Part1:

Good afternoon, everyone. My name is Zhou Zhiqiang and I’m very delighted to have the opportunity to **present** my internship work here. My presentation is in four parts. Firstly, I’ll introduce the background and key metrics in the dashboard. Secondly, a demonstration is given. Thirdly, I’ll talk about the workflow behind the dashboard. And finally, Q & A. Please feel free to interrupt me whenever you have any question.

Let’s start with a story. Suppose I’m an engineer in our insight team, this year the goal of our team is to attract more people to make contributions on Docs website and Alex lets me take charge of this task.

**Firstly,** take a look at this funnel. The absolute number makes not much sense here because with more topics published, the absolute number will be larger. Here we care more about the percentage. We can see that **in July**, the percentage of people who click the contribution button is 0.23% and the percentage of people who make a pull request is 1.70%, which are both relatively low.

So for me, how to increase the percentage?

Actually I have brought up some hypotheses, for example, we may give the people who make contributions some awards. It sounds reasonable but how to verify its feasibility. Currently, we lack an integrated UI hub to present the contribution situation. Thus a dashboard is made to help me judge whether a hypothesis works or not.

I’d like to introduce the key metrics in dashboard briefly: the publish here where people can modify one or more topics. And the publish is mainly accomplished by a pull request, so the pull request should be considered and a pull request will in turn cause a contribution in GitHub concept, the contribution and contributor are included.

Part2:

So Let’s turn to the dashboard. This is the overall view of the dashboard, which can be **divided** into four parts: the publish, pull Request, contribution & contributor and the final part, here on top. All the items are categorized by repository, site and tenant. And a time selector is used here to configure the time range.

For the three key metrics, the per day trend or per month trend and the total number are shown. **Suppose** that I have implemented an A-B testing, if the hypothesis works, I will see an obvious increase in these charts. As to publish, if I want more detailed information, I can look at these four charts which are **further categorizations** for publish.

For example, I can see the statistics of how many lines or words inserted or deleted in a topic in these two charts. **Look at** this pie graph, it illustrates the percentage of actions on topics: add, delete, edit or rename. And this bar chart shows the distributions of the number of topics per publish.

As I said just now, the publish is mainly accomplished by a pull request. I’d like to know the behaviors of the pull requests. Then I will turn to these three charts where the pull request is categorized by its status, the number of commits in a single pull request and the duration of pull request from created to merged.

For time reason, I won’t introduce these charts in details. You can know more about them by fetching the document whose link is below.

What’s more, by selecting the repo list, I can compare the performance of repos internally. If some certain repo performs much better than others, I can conversely look into this repo to find the reason why and make further hypothesis. Similarly, I can compare between tenants and sites by selecting the site list and tenant list.

And look at this time selector, for example, the Docs website is open to public in May, so I can drag it to see the data after May.

Part3:

To sum up, the dashboard offers a unified place to present the overall contribution situation and can help to judge whether a hypothesis is feasible and efficient or not, or to make further hypothesis.

Then I’d like to move on to the workflow behind the dashboard.

Firstly, we need to acquire the JSON data from GitHub and VSO by calling REST API. Secondly, the JSON data will be parsed. Thirdly, loading the information to database and finally using PowerBI to visualize the data.

So what are the challenges in this workflow? For me, the main work is to gather the data that is missing or incomplete previously. For example, as to pull request, all data is for GitHub repositories and I need to investigate how to fetch data from visual studio online for VSO repositories.

Do you still remember the changed lines and words in dashboard? Actually, GitHub only offers us the data of changed lines but as the topics are usually documents, the word change will be more accurate and reasonable. Thus, I need to design an algorithm to compare the changed topics before and after a publish, which is relatively complex.

Finally, for topic localization, the challenge lies in the complicated business logic, different repositories may have different naming rules and different ways of localizations. I have to design filters and patterns for repositories to handle this task.

Well, that’s all for my presentation. Thank you for listening and sincere thanks for Alex and Nanxuan ‘s hard work and helps in my internship. Any question?

Part4:

Thank you (for asking).

I hope this answers your question.

However, we don’t have any figures on that, so I can’t give you an accurate answer.

I’ll try to get back to you later.

Firstly, I need to get the difference between the topics before and after a publish. Because the majority part of the document remains the same, if we directly compare the whole topic, it wastes quite a lot of time.