at least one test, which indicates 20 ran no tests. Retrieving and tabulating additional values, discarding users with no test-runs, produces the following table.

# Tests run	# participants	Sum participants	% of total participating
$\#r \leq 1$	4	4	4.94%
$\#r \leq 5$	43	47	58.02%
$\#r \leq 6$	9	56	69.14%
$\#r \leq 7$	6	62	76.54%
$\#r \leq 8$	5	67	82.72%
$\#r \leq 9$	2	69	85.19%
$\#r \leq 10$	1	70	86.42%
$\#r \leq 12$	2	72	88.89%
$\#r \leq 15$	2	74	91.36%
$\#r \leq 20$	4	78	96.30%
$\#r \leq 32$	1	79	97.53%
$\#r \leq 49$	1	80	98.77%
$\#r \leq 100$	1	81	100.00%

Figure 4.14: Tabulated values of test-run groups with corresponding percentage of total active participants.

Participants that have run at most fifteen tests make up slightly more than 91% of the total number of participants and will be the regular group, denoted as $\#r \leq 15$. Inversely the remaining participants that have run sixteen or more tests in total, ~9% of the total amount of participant will be seen as the outlier group, denoted as $\#r \geq 16$.

4.3.5 Test type distribution among participants

There are no restrictions in place in regards to how a participant can choose which task types to preform in what order. By grouping all the task-runs based on the type of the task, it is possible to see if any type is more popular than the others.

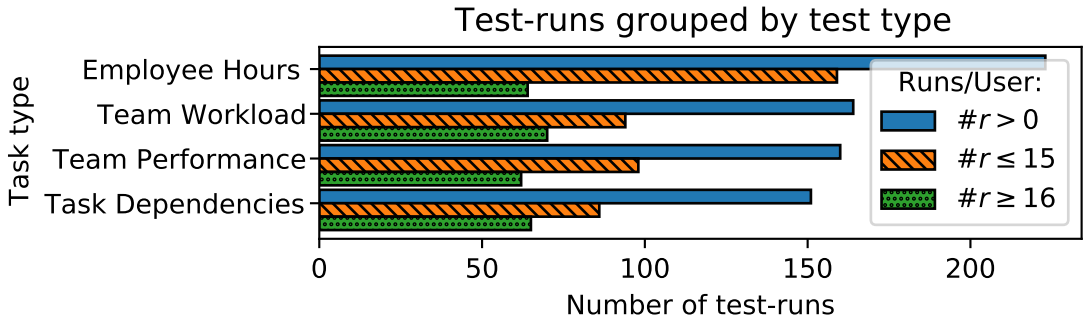


Figure 4.15: Distribution of task types among all total runs together with the total for the regular- and outlier-grouping respectively.

Examining the types distributed over all participants together with the different groupings, *Employee Hours* is the most executed test type, regardless of categorization.

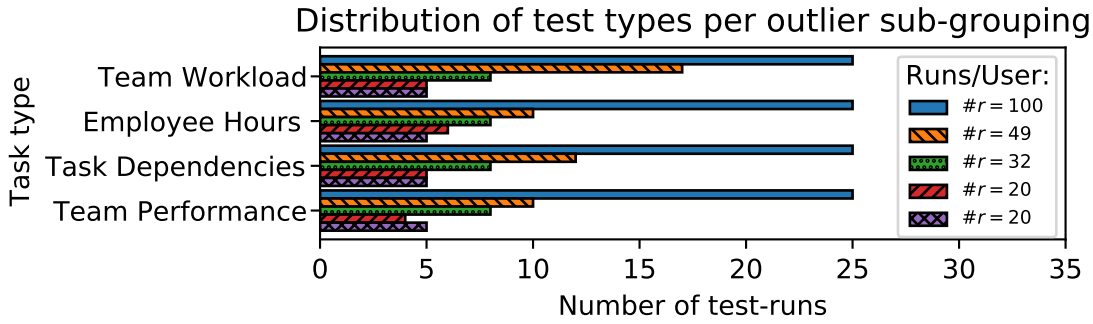


Figure 4.16: Detailed breakdown of task distribution for the outliers, multiples of identical distributions removed.

Taking a closer look at the distribution in the outlier group, $\#r \geq 16$, reveals that almost all of them, six out of the total seven, are symmetrically split between all four task-types. Out of the sums [20, 20, 20, 32, 49, 100], it is only 49 that is not evenly divisible by four. This has the added effect that the participants in the outlier grouping, even though they might have ran the most total tests comparatively, do not impact the type distribution since a symmetrical distribution cancels itself out in this case.

4.3.6 Checking for preferential task order

As with the task-types, there are no no restrictions on in what order a participant can preform tasks. Grouping the tasks for each participant and sorting them in chronological order results in a view into what order each participant choose to run the different tasks.

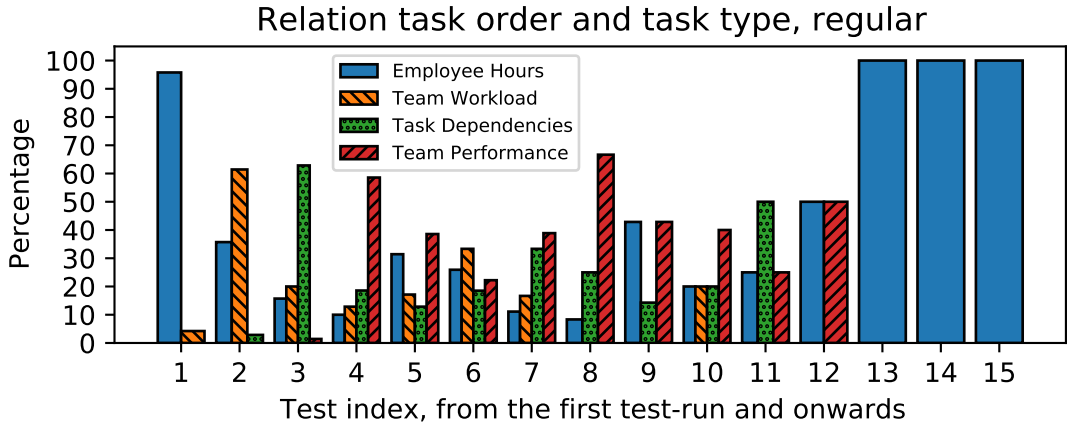


Figure 4.17: Bar-graph showing the ratio of specific task-types depending on the the chronological order of the run.

