Entrevista 05 - Entrevistado

Atualmente, eu sou desenvolvedor pleno web. A gente tem algumas plataformas voltadas ao público de forma externa, onde o cliente tem acesso, sem ser dentro da plataforma, ele consegue acessar como fonte de guia e, como posso falar, a documentação para estar utilizando o nosso serviço. E, além dessa, também tem outras aplicações, já dentro de um sistema maior, que o usuário precisa estar logado para fazer as transações que ele necessitar.

Então, tanto essa parte logada diretamente para o cliente, quanto para o usuário que está buscando a documentação de como utilizar o serviço. Bom, eu não tenho bem essa quantidade de projetos determinada. Eu faço parte de uma tribo que a gente atende a muitas das outras equipes da empresa.

Então, é uma equipe meio que de suporte a outras equipes. As outras equipes têm as demandas que são centralizadas na gente. Então, projetos mesmo da nossa equipe, eu estou em quatro, mas eles rodam em paralelo e a qualquer momento pode chegar coisa de outra equipe também para estar dando alguma força.

No back-end, nós temos quatro e de front apenas eu. Dentro da minha squad, são esses. A minha tribo possui três squads e a tribo, no geral, tem cerca de 22 desenvolvedores, sendo boa parte deles voltadas ao back-end.

Sim, sim. Basicamente, nossas demandas, a gente segue um quadro de atividades. Então, as atividades chegam, são quebradas lá no back-log, são priorizadas e dentro da plataforma do GitLab, cada atividade daquela vai ter um código.

Então, sempre que eu precisar ter alguma alteração, a gente cria um abrinte a partir daquele código do GitLab. O de tudo vai estar atrelado lá, qualquer pipe que a gente rodar, qualquer coisa vai estar atrelado àquela atividade, àquela task lá. E cada alteração no código costuma ser bem pontual mesmo, a partir de cada task.

Então, cada task lá vai gerar aquele código lá que a gente vai passar para abrinte, vai fazer as alterações em cima daquela abrinte, daí ele vai seguir a esteira lá de quem jogou promulgação, do que está para teste, até finalmente ser mergeado no código, que é quando ele está pronto para ir já para a produção. Mas antes de ir para a produção, a gente mantém essa abrinte à parte, faz toda a parte de teste, se tiver alguma alteração em cima daquilo, para finalmente, quando for aprovado, que é mergeado, que é feito o merge. E a task é finalizada.

Não, sempre costuma ser a mesma questão do GitLab ou GitHub, mas foi muito pouco, mas é na base do Git também. E a diferença que tinha mais de outras equipes, de outra empresa que eu participei, é que na primeira empresa, cada desenvolvedor tinha um fork do projeto. Então, existia o projeto Py, digamos, que era da empresa, e cada

desenvolvedor, enquanto colaborador, tinha um fork daquele projeto, e a gente podia fazer nossas brindes, nossas alterações dentro do nosso fork, e uma vez que a gente finalizava as atividades, a gente mandava para o repositório original, repositório Py, mas seguindo uma esteira bem parecida de, como eu posso dizer, de branch com o código da atividade.

E daí também tem toda a questão de precisar da atenção do rebase, do Fetch, do rebase com o repositório Py, para sempre estar pegando o código atualizado. Porque como tem outros desenvolvedores mexendo, de repente pode dar algum conflito, então a gente sempre tenta manter atualizada a origem. Isso.

Geralmente, quando é alguma coisa mais pontual, é direto. Agora, quando é uma demanda nova mesmo, chega uma funcionalidade nova, costuma ter também a parte do desenvolvimento de teste, não apenas o código em si, mas costuma ter o teste também. Também tem a validação do link, a gente tem algumas bibliotecas para estar validando o link, e outra ferramenta também, que ela vai procurar no código, de repente alguma coisa que pode estar, como eu posso falar, desconforme com outra tecnologia, com outra versão de navegador.

Até nome de variável também, ele pode dar alguma crítica, alguma sugestão para a gente estar alterando, variáveis que não estão sendo utilizadas, e métodos que estão muito longos, essa ferramenta vai indicando esses pontos. Bom, a gente utiliza ele bastante porque cada teste, cada atividade, cada demanda, cada problema, cada bug, ele fica bem separado, a gente consegue estar separando bastante o que vai estar entregando em cada branch, então se precisar voltar o código, alguma coisa quebrou, a gente tem aquele commit ali, aquela branch, aquele merge bem separado, não vai ter nada, como eu posso falar, tipo, não posso voltar o código todo porque tem uma coisa que funcionou nele e vai ficar, então sempre tem códigos separados para branches separadas e demandas separadas. E cada vez que a gente faz essa alteração que está para subir, a gente acaba gerando tags, então existem pipes diferentes que vão estar rodando para cada comentário que a gente colocar na tag, para cada, posso falar, tem algumas palavras-chave que a gente usa na tag que vai estar criando ali, vai estar chamando a pipe, seja a pipe de homologação, de produção, então vai ter ali aquele ambiente e daí tudo fica gerenciado pelas tags lá, as tags são sequenciais, basicamente, o vacionamento ocorre dessa forma e a gente consegue estar vindo ali, qual foi a versão que deu algum problema, se for para voltar também, a gente consegue voltar fácil para uma versão que estava funcionando e tem acesso.

A questão que desfavorece é que, às vezes, eu acho que coisas bem pontuais, acertos mais pequenos, acabam precisando burocratizar um pouco para entrar nesse processo, às vezes tem uma coisa mais pontual que a gente acaba precisando criar um card até para ficar documentado, eu entendo esse ponto, mas acaba meio que burocratizando demais para um acerto pequeno, coisa pequena, às vezes, um acordo de determinada

coisa, um ponto final, uma frase, ou qualquer coisa assim, a gente precisa estar passando essa esteira toda novamente para estar validando, acho que aí fica meio burocrático. Apesar dessa dificuldade de coisas mais pontuais, eu acho que ele vem funcionando bem assim, a gente tem bastante estabilidade e segurança no que faz, porque, como eu falei, eu trabalho, tenho os meus projetos, mas tenho projetos de outras equipes também, além de interagir com outras squads, pode chegar gente de outros times também, então a chance de a gente interagir com outro grupo, outro projeto é muito alta, e como todo mundo tem a mesma mentalidade, o mesmo processo, acaba sendo fácil a gente estar ajudando algum projeto, até um dia que migrar a equipe dentro da mesma empresa, a gente consegue pegar fácil, pegar rápido a lógica do time. Hoje a gente já está estabelecido no mercado, uma empresa do ramo financeiro, um banco digital, a gente já tem uma certa estabilidade e segurança no mercado, uma base consolidada de clientes.

Não, não, até porque na parte do front-end, a frente na qual eu mexo, eu acabo não esbarrando muito com outros desenvolvedores, de repente a parte do code review, que a gente costuma fazer entre equipes, mesmo não sendo daquele projeto diretamente, mas para estar encontrando alguma coisa que possa estar melhorando no código ali, aí é quando a gente acaba discutindo alguns pontos, mas conflito de código não é comum acontecer não. Não, acho que foi uma entrevista interessante, deu para cobrir bastantes pontos, foi legal. De repente, mas apenas como perguntasse sobre o versionamento, sobre como quando der algum problema, se você tem a segurança de voltar aquele, fazer um rollback daquele determinado ponto, acho que seria interessante também.

Obrigado.