Technology Radar

Um guia com opiniões firmes sobre as fronteiras de tecnologia

/thoughtworks

Volume 28

Sobre o Radar

Thoughtworkers são pessoas apaixonadas por tecnologia. Nós desenvolvemos, pesquisamos, testamos, contribuímos com código livre, escrevemos sobre e visamos a sua constante melhoria — para todas as pessoas. Nossa missão é liderar e promover a excelência de software e revolucionar a indústria de TI. Nós criamos e compartilhamos o Technology Radar da ThoughtWorks para apoiar essa missão. O Conselho Consultivo de Tecnologia (Technology Advisory Board, ou TAB), um grupo de líderes experientes em tecnologia da ThoughtWorks, é responsável por criar o Radar. O grupo se reúne regularmente para discutir a estratégia global de tecnologia da empresa e as tendências tecnológicas que impactam significativamente a nossa indústria.

O Radar captura o resultado das discussões do TAB em um formato que procura oferecer valor a uma ampla gama de indivíduos interessados, de pessoas que desenvolvem software a CTOs. O conteúdo é concebido para ser um resumo conciso.

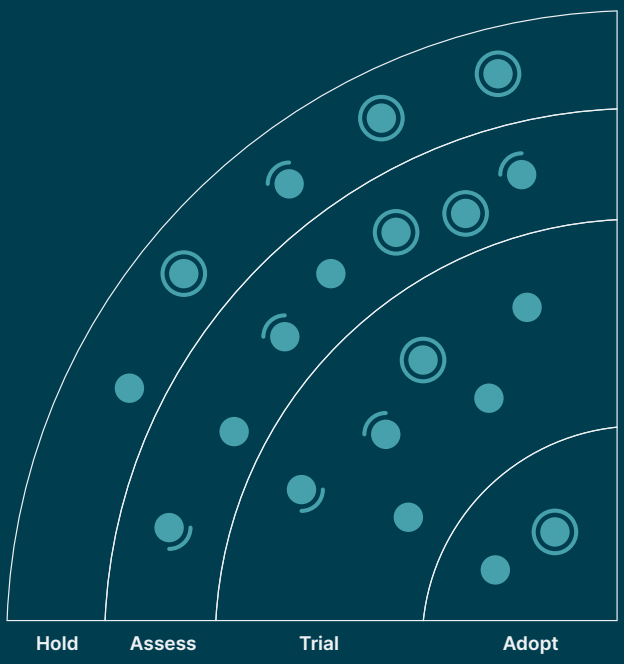
Nós encorajamos você a explorar essas tecnologias. O Radar é gráfico por natureza, agrupando os itens em técnicas, ferramentas, plataformas e linguagens & frameworks. No caso de itens que podem ser classificados em mais de um quadrante, escolhemos aquele que parece mais adequado. Além disso, agrupamos esses itens em quatro anéis para refletir a nossa posição atual em relação a cada um.

Para mais informações sobre o Radar, veja: thoughtworks.com/radar/faq.

Radar em um relance

A ideia por trás do Radar é rastrear coisas interessantes, que chamamos de blips. Organizamos blips no Radar usando duas categorias: quadrantes e anéis. Os quadrantes representam as diferentes naturezas dos blips. Os anéis indicam o estágio do ciclo de adoção em que consideramos que cada blip esteja.

Um blip é uma tecnologia ou técnica que desempenha um papel significativo no desenvolvimento de software. Os blips estão sempre em movimento, o que significa que suas posições no Radar estão constantemente mudando — geralmente indicando que nossa confiança tem crescido à medida que se movimentam entre os anéis



Adote: Acreditamos firmemente que a indústria deveria adotar esses itens. Nós os usamos quando são apropriados em nossos projetos.

Experimente: Vale a pena ir atrás. É importante entender como desenvolver essa capacidade. As empresas devem testar esta tecnologia em um projeto que possa lidar com o risco.

Avalie: Vale explorar com o objetivo de compreender como isso afetará sua empresa.

Evite: Prossiga com cautela.