Since the legend in figure 4.17 reflects the order in which the tasks appear in the user interface for participants, the majority choose to do the tasks in the presented order, top to bottom. As for the fifth test run, the majority choose do an extra of the last one (Team Performance), and after that, most participants opted to go back and do the first one again (Employee Hours).

Given the wider range of the data coming from the outlier group (1-100) the visualization needs to be altered slightly. Since there is not enough room to have the bars side by side, they have been stacked, with the largest bar being at the bottom.

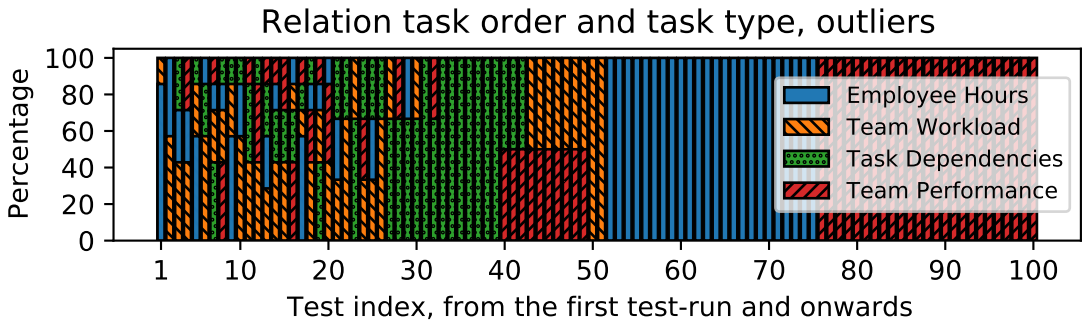


Figure 4.18: Stacked bars showing relation between task order and task type for the outlier group.

Initially the going-by-order tendency from figure 4.17 holds for the first and second index, but breaks down on the third, where the second option becomes the most picked one instead of the third. This is expected, since there are fewer participants in the outlier group, any deviations by a single participant will affect that ration more directly. Apart from that, it seems the preferable way to do thirty or more tests is to do them in batches.

4.3.7 Success-rates and task type

When completing a test-run the application marks the result as either correct or wrong in the underlying database. Extracting those answers and grouping them by the task-type makes it possible to compare the relative success- and failure-rates for each task-type.

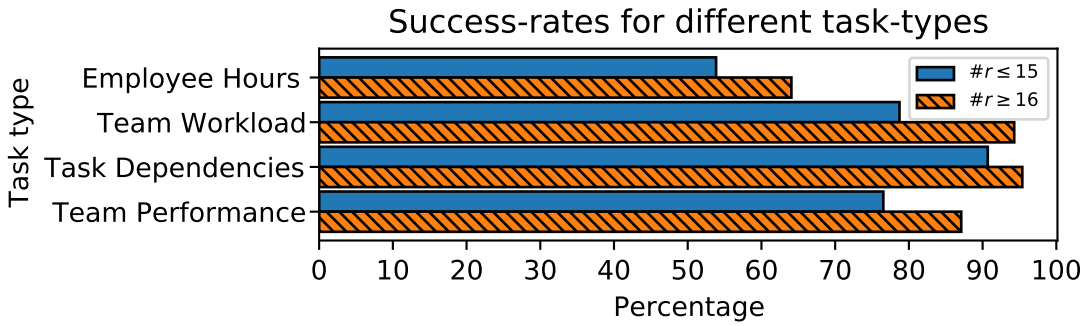


Figure 4.19: Bars showing percentages of all answers that are correct for a given task-type for regular and outlier groupings.

Reading the graph, the difficulty of the task-types in order of hardest to easiest is as follows: *Employee Hours* is the hardest to get correct, followed by *Team Performance*, *Team Workload* and finally, the easiest to get a correct answer on, *Task Dependencies*. According to the same data, the outlier group seems to be on average, more correct than the regular group.

4.3.8 Completion times - Task types and distribution

Computing the mean and average completion time for the task-runs is simply a matter of gathering all the start and stop times from the database and perform the corresponding arithmetics. The resulting times, split in regular and outliers, is shown below.

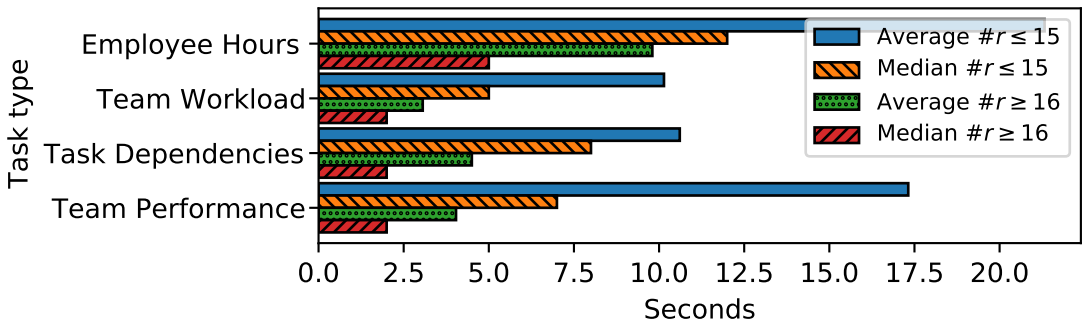


Figure 4.20: Average and median completion times for task-types.

