Internship
LIA, Software Test Engineer
40 YHp

Company:

2022

Supervisor:

Martin Pihl

Written by:

Carita Persson

Table of Contents

Abstract

This report is a journey, an internship where I got the opportunity to investigate the test engineering done at Storytel. It was not a clear path with focus on one project and how to work on that in the best way. This report is about technical testing, an analysis of the system architecture. Which tools are used, which areas do the test lead have a vision to improve and what can and cannot be done.

I got an insight in the work of a test lead, how an organization of this size do their test planning, test strategies and handles issues. This report gives an insight in how I concluded to look closer into the subject of agile environment when working with cloud-based microservices.

In a company that are in the start of implementing accessibility to their product, I got to use my knowledge of the topic in a real work project. To write a checklist adjusted for this specific company and the test engineers' specific routines and their level of knowledge was a challenge where I learned more how to improve my test planning for future projects.

We have during the education had courses and lectures about various areas of software testing, strategies, planning, bug reporting. In this report I reflect over the difference from theory to practice. What I learned most at this internship is to connect knowledge and skills to correct task. Before I still had some areas where it was difficult to know how it would be used, how would one know when to use it and what do we do when things is not as we learned.

1. Introduction

Planning the time

Already before the internship started, I had discussions with Martin, the test lead at Storytel. We talked about what could be set as a project during the internship and what would work for the company. Here it was decided that I would not be in a specific team, instead I would work with more technical testing. The main thought was to analyze and set up non-functional test within microservices, with a focus on performance or more so the communication between services.

So, the first day at the office in Lund I knew pretty much what was to expect. When I got to the office, we first went thru how the office was set up, met some of the employees and a brief introduction of how the system with platforms, frameworks etc. is set up.

2. Presentation of the company

Storytel is a subscribed audiobook and e-book streaming service. Their head office is in Stockholm, Sweden and they have 30 offices globally. The company was founded in 2005 and released their first own audiobook in 2011.

There are two areas within the company: Streaming and publishing. The steaming service has its own platform where both audiobooks and e-books are offered to 25 markets. The publishing is carried out from these publishing houses: Norstedts, Massolit, StorySide, Printz Publishing, People’s Press, Rabén & Sjögren, B.Wahlströms, Gummerus, Lind & CO and more.

They have about 700 employees within: People, Enterprise IT, Finance, Management, Content, Customer Engagement, Commercial & Marketing, Legal, Intelligence, Tech, Strategic Projects & Planning, Communication Teams.

All employees are connected to one specific office where they can go and work, but the company have a work from anywhere approach.

3. My time at Storytel

3.1 Thru full internship

The first week went a bit different than expected since the work computer got delayed, this meant that I didn´t have access to anything in the system. So, during the first one and a half week we talked and investigated what areas in the testing that needs improvement. Martin had an idea that I could look at Contract testing. But also made a point that the importance was not a finished project, an analysis and plan for their system would be just as valuable. During this time, I got the opportunity to join when they were doing some bug fixes and implementations in the code. It was a terrific way to get an overview of which tools, frameworks, and some information how their application is built. With my background as a system developer, for me it is easier to understand an application and architecture of a system by looking at the code. There was also a meeting called “Show and Tell” this is a monthly meeting where the employees can do a presentation of what they have been working with, a new feature, an improvement of something. Interesting to join and get more insight to different teams and their work. I would say that these first weeks was full of information, and it felt as I got a clear introduction even without computer and access.

When I got the work computer and access to their platforms, slack channel, and such I continued to look thru their system to gain more detailed knowledge. As mentioned, there was a main thought for this internship, and we had moved to the topic of contract testing. The journey of this internship has not had a clear path, with conclusions made from the research, ideas changed. And it came down to three main themes that I have been working within during these months. I have written about each in the sections below.

3.2 Technical Test Analysis

3.2.1 Degree project

The plan was that I would write my degree project during the internship and the base concept was to investigate best practice and how to set up non-functional testing within microservices. An experiential case study of this specific area of testing. During the time I was analysis their system architecture, test architecture I also talked with different people within the organization. This was test engineers, developers, tech leads and their perspective and opinions of what was of importance in the terms of testing. With these facts and the change of internship project plans I started to lean towards changing the main approach for the degree project. I realized that the problem areas with working agile with cloud-based microservices are far more complex than just the area I had in mind at first.