Novo **Mudança de anel** **Sem modificação**

Nosso Radar é um olhar para o futuro. Para abrir o espaço para novos itens, apagamos itens que não foram modificados recente, o que não é um reflexo de seu valor, mas uma solução para nossa limitação de espaço.

Contribuições

Contribuições O Conselho Consultivo de Tecnologia (TAB) é um grupo formado por 21 tecnologistas experientes da Thoughtworks. O TAB se reúne duas vezes por ano pessoalmente e quinzenalmente por videochamada. Sua principal atribuição é ser um grupo consultivo para a CTO da Thoughtworks, Rebecca Parsons.

O TAB atua examinando tópicos que afetam soluções de tecnologia e tecnologistas da Thoughtworks. Esta edição do Thoughtworks Technology Radar é baseada em uma reunião do TAB realizada remotamente em março de 2023.

Rebecca Parsons (CTO)

Martin Fowler (Chief Scientist)

Bharani Subramaniam

Birgitta Böckeler

Brandon Byars

Camilla Falconi Crispim

Erik Dörnenburg

Fausto de la Torre

Hao Xu

Ian Cartwright

James Lewis

Marisa Hoenig

Maya Ormaza

Mike Mason

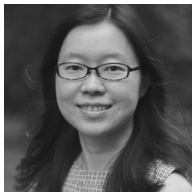
Neal Ford

Pawan Shah

Scott Shaw

Selvakumar Natesan

Shangqi Liu



Sofia Tania

Vanya Seth

Tradução: Akina Kurita, Alexandre Ramires, Amanda Batista, Augusto Alessio, Camila Crispim, Felipe Damasceno, Guilherme Silveira, Karolina Pacheco, Leonardo Ramos, Luciano Ramalho, Patrick Prado, Ricardo Cavalcanti e Thiago Gregorio.

Temas

A ascensão meteórica da IA prática

Não, este texto temático não foi escrito pelo ChatGPT. A inteligência artificial (IA) vem borbulhando discretamente em áreas especializadas há décadas, e ferramentas como o GitHub Copilot já existem (e vêm gradualmente sendo adotadas) há alguns anos. No entanto, nos últimos meses, ferramentas como o ChatGPT nos reorientaram de maneira radical para o que é possível e tornaram as ferramentas amplamente disponíveis. Vários blips nesta edição do Radar abordam usos práticos de IA para projetos (além de sugerir qual código requer ajustes): Desenvolvimento orientado a testes auxiliado por IA, valendo-se da IA para ajudar a criar modelos de análise e muito mais. Semelhante a como as planilhas permitiram que os contadores parassem de usar máquinas de somar para recalcular planilhas complexas manualmente, a próxima geração de IA assumirá tarefas para aliviar os trabalhadores de tecnologia, incluindo pessoas desenvolvedoras, substituindo tarefas tediosas que exigem conhecimento (mas não sabedoria).

No entanto, alertamos contra usos excessivos ou inadequados. No momento, os modelos de IA são capazes de gerar um bom primeiro rascunho. Mas o conteúdo gerado sempre precisa ser monitorado por um humano que possa validá-lo, moderá-lo e usá-lo com responsabilidade. Se essas precauções forem ignoradas, os resultados podem levar a riscos de reputação e segurança para organizações e usuários. Até mesmo algumas demonstrações de produtos advertem os usuários: “O conteúdo gerado por IA pode conter erros, certifique-se de que é preciso e apropriado antes de usá-lo”.

Acessibilidade acessível

A acessibilidade tem sido uma consideração importante para as organizações há muitos anos. Recentemente, destacamos as experiências de nossas equipes com o crescente conjunto de ferramentas e técnicas que agregam acessibilidade aprimorada ao desenvolvimento, e várias regiões em que nossas equipes destacaram o conhecimento dessas técnicas por meio de campanhas de conscientização. Apresentamos blips relacionados à acessibilidade no desenvolvimento do pipeline de integração contínua, manuais de design, testes de acessibilidade guiados por inteligência, linting e testes de unidade. A crescente conscientização acerca deste tão importante tópico é bem-vinda; técnicas que dão a mais pessoas acesso à funcionalidade de maneiras aprimoradas, só podem ser uma coisa boa.