Employee Hours has, according to the data, the longest average and median completion time of all the task types, which holds for both groupings. This makes sense since that the earlier analysis, looking at the overall success-rate for different tasks-types, points to *Employee Hours* being the hardest task type of the four.

By adding a few percentiles to a histogram based on the completion times for all tasks and participants, it is possible to determine that that 50% of all tasks-runs were completed in 7 seconds or less, 90% in 10 seconds or less and 95% of all runs completed below 42 seconds.

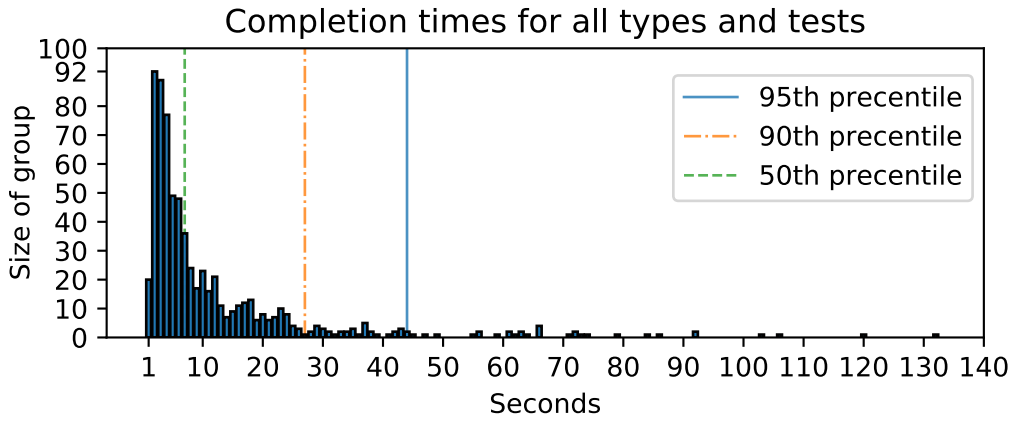


Figure 4.21: Histograms showing groupings of completion-times for all users rounded to nearest second.

Reusing the previous data by splitting it and creating one histogram each for the regular and outlier group gives a view of how the completion times are distributed for each of the groups respectably, result shown below.

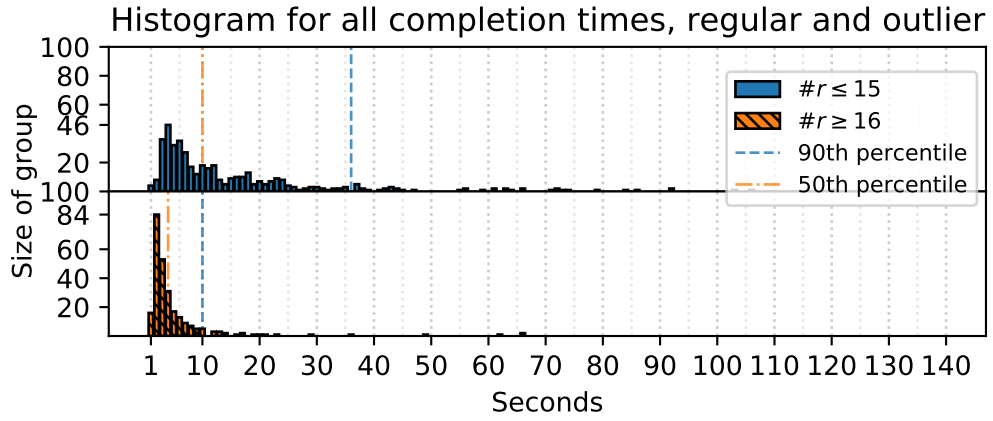


Figure 4.22: Histograms showing groupings of completion-times rounded to nearest second for; all, regular and outliers user-groups.

For the outlier group, 50% of all tests were completed in 3 seconds or less, compared to 9 seconds or less for participants in the regular group. This pattern is repeated for the 90th percentile, where 90% of the tests in the outlier group were completed in 9 seconds or less, compared to 35 seconds or less in the regular group.

4.3.9 Post-survey questions

In order to gather feedback for evaluation and possible incorporation into the next design iterations, the test concludes with a second questionnaire with eight questions on a five-point Likert scale. The goal of this questionnaire is to evaluate what participants thought about the setup, and if they have any suggestions, comments or improvements.

Q1: The goal of each task was clear.

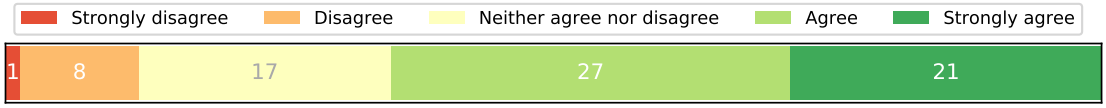


Figure 4.23: Answers for the post-questionnaire Q1.

In regards to the overarching platform-design, the desire was to create a set of fairly challenging tasks, surrounded by a interface that is easy to navigate and simple to understand, while also providing enough information to facilitate the completion of tasks. And since most of the participant felt they knew what to do in regards to the tasks, this seem to have been the case.

Q2: Test-application looks good.

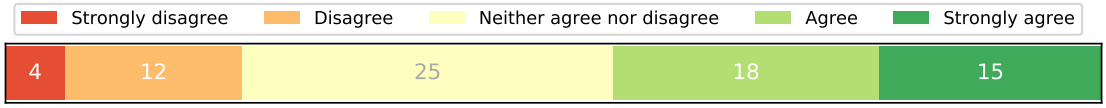


Figure 4.24: Answers for the post-questionnaire Q2.

This reads either as the participants being nice since the tone of the project is personal, or the design was too polished. If the latter case is true, it indicates that this first iteration should have been in the hands of participants sooner.

Q3: Use of colors helped with the tasks.