So, I kept the main question but changed the approach to include: What are the problems within software testing when working agile with cloud-based microservices?

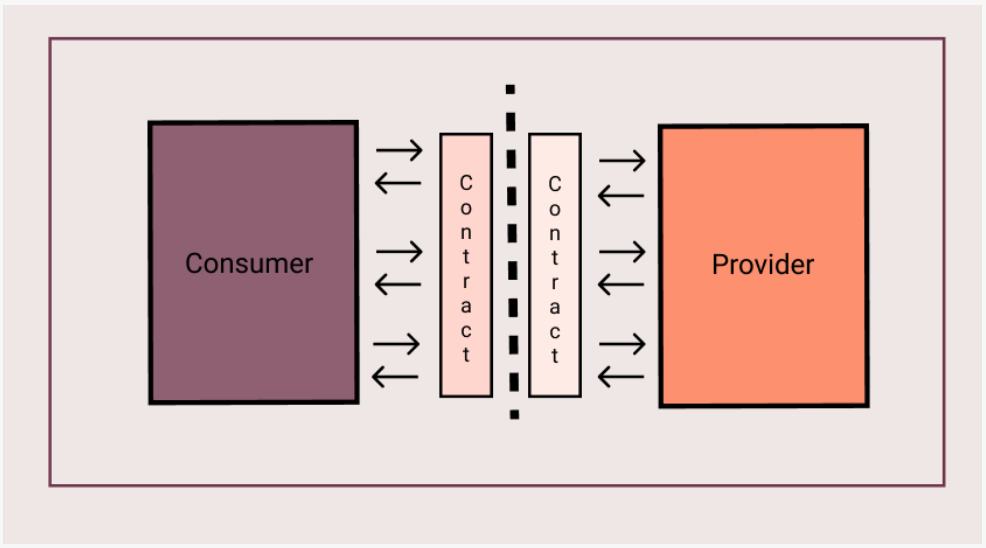
What could be a best practice approach to reach most effective testability, traceability, observability for a microservice environment? And what are the impressions of the priority of testing microservices/container applications among test leads?

The thesis is written in parallel with the internship and this gave me an opportunity to look closer into areas I might not been able to otherwise. I will not go into the technical details or findings about the research, then I recommend reading my finished degree project instead. Below follow some of the areas I investigated during the internship and our technical test analysis.

3.2.2 Contract Testing

As said above we started with the idea to research what was needed for contract testing. There was a developer who had made research on the topic and the test team had been talking about implement it. It was basically just to jump in and get started researching the topic since Contract testing is not something I have done before. I did know about the concept and its basics but not much more. This developer had made a presentation of the work that I looked at, it was directed towards Provider-driven Contract testing and focus on backend. I looked closer into Consumer-driven contract testing. The reason for this was three things: It seemed interesting, and I could see the benefits on focusing on testing from a user’s perspective. Secondly that the company´s product is mobile applications on android and iOS, these are the priority since their website do not have an option to listen/read the books. The third reason was the fact that Martin mentioned the problem with end-to-end test and flaky results. With consumer-driven testing we can both monitor and prevent issues on low-level before it reaches the clients. We test integration points in isolation to ensure the services works together before deployment. I built a small kotlin application just to research this further, more so for my degree project to be able to analyze the coverage we can reach within microservices. I would say that this will probably take some time to implement in a complex system with containers, microservices etc. But the coverage and reliability would be worth it, could save the time it takes to write full integrated test (end-to-end) and avoid mentioned problems.

Here we can see the concept of contract testing:



Consumer test the behavior against provider test doubles, the test doubles are written in a contract. The request sent is replayed against provider API and verified against consumer expectations written in its contract. The contracts should be shared amongst teams, so the teams know provider/consumer expectations and makes collaboration and this type of testing possible.

We had a meeting with the developer mentioned above and during that meeting and after we came upon problems to continue this project. Both the fact that contract testing, both consumer/provider needs to involve communication between teams, access to code and planning of which areas that are critical to involve in the testing. We concluded that it would be too big of a project to do during the time for my internship, still something that would be useful and make an improvement.

3.2.3 Synthetic monitoring

During the discussions mentioned in previous section synthetic monitoring was a topic mentioned and we decided that I should investigate it. There was a tool set up called

Blazemeter/Runscope which was set up with the thought for the teams to do API testing and synthetic monitoring. Almost no team was using it though, and I went on to figure out why and what was used instead. Here I came across different platforms, Postman, Blazemeter, GitHub, Slack, New Relic and GCP. With agile teams and they decide within the team what tools to use it was at times difficult to find the right person to ask questions about this, or to find the person who could give access to a tool/software.