Lambda quicksand

Funções sem servidor — AWS Lambdas — aparecem cada vez mais nas caixas de ferramentas de pessoas arquitetas e pessoas desenvolvedoras, e são empregadas em uma ampla variedade de tarefas úteis que trazem os benefícios da infraestrutura em nuvem. No entanto, como muitas coisas úteis, as soluções às vezes começam adequadamente simples, mas depois, a partir de um sucesso gradual e persistente, continuam evoluindo até ultrapassarem as limitações inerentes ao paradigma e afundarem na areia sob seu próprio peso. Ainda que vejamos muitas aplicações bem-sucedidas de soluções de estilo Serverless, também ouvimos muitas advertências a respeito de nossos projetos, como o antipadrão Lambda pinball. Também vemos mais ferramentas que parecem resolver problemas, mas são propensas a uso indevido. Por exemplo, ferramentas que facilitam o compartilhamento de código entre Lambdas ou orquestram interações complexas podem resolver um problema comum simples, mas correm o risco de recriar alguns antipadrões de arquitetura terríveis com novos blocos de construção. Se você precisa de uma ferramenta para gerenciar o compartilhamento de código e implantação independente em uma coleção de funções Serverless, talvez seja hora de repensar a adequação da abordagem. Como todas as soluções de tecnologia, o modelo sem servidor tem aplicações adequadas, mas muitos de seus recursos incluem compensações cujas desvantagens se tornam mais agudas à medida que a solução evolui.

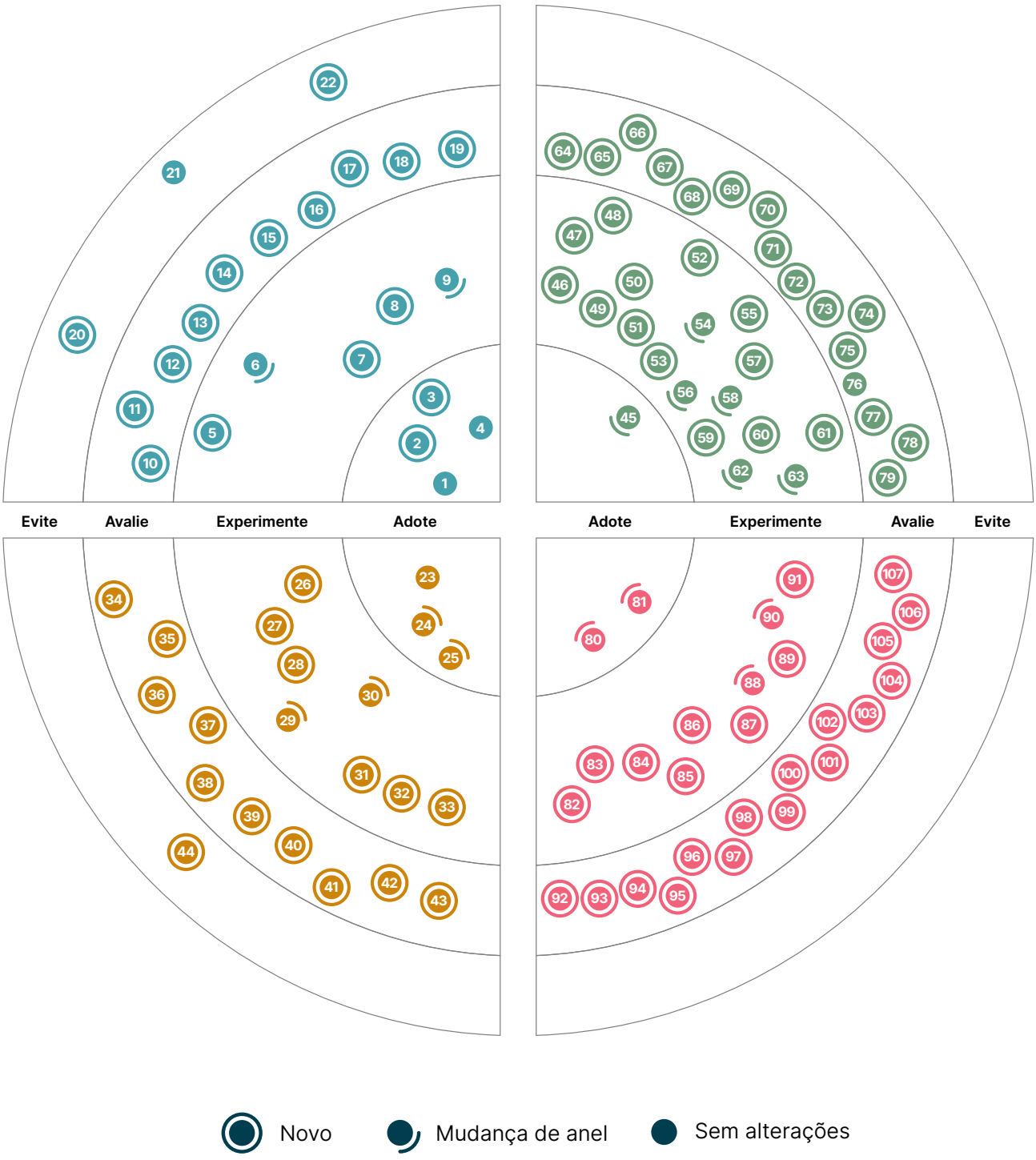
O rigor da engenharia encontra sistemas analíticos e IA