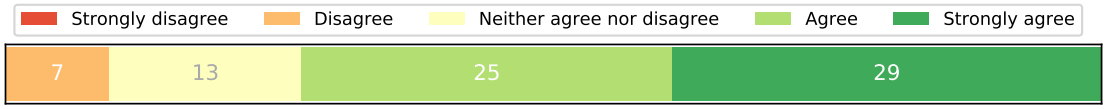


Figure 4.25: Answers for the post-questionnaire Q3.

Most people agreed that the color helped them preform their tasks. This question could be augmented with additional questions that asks more specifically about the perceived help. Additionally, it could be complemented with task-permutations that do not contain any color in order to gather test-data about the possible difference.

Q4: Amount of information was adequate.

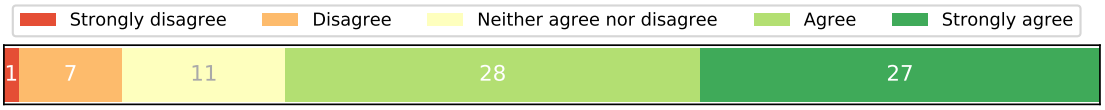


Figure 4.26: Answers for the post-questionnaire Q4.

The Majority answered that the information that they were provided was adequate. Again, it would be very interesting to ask the participants that disagree what they felt was missing.

Q5: Test-application is easy to to navigate.

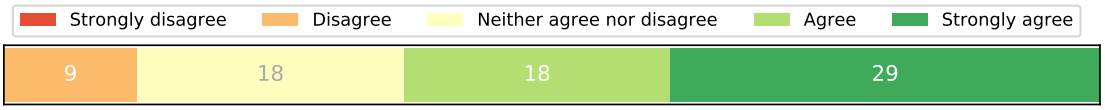


Figure 4.27: Answers for the post-questionnaire Q5.

As stated earlier, participants should only need to apply them self when doing the actual tests. The goal is that the navigation should be easily traversable, which the majority of participants seem to agreed with.

Q6: Appropriate choice of colors.

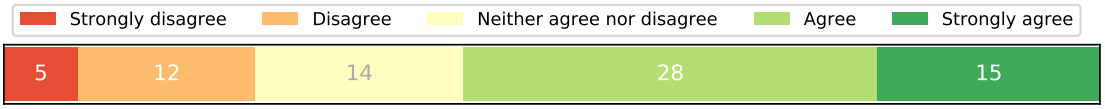


Figure 4.28: Answers for the post-questionnaire Q6.

The questionnaire shows that most participant thought that choice of colors were appropriate.

Q7: Language used was easy to understand.

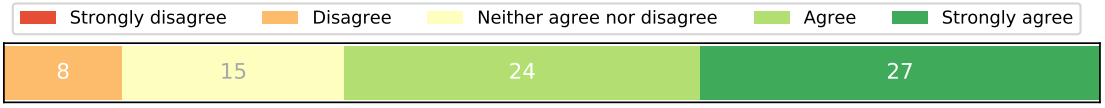


Figure 4.29: Answers for the post-questionnaire Q7.

A majority of participants agreed that the language used in the application was easy to understand, with none of the participants strongly disagreeing with the statement.

Q8: Easy to understand what to do next.

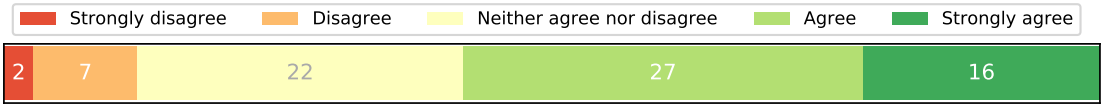


Figure 4.30: Answers for the post-questionnaire Q8.

It is encouraging that most participants felt that they knew what to do next when performing the tests. It would of course be preferable if every one felt they knew what to do, but the result is encouraging.

Chapter 5

Discussion

This section reflects on the general processes underpinning the procject as a whole and any improvemets or special considersations that have come to light during its execution and completion.

5.1 The design process

Having the paper prototypes as an initial lo-fi starting point and conducting the interviews to determine which to use worked very well, and is an element that should be repeated if a similar process would be done in the future.

That being said, it is regrettable that the iterative design process did not have a chance to get into gear and display its full potential over multiple design runs. Only managing to get through one design-implement-release-and-test-cycle really hamstrung the possibility of experience positive effect of the process of dynamically evolving the platform over time in response to user feedback.

5.2 The development process

It was very easy to set up and get starting working on creating the platform both with Python and Flask. In regard to the final performance and functionality of the platform in its current state is satisfactory for the indented use.

However, given the state of the code, it would probably be very hard for someone else to re-purpose the platform for needs somewhat outside of what was done here without requiring a significant re-write.

The only remedy for this that comes to mind is more development experience in regards to this type of web related projects. It feels like it is safe to assume, that the majority of the target audience; interested individuals inside of the gaming-industry, wanting to do remote tests, would, much like the author, lack extensive experience in coding this type of project and would likely fall into the same traps given a wide-open playing field.

One way to offset this lack of experience somewhat would be to leverage the accumulated developer time put into a more opinionated web-framework like the aforementioned Django[28] project.

5.3 Deployment and gathering participants

Deploying the platform on a local server exposed to the internet was done without much hassle, and it was easy enough to gather participants by sharing a url-link to the application.

In hind-sight, the data collection would benefit from having a more robust mechanic in place in order to differentiate if a significant portion of the participants came from a specific location, as an example, Massive.

Since the initial distribution-plan did not include the internal Massive mailing list, the platform did not have this kind of mechanic in place, witch made int impossible to definitely split the final set of participants along that grouping if needed.

5.4 Results

Event though the data gathered from the tests was rudimentary and the design of the test tasks as basic as possible they were still enough to determine some interesting characteristics about the different represented cases and how users interacted with them.

Given more time and resources it would be trivial to expand the system to perform more multi-faceted data collection together with more intricately design tasks in order to test specific usability target beyond time-to-completion.