3.2.4 Tool Sprawl

Here I analyzed the reasons of why some specific tools was used by some teams. All different tools used and what limits they had. Here I went thru lots of documentation about different tools and software, both the ones used and the ones that could be integrated. From the answers about the reason for tool sprawl was the fact that some tools couldn´t do what the team needed or that they already used another when some of the other was implemented.

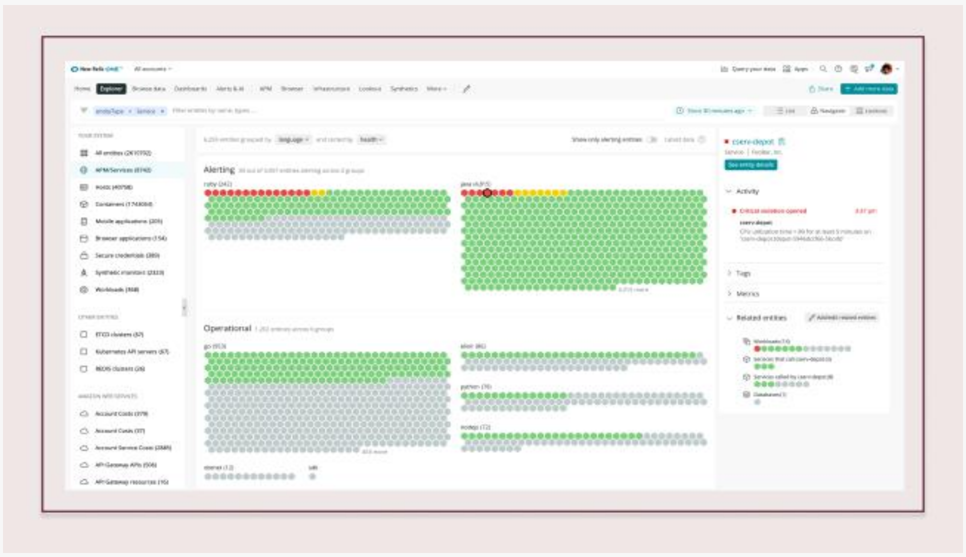
For example, GCP, includes a lot of services for almost every area needed in cloud-based microservices/container development and testing. But after researching the PaaS closer there is no possibility to set up synthetic monitoring, not within GCP, not with third-party software. There are also limitations in their monitoring service when it comes to customized dashboards.

3.2.5 Monitoring, Tracing, Observing

Customized dashboards were another topic we talked about during different conversations. To be able to monitor the system performance, both high and low-level. In combination with smooth traceability. Today there are a lot of log files but no structure or documentation that the testers can use to work effectively with the data. With all these different topics collected during the research I started to look if there was any tool that had all these included. If it could be a way to minimize tool sprawl, erase flaky tests, make the test engineers have a better overview over various parts and of the full system. There was one tool that already had been used within the company, I say had since it was no longer actively used. New Relic, here I looked thru their documentation, looked at all their courses to get a deeper understanding. I did not have access to the tool, so all this was done while waiting to find the right person responsible. New Relic includes diverse ways to set up the integration between the tool and main platform, like GCP.

They have experimental courses on their website that I went thru and built a tutorial app to try out different features, some things were: build custom dashboards with NRQL (new relics SQL), measuring performance, alert different critical conditions with Alert Quality Management, how to trace alerts, and explore data with full-stack observability.

Here is an example from their website:



(Visual view of observability navigator, from New Relics website)

It is their navigator, here we can filter views by entities and see the full system health. The colors make it easier to find issues. There are several other features in this tool that makes it interesting and useful when it comes to collect data at one place.