Nós vimos por muito tempo "a incorporação da qualidade" como um aspecto essencial do desenvolvimento de análises e modelos de aprendizado de máquina confiáveis. Transformações orientadas a testes, testes de sanidade de dados e testes de modelo de dados fortalecem os pipelines de dados que alimentam os sistemas analíticos. A validação do modelo e a garantia de qualidade são cruciais para combater os vieses e garantir sistemas éticos de ML, com resultados equitativos. Ao integrar essas práticas, as empresas ficam mais bem posicionadas para aproveitar a IA e o aprendizado de máquina e criar soluções responsáveis e orientadas por dados que atendem a uma base de pessoas usuárias diversificada. O ecossistema de ferramentas correspondente não parou de crescer e amadurecer. Por exemplo, a Soda Core, uma ferramenta de qualidade de dados, permite a validação dos dados assim que chegam ao sistema, e verificações de monitoramento automatizadas para anomalias. O Deepchecks permite a interseção de integração contínua e validação de modelo, uma etapa importante na incorporação de boas práticas de engenharia em configurações de análise. O Giskard ajuda a garantir a qualidade de modelos de IA, permitindo que os designers detectem vieses e outras facetas negativas dos modelos, o que se alinha com nosso incentivo para pisar em águas éticas com cuidado ao desenvolver soluções com IA. Encaramos essas ferramentas de amadurecimento como mais uma evidência da integração de análise e aprendizado de máquina e sua integração às boas práticas de engenharia.

Declarar ou programar?

Uma discussão aparentemente perpétua que acontece em todas as reuniões do Radar ganhou destaque particular desta vez - para determinada tarefa, você deve escrever uma especificação declarativa usando JSON, YAML ou algo específico do domínio como HCL, ou deve escrever código em uma linguagem de programação de uso geral? Por exemplo, discutimos as diferenças entre Terraform Cloud Operator e Crossplane, usar ou não o AWS CDK e usar Dagger para programar um pipeline de implantação, entre outros casos. Especificações declarativas, embora geralmente mais fáceis de ler e escrever, oferecem abstrações limitadas que levam a código repetitivo. Linguagens de programação convencionais podem usar abstrações para evitar a duplicação, mas essas abstrações podem tornar o código consideravelmente mais difícil de seguir, especialmente quando as abstrações estão dispostas em camadas, após anos de alterações. Em nossa experiência, não há uma resposta universal para a pergunta acima. As equipes devem considerar ambas as abordagens e, quando uma solução for difícil de implementar de forma limpa em um tipo de linguagem, elas devem reavaliar o outro tipo. Pode até fazer sentido dividir preocupações e implementá-las com diferentes linguagens.