Overall, the size of the response, and how easy it was to run large batches of remote test and extract data from them with little to no experience was surprising. Especially since the data was only collected during five days before focus was shifted to processing said collected data.

5.5 Possible improvements

Even though the deployed platform impressed with the ease it collected response data, there are some considerations that should be taken into account if this process, or similar was to be repeated.

5.5.1 Let go early, fit multiple design iterations

In the gaming industry there is a commonly performed practice referred to as “creating a vertical slice”. And though there does not seem to exist an official recording of its origin, or agreed upon definition, the different variant heard by the author are similar enough that the general understanding of what the process strives to do will be stated in the context of this report as follows:

When producing a vertical slice, the goal is to create, with the least effort possible, something that engages the whole set of interconnected systems that will appear in the finalized project or product in order to verify that it is a viable effort as early as possible.

Viewing the development of this platform through this lens, the initial vertical slice should have consisted of an non-styled html-page with a single input field and submit button connected to a database, accessible by a url-link that should have been distributed for feedback and iteration within the first days of starting development.

5.5.2 Opt-in follow-up

From its inception, this project has had an clear goal of keeping the participants as anonymous as possible and only collect the bare necessities to perform basic analysis.

However, after processing the data, there are some answers that would have been interesting if they could have been followed up on. The most plausible solution while still keeping the anonymity of the participants would be some kind of option for a opt-in follow-up through anonymized email communication or similar.

5.5.3 Investigate and leverage frameworks

More time should have been spent investigating different available frame-works, both for the front-end and back-end of the platform development, given the plethora of available web-related solutions and frameworks available.

Assuming one is not already experienced, there is much to be gained by adopting one a frameworks with this type of project, even if, after gaining some experience, the initially chosen framework is switched out for a better fit.

5.6 Threats to validity

This section highlights any known discrepancies related to the theory, implementation or execution of the project that could introduce threats to any results based on the collect data.

5.6.1 Online testing and latency

Since these tests are performed online, there is always the possibility that the quality of the connection, or rather lack thereof, could influence the measurements collected by the platform.

Further iterations on similar type of timed testing should integrate some type of analysis of the connection quality established with the participant in question over a set period of time. Even though it would be hard to mitigate the effect of connection-influence, having access to this type data would make it possible to at least generate a confidence interval in regards to the registered values.

5.6.2 Default age value

Analyzing the returned data from the test pre-questionnaire, the most common age was twenty-five, which coincidentally was the precise value of the only pre-filled input field in any of the questions. Going forth, non of the fields should have a default value in order to avoid this kind of bias.

5.6.3 Users participating multiple times

Even though multiple test-runs with the same user beyond the initial five test-task runs was encouraged, running several tests as different users was not. Doing some basic correlation of the logs on the server running the platform, it seems that at least a handful of participants ran tests as different user-ids. Since this could introduce some unexpected biases to the collected data, it would be interesting to add some kind of flagging mechanism for when many different users appear to be coming from the same source, while still keeping it anonymous.

Chapter 6

Conclusions

Even though they have old roots, the concepts of user-centered design and usability-testing still feel fresh and powerful when used as the basis for a software-development process. And while it was easier to setup the initial framework for gathering and processing user-submitted test data through the internet than anticipated. The lack of experience developing a web-based project of this size combined with choosing a bare-bones web-framework with little built-support made it impossible to keep up the iteration speed required for truly benefiting from this type of iterative, design and user-centered software-development.

In conclusion, it is possible, even for a single developer with some interest, to kickstart a platform for usability-testing over the internet. And while having the testing being performed online brings many interesting advantages with it, there are some extra consideration that need to be considered. Among them, ensuring that any identifiable data is about the participants is safely stored and not accessible from the internet. And it also becomes vital to investigating and deploying procedures to mitigate and catch data anomalies that are the result of outside influence, such as unstable internet connections.

Chapter 7

Popular Science

References

[1] Jessica R. Mesmer-Magnus and Leslie A. DeChurch. 'Information sharing and team performance: A meta-analysis.' In: *Journal of Applied Psychology* 94.2 (2009), pp. 535–546. ISSN: 0021-9010.

[2] Muammer Ozer and Doug Vogel. 'Contextualized Relationship Between Knowledge Sharing and Performance in Software Development.' In: *Journal of Management Information Systems* 32.2 (2015), pp. 134–161. ISSN: 07421222.

[3] *English*. DATASPELSBRANSCHEN. URL: `https://dataspelsbranschen.se/english` (accessed 19/05/2020).

[4] *Spelutvecklarindex 2019*. DATASPELSBRANSCHEN. URL: `https://dataspelsbranschen.se/s/Spelutvecklarindex2019_v3.pdf` (accessed 19/05/2020).

[5] *2019 Essential Facts About the Computer and Video Game Industry - Entertainment Software Association*. Entertainment Software Association. URL: `https://www.theesa.com/wp-content/uploads/2019/05/ESA_Essential_facts_2019_final.pdf` (accessed 20/05/2020).

[6] *Entertainment Software Association | U.S. Economic Growth*. Entertainment Software Association. URL: `https://www.theesa.com/industry/economic-growth/` (accessed 19/05/2020).

[7] G.M. Donahue. 'Usability and the bottom line'. In: *IEEE Software, Software, IEEE, IEEE Softw* 18.1 (2001), pp. 31–37. ISSN: 0740-7459.

[8] T. Boren and J. Ramey. 'Thinking aloud: reconciling theory and practice'. In: *IEEE Transactions on Professional Communication, Professional Communication, IEEE Transactions on, IEEE Trans. Profess. Commun* 43.3 (2000), pp. 261–278. ISSN: 1558-1500.

[9] K. Anders Ericsson and Herbert A. Simon. *Protocol analysis : verbal reports as data*. A Bradford Book. MIT Press, 1993. ISBN: 0262050471.