This diagram shows the infrastructure:

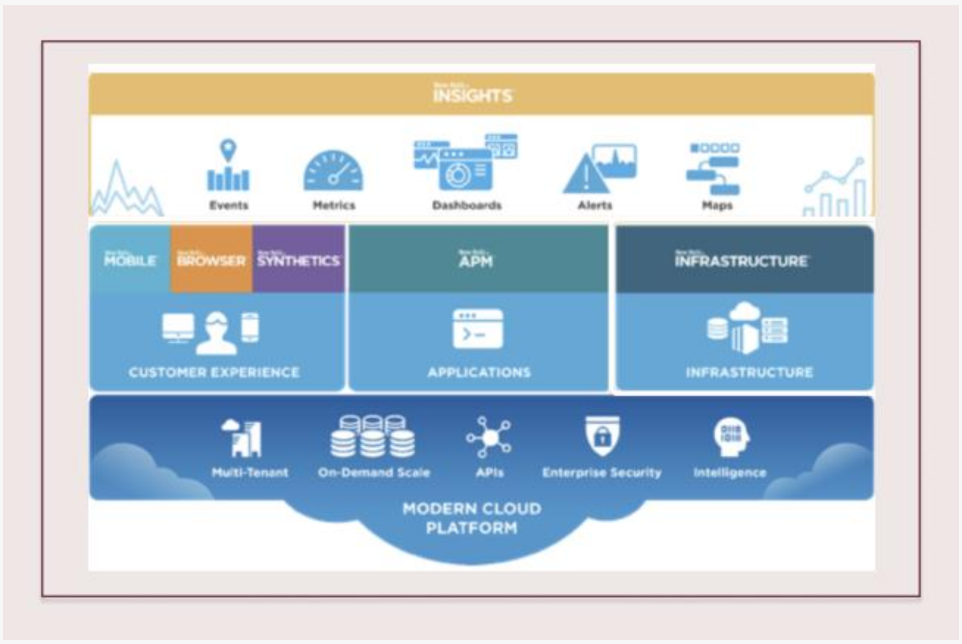


Photo from New Relics website, blogs

After gone thru the documentation and investigated how Storytel had integrate New Relic with GCP. This led to the same problem as before that it was integrated thru the code, which means that I could not do any set up of monitoring dashboards that was in the test leads vision for the future testing structure. It would not be enough time to learn both the way to integrate and to set up the boards. We also had a discussion with the people responsible for it, and they had just started to organize a way to set up new relic in a way that could benefit the teams. So again, we concluded that the project idea was not possible. But I learned a lot from this, new monitoring techniques I was not familiar with before. The structure and how to work with the features in GCP, I have taken some courses about GCP earlier from my own interest in the topic, but here I was able to access and research it closer in a full working set up. With my earlier experience working with Azure, it was interesting to be able to compare the differences and see pros and cons about both. And it gave me a new area to research and learn more about. This is a topic I will continue studying.

3.3 Accessibility

I have a specialized knowledge and large interest in digital accessibility and Storytel have just started to educate the employees on this topic. They are working towards making their products accessible and approve of the EN 301 549 European accessibility law.

One main problem with WCAG and EN 301 549 is that much of the guidelines are directed towards web, not mobile. After all the current guidelines was written in 2008, smartphones barely existed back then. The test team talked about this in their meetings and that they sometimes don´t know where to start and how to translate the success criteria to mobile. Therefor me and Martin talked about it and decided that I would write a checklist, focused on mobile and minimalized to not become too overwhelming to use as a start.

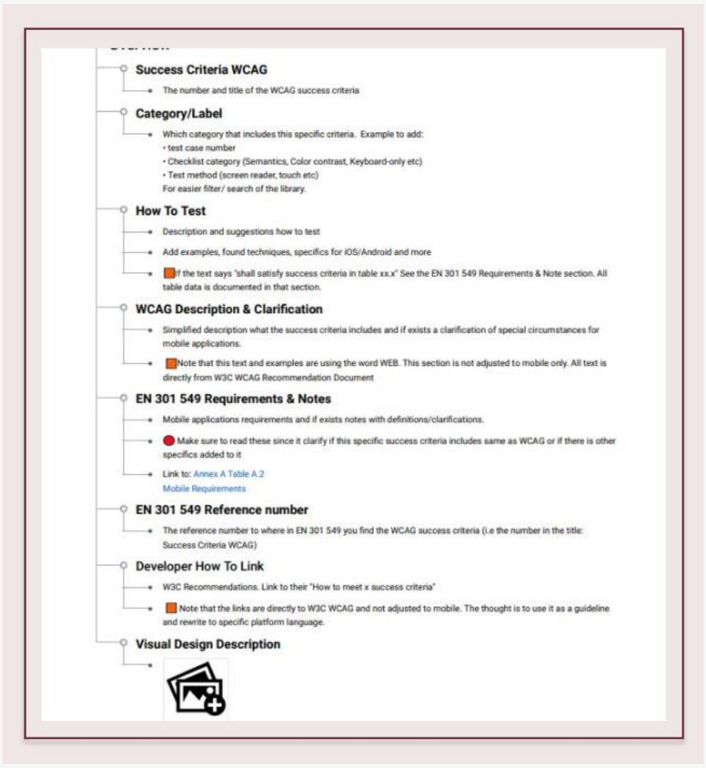
The first thing I did here was a one-page checklist, where I made 12 categories:

- Color/Contrast
- Context changes / Dynamic Content
- Custom controls
- Form Fields and Labels
- Form Errors
- Images
- Keyboard and Touchscreen Navigation
- Links & Buttons
- Readability
- Semantics and Structure
- Tables
- Timing

Then just the basics of what each category includes, it is kept simple to be easy to use.

Storytel had hired a company that do courses a lecture to educate about accessibility and had material from that. This material was in 100+ pages in PowerPoint format. So, my next thing was to collect this material, connect it with WCAG success criteria, and then connect that to the correct section in the EN 301 549, and add a section how to test. Even for me who have good knowledge on this topic and the details it is a challenge to write it. And something I never done before is to see this from the perspective of testing, and even less mobile only. I learned a lot of new things writing this and got a deeper understanding for the testing.
This project is still not completely done, I wish I could have finished it before the internship was over, but the time was too short. Most things are done, all success criteria and categories seen in picture below are done, except no images are added. I also been in contact with the UX-designer, and they are working on similar project and hopefully this is something they

can use in the future and add images for even more clarity. Here is a photo of the Overview document, I made this document so it will be easier to understand where to find all information in the main Accessibility Reference Library:



Success Criteria WCAG
- The number and title of the WCAG success criteria

Category/Label
- Which category that includes this specific criteria. Example to add:
  • test case number
  • Checklist category (Semantics, Color contrast, Keyboard-only etc)
  • Test method (screen reader, touch etc)
  For easier filter/ search of the library.

How To Test
- Description and suggestions how to test
- Add examples, found techniques, specifics for iOS/Android and more
- 🟧If the text says "shall satisfy success criteria in table xx.x" See the EN 301 549 Requirements & Note section. All table data is documented in that section.

WCAG Description & Clarification
- Simplified description what the success criteria includes and if exists a clarification of special circumstances for mobile applications.
- 🟧Note that this text and examples are using the word WEB. This section is not adjusted to mobile only. All text is directly from W3C WCAG Recommendation Document

EN 301 549 Requirements & Notes
- Mobile applications requirements and if exists notes with definitions/clarifications.
- 🔴 Make sure to read these since it clarify if this specific success criteria includes same as WCAG or if there is other specifics added to it
- Link to: Annex A Table A.2
  Mobile Requirements

EN 301 549 Reference number
- The reference number to where in EN 301 549 you find the WCAG success criteria (i.e the number in the title: Success Criteria WCAG)

Developer How To Link
- W3C Recommendations. Link to their "How to meet x success criteria"
- 🟧 Note that the links are directly to W3C WCAG and not adjusted to mobile. The thought is to use it as a guideline and rewrite to specific platform language.

Visual Design Description

The documents were written in a program called Delibr, never used it before but Storytel uses it and to avoid using a new tool I decided to go for this one. Also, since it is not possible to create foldable sections in google docs, this was the best option to keep the large document usable.

While writing this the winter holiday was coming and this meant well yes vacation time for many, but also new app version release. Martin asked me if I wanted to be involved in the testing of the app. No way that I wanted to miss that opportunity to gain experience, so the accessibility document was on hold for a time, and I finished just the last days of my internship.

3.4 App release

To get started with the app testing I got some test cases added to start working on. This is done in Jira; I do know a bit about Jira since before but never worked in it for real projects. So, even if I felt a bit lost at first or maybe mostly nervous to not do anything wrong that could cause problems, this was interesting and informative. I did test of both the android app and iOS on iPad. All this process was new experience, to set up TestFlight for iOS and internal testing for android. I have done app testing on android before, both others and my own but that has been connected to the computer, not internal test. To see the full process of testing:

- From test board

- in Jira you have the option to only see the test cases signed to you.
- Start the executing and do step by step test
- Some included test different feature with SQL in database.
- If an issue is found: First read the expected results to see if this is already known but not a priority for now.
- Look thru the reported issues and either connect a bug to already existing issue or
- create a new bug report.