O Radar



O Radar

Técnicas

Adote

1. Aplicando gestão de produto a plataformas internas
2. Infraestrutura de CI/CD como serviço
3. Redução de dependências
4. Custo de execução como função de aptidão arquitetural

Experimente

5. Anotações de acessibilidade em projetos
6. Plataformas low-code delimitadas
7. Front-ends de demonstração para produtos somente de API
8. Arquitetura lakehouse
9. Credenciais verificáveis

Avalie

10. Design de teste de componentes ciente da acessibilidade
11. Desenvolvimento de testes primeiro auxiliado por IA
12. LLMs específicos de um domínio
13. Testes de acessibilidade guiados inteligentes
14. Logseq como base de conhecimento da equipe
15. Engenharia de prompt
16. Análise de conectividade ao testar infraestrutura
17. LLMs Auto-hospedados
18. Rastrear a saúde em vez da dívida tecnológica
19. Confiança zero para pipelines de CI/CD

Evite

20. Gerenciamento casual de webhooks
21. Lambda pinball
22. Planejando para utilização total

Plataformas

Adote

23. Contentful
24. GitHub Actions
25. K3s

Experimente

26. Apache Hudi
27. Arm na nuvem
28. Ax
29. DuckDB
30. Feature Store
31. RudderStack
32. Strapi
33. TypeDB

Avalie

34. Autoware
35. Cozo
36. Dapr
37. Immuta
38. Matter
39. Modal
40. Neon
41. OpenLineage
42. Passkeys
43. Spin

Evite

44. Denodo como ferramenta primária de transformação de dados

O Radar

Ferramentas

Adote

45. DVC

Experimente

46. Akeyless
47. Apicurio Registry
48. EventCatalog
49. FOSSA
50. Gitleaks
51. Helmfile
52. IBM Equal Access Accessibility Checker
53. Ktlint
54. Kubeflow
55. Mend SCA
56. Mozilla SOPS
57. Ruff
58. Soda Core
59. Steampipe
60. Terraform Cloud Operator
61. TruffleHog
62. Typesense
63. Vite

Avalie

64. axe Linter
65. ChatGPT
66. DataFusion
67. Deepchecks
68. Ferramentas de tradução de tokens de design
69. Devbox
70. Evidently
71. Giskard
72. GitHub Copilot
73. iamlive
74. Kepler
75. Operador de segredos externos do Kubernetes
76. Kubeshark
77. Obsidian
78. Ory Kratos
79. Philips's self-hosted GitHub runner

Evite

—

Linguagens e Frameworks

Adote

80. Gradle Kotlin DSL
81. PyTorch

Experimente

82. dbt-unit-testing
83. Jetpack CameraViewfinder
84. Jetpack DataStore
85. Mikro ORM
86. Preferência de linguagem por app
87. Quarto
88. River
89. Stencil
90. Synthetic Data Vault
91. Vitest

Avalie

92. .NET 7 Native AOT
93. .NET MAUI
94. dbt-expectations
95. Directus
96. Ferrocene
97. Flutter para dispositivos embarcados
98. Fugue
99. Galacean Engine
100. LangChain
101. mljar-supervised
102. nanoGPT
103. pandera
104. Qwik
105. SolidJS
106. Turborepo
107. WebXR Device API

Evite

—

Técnicas

Adote

1. Aplicando gestão de produto a plataformas internas
2. Infraestrutura de CI/CD como serviço
3. Redução de dependências
4. Custo de execução como função de aptidão arquitetural