[10] Jakob Nielsen. *Usability engineering*. AP Professional, 1993. ISBN: 1125184069.

[11] J. May. 'YouTube Gamers and Think-Aloud Protocols: Introducing Usability Testing.' In: *IEEE Transactions on Professional Communication, Professional Communication, IEEE Transactions on, IEEE Trans. Profess. Commun* 62.1 (2019), pp. 94–103. ISSN: 0361-1434.

[12] *Ergonomics of human-system interaction — Part 11: Usability: Definitions and concepts*. Standard. International Organization for Standardization, 2018.

[13] *User centered system design : new perspectives on human-computer interaction*. Erlbaum, 1986. ISBN: 0898597811.

[14] Jan Gulliksen et al. 'Key principles for user-centred systems design'. In: *Behaviour & Information Technology* 22.6 (2003), pp. 397–409. ISSN: 0144929X.

[15] Jeffrey Rubin and Dana Chisnell. *Handbook of usability testing: how to plan, design and conduct effective tests*. John Wiley & Sons, 2008.

[16] Steve Krug. *Don't make me think, revisited : a common sense approach to web usability*. New Riders, 2014. ISBN: 0321965515.

[17] Donald A. Norman and Donald A Norman. *The design of everyday things*. Basic Books, 2013. ISBN: 0465072992.

[18] *Ergonomic requirements for office work with visual display terminals (VDTs) — Part 12: Presentation of information*. Standard. International Organization for Standardization, 1998.

[19] *Welcome to Python.org*. Python Software Foundation. URL: `https://www.python.org` (accessed 04/03/2020).

[20] *Stack Overflow Developer Survey 2019*. Stack Exchange Inc. URL: `https://insights.stackoverflow.com/survey/2019` (accessed 04/03/2020).

[21] *Stack Overflow - Where Developers Learn Share, & Build Careers*. Stack Exchange Inc. URL: `https://stackoverflow.com/` (accessed 04/03/2020).

[22] *JetBrains: Developer Tools for Professionals and Teams*. JetBrains s.r.o. URL: `https://www.jetbrains.com/` (accessed 04/03/2020).

[23] *PyCharm: the Python IDE for Professional Developers by JetBrains*. JetBrains s.r.o. URL: `https://www.jetbrains.com/pycharm/` (accessed 04/03/2020).

[24] *Python Developers Survey 2019 Results*. JetBrains s.r.o. URL: `https://www.jetbrains.com/lp/python-developers-survey-2019/` (accessed 04/03/2020).

[25] *SciPy.org — SciPy.org*. SciPy developers. URL: `https://www.scipy.org` (accessed 10/03/2020).

[26] Pauli Virtanen et al. 'SciPy 1.0: fundamental algorithms for scientific computing in Python'. In: *Nature methods* 17.3 (2020), pp. 261–272. ISSN: 1548-7105.

[27] *Flask | The Pallets Projects*. Armin Ronacher. URL: `https://palletsprojects.com/p/flask/` (accessed 10/03/2020).

[28] *The Web framework for perfectionists with deadlines | Django*. Django Software Foundation. URL: `https://www.djangoproject.com/` (accessed 10/03/2020).

[29] *W3C SVG Working Group*. W3C. URL: `https://www.w3.org/Graphics/SVG/` (accessed 12/03/2020).

[30] *ABOUT W3C*. W3C. URL: `https://www.w3.org/Consortium/` (accessed 12/03/2020).

[31] Anuj A. Nanavati and Randolph G. Bias. 'Optimal Line Length in Reading–A Literature Review.' In: *Visible Language* 39.2 (2005), pp. 121–145. ISSN: 0022-2224.

[32] William Albert, Thomas Tullis and Donna Tedesco. *Beyond the Usability Lab. Conducting Large-scale Online User Experience Studies*. Elsevier, 2010. ISBN: 9780123748928.

Chapter 8

Appendix

Introduction to this usability-study

First off, thank you for taking the time to participate in this usability study! My goal is to take no more than fifteen minutes of your time.

Since this script serves as a replacement for me overseeing the test-session personally, I will keep the tone conversational and non-formal.

What is this? - This usability study is part of my master's thesis at the Department of Design Sciences - Lund University - Faculty of Engineering.

Purpose? - The goal is to examine how different stylistic elements, such as, color and contrast impact the completion time of simple, pointer-based task.

What are we doing exactly?

After pressing the Submit button below, you will be taken to the main page and presented with four different types of tasks, together with the stats for your session.

Each of the available tasks is an abstracted representation of what a manager might want to know about their team: team-performance, overbooked employees, etcetera.

Pressing a task button will take you to a more detailed description of what the goal of each particular task is.

Getting cold feet

If you feel that you no longer want to participate, for any reason, there is a 'Abort Test' button slightly below the task buttons. Pressing this button and confirming on the next page will delete all information about the session from the database, no hard feelings.

Data collection and publication

My goal is to record as little, ideally no, personally identifiable information about the participants of this test.

If identifiable patterns emerge I will not be including any of that information in anything that I make publicly available.

I will do some statistical analysis on these results, but, any results will be presented as an aggregate and not focus on individual test-results.

Surveys

Pre-test

In order to know how you compare to other participants, there will be a few questions about age, computer-literacy etcetera before you can start doing the tasks.

Post-test

I've also included a short, less than ten questions, post-test survey. This survey is accessible after completing five or more tasks (but you are, of curse welcome to do as many as you like!). The post-survey needs to be completed in order for the results of the session to register correctly.

☐ I have read and understand what I'm participating in.

Submit

Figure 8.1: Capture of information and consent page.