Follow it all in different tools, Jira, GCP, Big Query, email, Slack. During this time, I had impressive help from some test engineers, they took their time to help when needed and answered questions along the way. I have to say that it truly shows how great of a company Storytel is. Their work from anywhere approach, which means everyone is working remote (at least now in time of covid) do not impact their communication. People are friendly, helpful, and structured in their work. This experience is something I value high, it shows that people are professional, care about their work and their colleagues. Some of the issues I found, (which was not that many I must say) was then assigned by developers to solve. One of those issues was still ongoing when my last day was coming up, I had been following the process and the discussion around it and felt a bit like not reading the end of a book, you want to read the rest. It is interesting to see the process happening and how testers/developers and developers in different teams communicate and work together with it.

4. Discussion

The main project was not possible to continue with, and all the similar project that came up along the way had the same result. This is correct at least from the perspective of having a finished project to present after the internship. At first this felt like a failure, a disappointment. The reason for this is that I always want to do my best and be throughout with my performance. To not really be able to show a result of the analysis and all the research with something that shows that the thoughts will improve the issues discussed.

There were a lot of waiting time to get access to systems, to get answers or even find the right person we were looking for. Some days before I started to focus on the accessibility there were mostly waiting and since I didn´t work within any team or had contact with anyone I couldn´t figure out or look if there was any other way or something else, I could do. This was also before I attended any tester meetings, so I learned most of the organizations architecture and system/work setup by reading thru all the documents, checklists, onboarding instructions from the different teams. Then again, I did base my degree project on this topic, in a way and learned a lot by all the back and forth. It is another knowledge learned and it did give room for interesting discussions that gave insight in both testing and development.

When I started to write my degree project, I did realize how much I had learned and perspectives I did not have before. Me and Martin had daily scrum morning meetings and those was helpful both to discuss how to move forward, to learn more about his job as test lead and to see the reality he is facing with agile teams and its architecture. The differences from the lectures in the education about test planning, test strategies are significant in some areas. The analysis of system architecture and tools was interesting, some or most I have used or come across before. The difference is the set up and usage of some software, also including the testing/monitoring that is directed towards other than API. All that was new to me, Blazemeter/Runscope, New Relic. GCP was new to use too, I have worked with Azure and AWS so had a rather good idea how it would work but to get the opportunity to study it closer running was something I now feel will be useful in future work. So, my conclusion of this part of the internship is that it was inspiring, it has given me motivation to continue study the topic and learn how to use it in my test engineer profession.

The testing of the app was not planned at all, and I jumped right into it the same day we talked about it. Since I had read all the documents before I knew somewhat how the structure was supposed to be. At first, I got two test cases assigned to me, and the tester who assigned them explained some basics. At first, I caught myself to look thru the test case and making a mental checklist based upon the school lectures. Like, ok do this follow the "schoolbook example" is everything there that we have learned. For myself this says quite a bit of theory vs reality, but mostly how much I learned thru the years of this education. The test cases are written well and most of them that need a step-by-step instruction do have it, with good explained expected results. These first test cases I decided to check my knowledge to connect the test methods, techniques we have learned with the specific step in the test case. Those things were only for me to use the opportunity for better understanding.

One not an issue, issue, doing these tests as a book nerd is that I came across too many interesting books and my bookshelf is now filled up with even more books to read. I always enjoyed books and that made it easier to connect and care for the product. To work with a product, you use yourself and involves a personal interest is motivational and maybe even makes one want to perform the work even better.

I wouldn´t say that the task of executing the test were where I learned the most, it was the full process of it. To see how the test captain and the testers assigned the test cases, the amount of test needed, priority of them. Then the differences in issues between platforms, even as a developer I learned here. Then the conversations both that I had and others in the slack channel for the testing were another useful part to learn from. The communication, fast responses and respectful interaction and understanding when someone is new and not familiar with something. To follow the process from start and see how next release was done, the adjustments, how the test team structure the task of test captain, the communication in the test meetings and more.

5. Conclusions

Sitting down now reflecting over these months at Storytel I must say that I could not have gotten a better opportunity to do my internship. The number of new things I have learned and in so many areas of testing. I am grateful that I got to work with different topics, not at all as expected. But in the end, I would say expect the unexpected is a part of being a software test engineer. It is what we do with what we are facing that is of importance.