Experimente

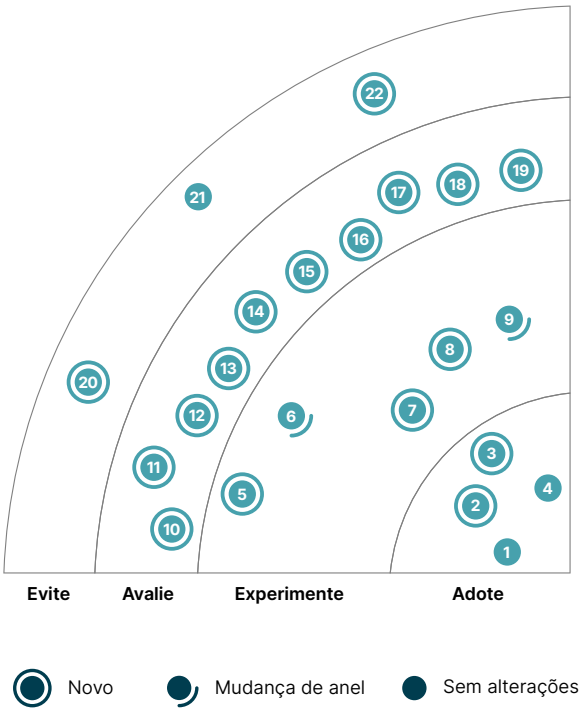
5. Anotações de acessibilidade em projetos
6. Plataformas low-code delimitadas
7. Front-ends de demonstração para produtos somente de API
8. Arquitetura lakehouse
9. Credenciais verificáveis

Avalie

10. Design de teste de componentes ciente da acessibilidade
11. Desenvolvimento de testes primeiro auxiliado por IA
12. LLMs específicos de um domínio
13. Testes de acessibilidade guiados inteligentes
14. Logseq como base de conhecimento da equipe
15. Engenharia de prompt
16. Análise de conectividade ao testar infraestrutura
17. LLMs Auto-hospedados
18. Rastrear a saúde em vez da dívida tecnológica
19. Confiança zero para pipelines de CI/CD

Evite

20. Gerenciamento casual de webhooks
21. Lambda pinball
22. Planejando para utilização total



Novo Mudança de anel Sem alterações

1. Aplicando gestão de produto a plataformas internas

Adote

Continuamos recebendo bons feedbacks das equipes que aplicam gestão de produto a plataformas internas. Um recurso importante a ser lembrado, no entanto: não se trata apenas de estrutura de equipe ou mudar o nome de equipes de plataforma existentes; trata-se também de aplicar práticas de trabalho centradas no produto dentro da equipe. Especificamente, recebemos feedback de que as equipes enfrentam desafios com essa técnica, a menos que tenham uma mentalidade centrada no produto. Isso provavelmente significa papéis adicionais, como gerente de produto, além de mudanças em outras áreas, como coleta de requisitos e medição de sucesso. Trabalhar dessa maneira significa estabelecer empatia com as consumidoras internas (as equipes de desenvolvimento) e colaborar com elas no design. As gerentes de produto da plataforma criam roteiros e garantem que a plataforma agregue valor aos negócios e aprimore a experiência da desenvolvedora. Continuamos a ver esta técnica como chave para construir plataformas internas para lançar novas soluções digitais de forma rápida e eficiente.

2. Infraestrutura de CI/CD como serviço

Adote

As opções para infraestrutura de CI/CD como serviço se tornaram tão diversas e maduras que os casos nos quais vale a pena gerenciar toda a sua própria infraestrutura de CI/CD internamente estão se tornando muito raros. Usar serviços gerenciados como GitHub Actions, o Azure DevOps ou o Gitlab CI/CD traz todas as vantagens (e desvantagens) comuns dos serviços gerenciados na nuvem. Não é preciso gastar tempo, esforços e custos de hardware na manutenção e na operação de uma infraestrutura muitas vezes complexa. As equipes podem se valer da elasticidade e do autosserviço, enquanto obter mais agentes corretos, um novo plugin ou um novo recurso é frequentemente um gargalo em empresas que hospedam seu próprio CI. Mesmo os casos de uso que exigem que a compilação/integração e a verificação sejam executados em hardware próprio podem agora ser atendidos por executores auto-hospedados (escrevemos sobre alguns para o GitHub Actions, controlador /executor de ações e para o Philips's self-hosted GitHub runner). Observe, entretanto, que você não vai receber segurança automática apenas por usar serviços gerenciados; apesar dos serviços mais maduros oferecerem todos os recursos de segurança necessários, é necessário que você mesmo os use para implementar um modelo de segurança de zero confiança para sua infraestrutura de CI/CD.