Figure 8.2: Interface draft 1.1

Figure 8.3: Interface draft 1.2

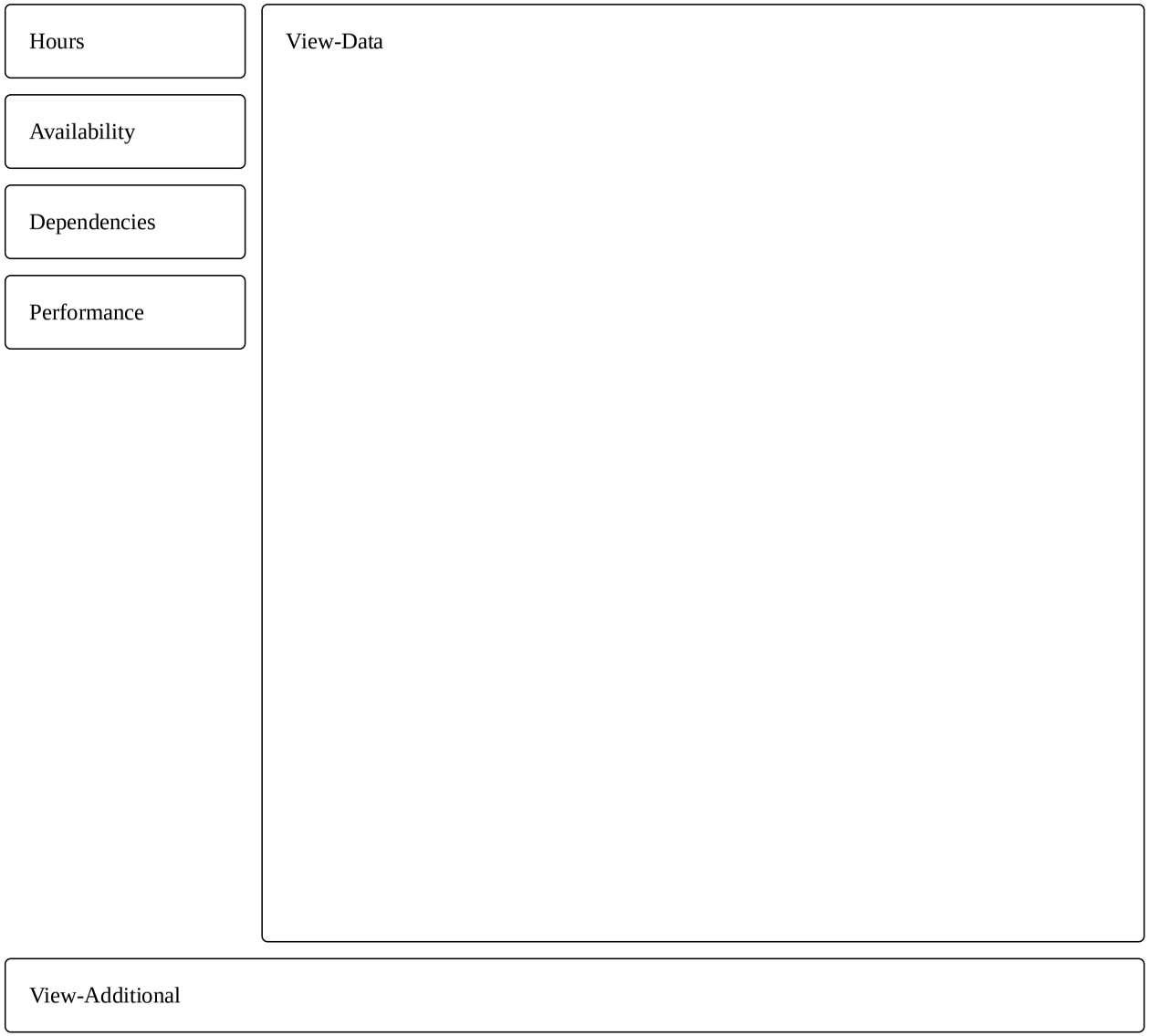


Figure 8.4: Interface draft 1.3

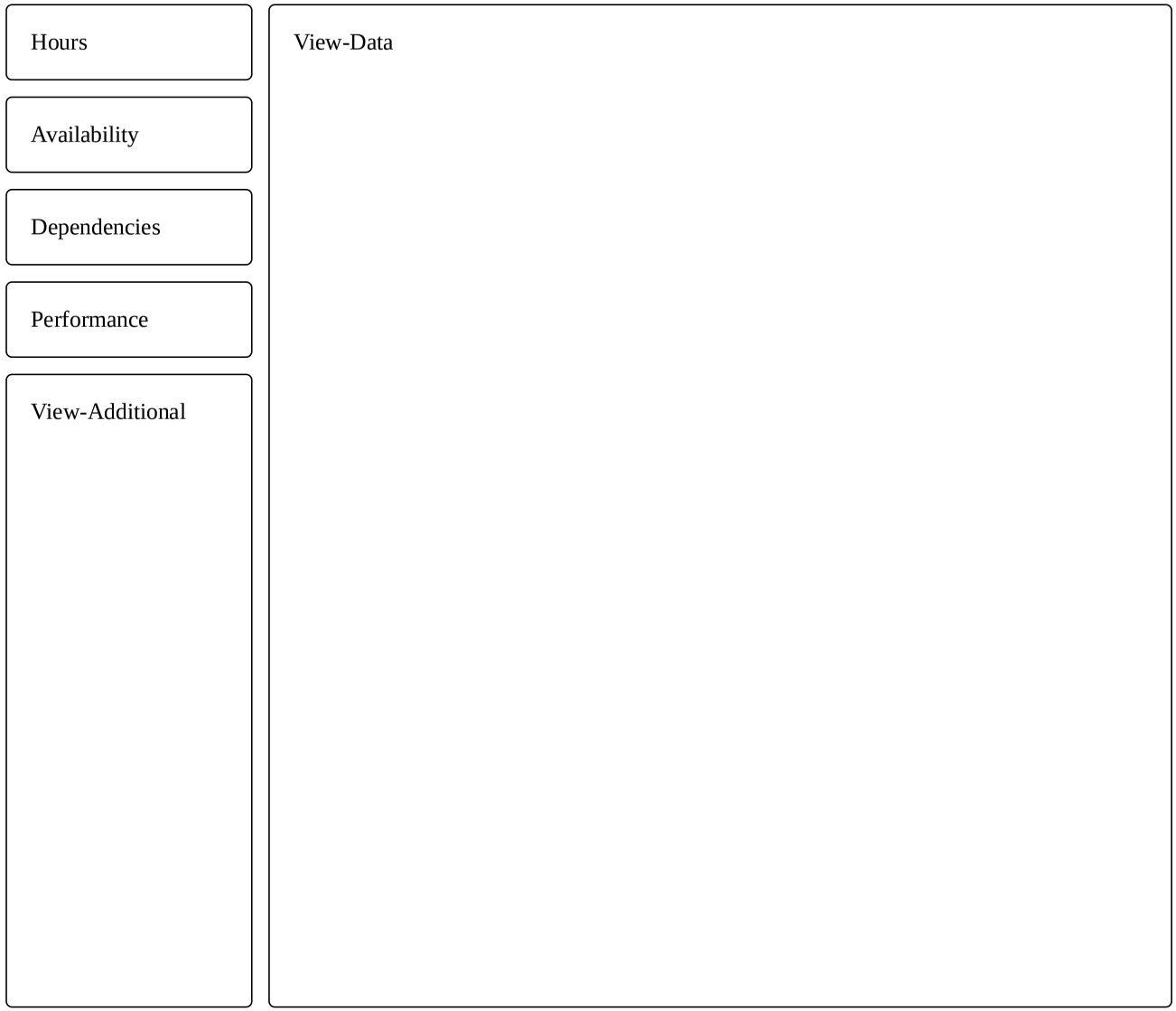


Figure 8.5: Interface draft 1.4

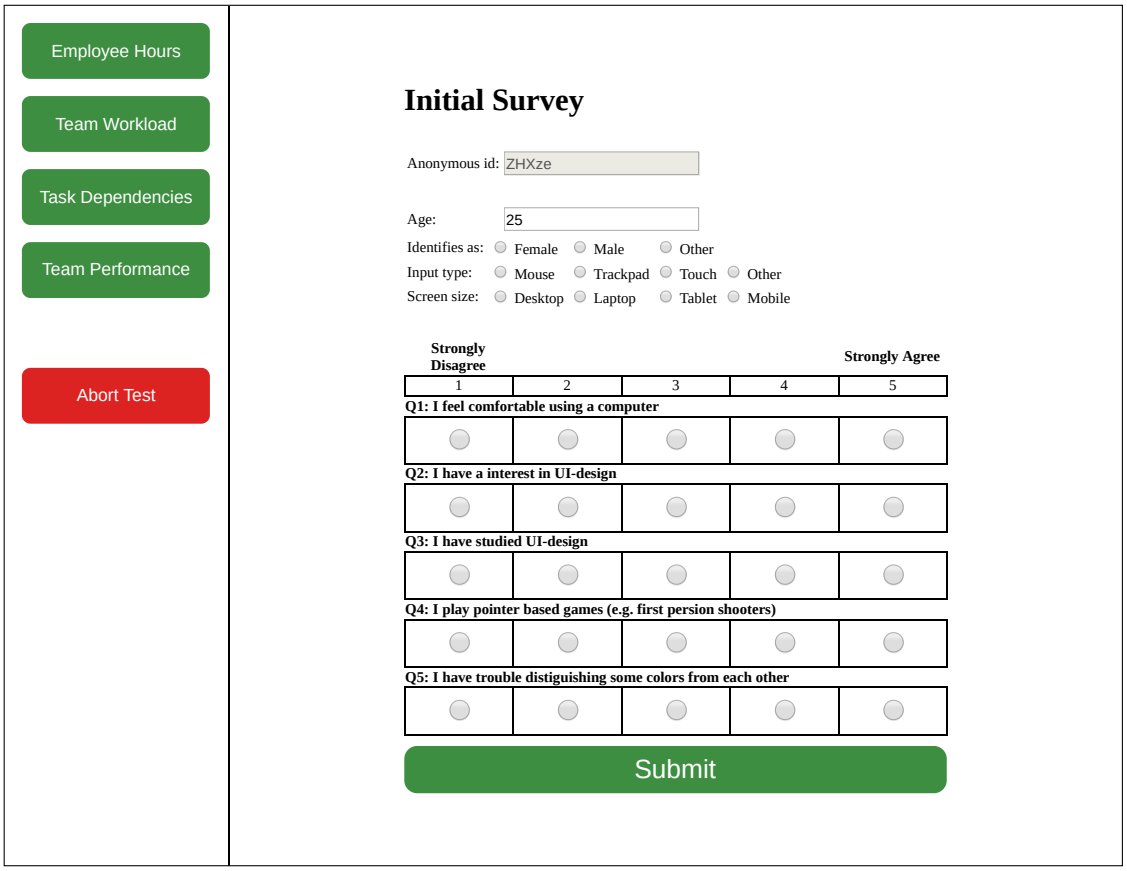


Figure 8.6: Capture of initial survey page.

	Strongly Disagree				Strongly Agree
	1	2	3	4	5
The goal of each task was clear	◯	◯	◯	◯	◯
Test-application looks good	◯	◯	◯	◯	◯
Use of colors helped with the tasks	◯	◯	◯	◯	◯
Amount of information was adequate	◯	◯	◯	◯	◯
Test-application is easy to to navigate	◯	◯	◯	◯	◯
Appropriate choice of colors	◯	◯	◯	◯	◯
Language used was easy to understand	◯	◯	◯	◯	◯
Easy to understand what to do next	◯	◯	◯	◯	◯

Other comments:

Note: Submitting the survey will automatically log you out and return you to the consent form. If you want to continue to do tasks as the current user-id click Cancel below.

Submit Return to tests

Figure 8.7: Capture of the post-survey.