I had knowledge and felt pretty secure in this role of profession before the internship, with a humble understanding that I have much more to learn of course. But the conversations, the knowledge shared and the employee's treatment of new colleagues or even "just the intern" made it easy to feel comfortable and motivated on a whole different level. I was at the office in Lund for a few weeks, and some uneven days but since late December the offices closed completely because of the new Covid-recommendations. (Which only that is another reason to appreciate this company, they care about their employees being healthy, to respect the healthcare system, and as I see it, trust their employees to be responsible and skilled enough to do their work from anywhere.) Anyways, the offices being closed have not changed anything in that feeling of comfort and motivation. And the last but absolutely not least is the fantastic positivity that is among the employees, even in a time where there are issues, and one would expect stress and maybe even short pessimistic thoughts. I know that I will look back on this experience in future work, to use all new knowledge and know how inspiring the test engineering profession can be.

6. Syllabus

6.1 Requirements for this course:

After completed course, the student must have knowledge of/ of:

1. How the software testing is handled at the internship workplace

Even with not being in a specific team and follow their everyday work, all the different tasks I have done, and the test meetings have given an insight in Storytel´s testing approach. The onboarding documents gives an insight in all tools used, and I used most of them in one way or another. To follow Martin´s work and see what a test leads work can involve, decisions and planning gave another insight in not only his work but in some of the organizational structure for tech, development as well.

To attend the weekly test-club meetings and listen to their conversations about various parts of their testing also involved test engineers from different teams, different platforms, and some variety in focus areas. I would say that the only area of testing that was not included in some way was security testing. Although, I got some insight in how this could be handled too, in case of risks. Nothing that was affected but we all have heard of the Log4j incident, and probably every single company with some sort of security mind mentioned this. Not specifically testing but still interesting and worth to mention.

The accessibility was another interesting insight, both for me personally and professionally. They are just as many others in the starting point of making their application more accessible and I have not gotten the opportunity to see this phase in real before. I have studied it, and in my specialization to Accessibility specialist the part of test planning and test strategies is included. But as with everything else, theory and real work is not the same thing.

App version release was probably the part that involved most of the topics we have gone thru during the courses, test execution, test cases, bug reports, think as a tester. The part I would say I did not get detailed insight in was the planning within the teams. But that part was discussed in the weekly meetings and app release meetings.

2.Report writing

Report writing as in degree project, internship report. Yes, that I have done. The accessibility reference library might somewhat be a report, or no. It is high level strategy more so. None of the projects I have been working on has included to write a full report in my definition of report. Bug reports I wrote a few when something was found.

3. How best-practice is in theory not always applicable in projects due to different project constraints such as time, budget, and staff

This one I would say I aced in experience with the analysis and outcome of performance, contract testing and monitoring tools project idea. I write that with a bit of humor, but I genuinely mean it, that it was not applicable du to constraints, it is an experience I am grateful to have. I learned so much from it and to investigate all those various parts of the system not only gave me knowledge of this specific company´s work, but it also gave me things to use in the future and motivation to continue my research of the topics.

Another part that I think fits here is how to write test cases, some test cases are just as “by the book” and some adjusted to the specific test case.

After completed course, the student must have skills in:

4. Work with software testing according to the internship workplace´s guidelines

During the testing of the apps, I had conversations with some test engineers in different teams, they shared best practices and protocol guidelines they use. What and what not to do in different situations.

5. Writing bug reports

Already mentioned this above, the most challenging thing here I would say was to understand when and when not do certain things. Testers have their own way to work and explain things differently. This made it confusing to know what could be seen as something that should be a new bug report or not. My best knowledge gained here except from the obvious of being meticulous and put pride to your work, is to ask when you are unsure of something, no one will see that as a weakness.

6. Perform tasks at different phases of the test process

Analysis of test architecture, writing checklist to start accessibility testing, be active at an app version release and in the end of the process.

After completed course, the student must have the skills to:

7. Independently reflect on selected test methods used in completed projects at the internship company.

Reflecting during the process of all the different tasks done during the internship is the one thing I done the most. Both with the thought of my degree project and to try to connect the tasks with what we have talked about in the lectures in different courses. That is what has given the realization of how much we have learned during the education and how valuable it has been to get the perspectives and opinions from teachers and the different test engineers active in the field that have visit and done their presentations of how they work with testing.