3. Redução de dependências

Adote

Kits e modelos iniciais são amplamente usados em projetos de software para acelerar a configuração inicial, mas podem gerar muitas dependências desnecessárias para um projeto específico. É importante praticar a redução de dependências — examinando periodicamente essas dependências e removendo todas as que não são usadas. Isso ajuda a reduzir os tempos de compilação e implantação e diminui a superfície de ataque do projeto, removendo possíveis vulnerabilidades. Embora essa não seja uma técnica nova, dada a frequência crescente de ataques às cadeias de fornecimento de software, defendemos uma atenção renovada a ela.

4. Custo de execução como função de aptidão arquitetural

Adote

Estimar, rastrear e prever automaticamente o custo de execução da infraestrutura de nuvem é crucial para as organizações de hoje. Os modelos de preços inteligentes dos provedores de nuvem, combinados com a proliferação de parâmetros de preços e a natureza dinâmica da arquitetura atual, podem levar a custos de execução surpreendentemente altos. Embora essa técnica esteja em adoção desde 2019, queremos destacar a importância de considerar o custo de execução como função de aptidão arquitetural , especialmente hoje, devido à adoção acelerada da nuvem e à crescente atenção às práticas de FinOps. Muitas plataformas comerciais fornecem ferramentas que podem consolidar e explicar os custos da nuvem para os líderes empresariais. Alguns deles são projetados para mostrar os custos de execução da nuvem para organizações financeiras ou para unidades de negócios.

No entanto, as decisões de consumo de nuvem geralmente são feitas no nível de engenharia, onde os sistemas são projetados. É importante que as engenheiras que tomam as decisões de projeto tenham alguma forma de prever o impacto de custo de suas decisões arquitetônicas. Algumas equipes automatizam essa previsão no início do ciclo de vida do desenvolvimento. Ferramentas como Infracost ajudam as equipes a prever o impacto do custo ao pensar em possíveis alterações na infraestrutura como código. Esse cálculo pode ser automatizado e inserido no pipeline de CD. Observe que o custo será afetado por decisões arquitetônicas combinadas com os reais níveis de uso; para fazer isso de maneira correta, você precisa de boas projeções dos níveis de uso esperados. O feedback antecipado e frequente acerca do custo de execução pode impedir que ele suba. Quando o custo previsto se desvia do esperado ou aceitável, a equipe pode discutir se é hora de evoluir a arquitetura.

5. Anotações de acessibilidade em projetos

Experimente

Quanto mais cedo a acessibilidade for considerada na entrega de software, mais fácil e barato será garantir que o que for criado, funcionará para o maior número possível de pessoas. Ferramentas que ajudam a comunicar anotações de acessibilidade em design de projetos permitem às equipes considerar elementos importantes como a estrutura do documento, HTML semântico, e textos alternativos desde o início do trabalho. Isso permite garantir que as interfaces da usuária atendam aos padrões globais de acessibilidade e resolvam falhas comuns que são, na verdade, muito fáceis de evitar. O Figma oferece uma variedade de plugins de notação de acessibilidade: o A11y Annotation Kit, a Biblioteca de anotações de acessibilidade do Twitter e o Axe for Designers do conjunto de ferramentas do Axe.

6. Plataformas low-code delimitadas

Experimente

Sempre fomos defensoras de que se escreva menos código. Simplicidade é um dos valores fundamentais subjacentes aos nossos padrões sensatos para o desenvolvimento de software. Por exemplo, tentamos não antecipar necessidades, e só introduzimos código que satisfaça requisitos de negócios imediatos, e nada mais. Uma maneira de conseguir isso é criar plataformas de engenharia que tornem possível tal ação em âmbito organizacional.

