### A seguir algumas ferramentas que vão compor minha stack de desenvolvimento, para casa fase, considerando suas funcionalidades, eficiência e relevância para o ciclo de vida do desenvolvimento de software. É importante observar que, ao avaliar essas ferramentas, foi levado em consideração que a implementação de algumas delas pode requerer investimento em assinaturas ou licenças, ou seja, será necessário investimento financeiro. A decisão de adoção dessas ferramentas não apenas considerou a excelência em desempenho e capacidades, mas também a viabilidade econômica em relação aos benefícios proporcionados. **Colaboração:**

* **Comunicação: Teams**
  + **Justificativa:** O Slack é amplamente utilizado para comunicação em equipe devido à sua capacidade de criar canais temáticos, integração com diversas ferramentas, compartilhamento de arquivos e facilidade de comunicação em tempo real.
* **Colaboração em Documentos: Confluence**
  + **Justificativa:** O Confluence é ideal para controle de documentação no desenvolvimento por sua colaboração eficiente, centralização de informações, integração com ferramentas como Jira, rastreabilidade, flexibilidade, facilidade de busca, documentação em tempo real e suporte a fluxos de trabalho. Essas características promovem eficiência e consistência na gestão da documentação.
* **Gerenciamento de Projetos: Jira**
  + **Justificativa:** O Jira é uma ferramenta robusta para o rastreamento de tarefas, gerenciamento de projetos e acompanhamento do progresso. Ele é altamente configurável e se integra bem com outras ferramentas de desenvolvimento, possibilitando o uso de quadros Kamban, utilizando como metodologia ágil.

### **Build:**

* **Controle de Versão: Git (GitHub)**
  + **Justificativa:** O Git é amplamente adotado para controle de versão devido à sua eficiência e flexibilidade. O GitHub fornece uma plataforma colaborativa para hospedar repositórios Git.
* **Integração Contínua: Jenkins**
  + **Justificativa:** Jenkins é uma ferramenta de integração contínua robusta e altamente configurável. Permite a automação de compilação, testes e implantação, melhorando a eficiência do processo de desenvolvimento.

### **Teste:**

* **Testes Automatizados: Selenium (para testes de interface) e JUnit (para testes unitários)**
  + **Justificativa:** O Selenium é poderoso para testes de interface de usuário, enquanto o JUnit é amplamente utilizado para testes unitários em Java. Ambos são bem estabelecidos e suportam automação eficiente de testes.
* **Gestão de Testes: TestRail**
  + **Justificativa:** TestRail é uma ferramenta eficaz para gerenciar casos de teste, planejamento de testes e rastreamento de resultados. Integra-se bem com outras ferramentas de desenvolvimento.

### **Implantação:**

* **Orquestração de Contêineres: Kubernetes**
  + **Justificativa:** Kubernetes é uma solução robusta para orquestração de contêineres, facilitando a implantação, escalabilidade e gerenciamento de aplicações em contêineres.
* **Ferramenta de Implantação Contínua: Ansible**
  + **Justificativa:** Ansible é uma ferramenta de automação que pode ser usada para configurar, implantar e gerenciar infraestrutura de maneira eficiente e consistente.

### **Execução:**

* **Monitoramento: Prometheus (para métricas) e ELK Stack (para logs)**
  + **Justificativa:** Prometheus é eficaz para coleta e consulta de métricas, enquanto o ELK Stack (Elasticsearch, Logstash, Kibana) é excelente para processamento e visualização de logs.
* **Orquestração de Serviços: Docker Compose**
  + **Justificativa:** Docker Compose facilita a definição e execução de aplicativos Docker multi-container, simplificando o gerenciamento de serviços em um ambiente de desenvolvimento e produção.