Este também é o objetivo declarado de muitas plataformas low-code, que crescem em popularidade no momento. Plataformas como Mendix ou Microsoft Power Apps podem expor processos de negócios comuns para reutilização e simplificar os problemas de implantação de novas funcionalidades e colocá-las nas mãos das usuárias. Essas plataformas deram grandes passos nos últimos anos com capacidade de teste e suporte para boas práticas de engenharia. Elas são particularmente úteis para tarefas simples ou aplicativos acionados por eventos. Contudo, exigir que se adaptem a uma gama quase infinita de requisitos de negócios é algo complexo. Ainda que as pessoas desenvolvedoras estejam escrevendo pouco código (ou nenhum código), também devem especializar-se em uma plataforma comercial abrangente. Aconselhamos que as empresas considerem se precisam de todas as funcionalidades que tais produtos oferecem ou se é melhor ir atrás de Plataformas low-code delimitadas, desenvolvendo sua própria plataforma como um produto interno ou restringindo cuidadosamente o uso de produtos comerciais low-code para aquelas tarefas simples nas quais eles se destacam.

7. Front-ends de demonstração para produtos somente de API

Experimente

Um dos grandes desafios no desenvolvimento de APIs é capturar e comunicar seu valor comercial. APIs são, por natureza, artefatos técnicos. Levando-se em conta que as pessoas desenvolvedoras podem facilmente compreender payloads JSON, especificações OpenAPI (Swagger) e demos do Postman, as partes interessadas nos negócios tendem a responder melhor às demonstrações com as quais podem interagir. O valor do produto é articulado de forma mais fácil quando você pode vê-lo e tocá-lo, e é por isso que às vezes achamos que vale a pena investir em frontends de demonstração para produtos somente de API. Quando uma UI gráfica personalizada é criada junto com um produto de API, as partes interessadas podem ver analogias com formulários ou relatórios em papel que podem ser mais familiares para elas. À medida que o modelo de interação e a riqueza da UI de demonstração evoluem, decisões mais bem informadas podem ser tomadas acerca da direção que deve seguir o produto de API. Trabalhar na interface da usuária tem o benefício adicional de aumentar a empatia das pessoas desenvolvedoras pelas usuárias corporativas. Esta não é uma técnica nova — temos feito isso com sucesso quando necessário, desde que os produtos de API existam. Porém, como essa técnica é pouco conhecida, achamos importante chamar a atenção para ela.

8. Arquitetura lakehouse

Experimente

A Arquitetura Lakehouse é um estilo de arquitetura que combina a escalabilidade dos data lakes com a confiabilidade e o desempenho dos data warehouses. Ela permite que organizações armazenem e analisem grandes volumes de dados variados em uma única plataforma, ao invés de ter os dados separados em camadas de data lakes e data warehouses, e usando as mesmas ferramentas e técnicas familiares, baseadas em SQL. Apesar do termo ter sido muitas vezes associado a fornecedores como Databricks, alternativas abertas como Delta Lake, Apache Iceberg e Apache Hudi valem a pena ser considerados. A arquitetura lakehouse pode complementar implementações de malha de dados. Equipes autônomas de produtos de dados podem escolher aproveitar as vantagens de uma Lakehouse em seus produtos.

9. Credenciais verificáveis

Experimente

Quando o incluímos no Radar pela primeira vez, há três anos, as credenciais verificáveis (CV) eram um padrão intrigante com algumas aplicações promissoras em potencial, mas não amplamente conhecidas ou compreendidas fora da comunidade de entusiastas. Isso era particularmente verdadeiro quando se tratava de instituições credenciadas, como governos estaduais, que seriam responsáveis pela implementação dos padrões. Três anos e uma pandemia depois, cresceu a demanda por credenciais eletrônicas criptograficamente seguras, que respeitam a privacidade e são verificáveis por máquina. Como resultado, os governos começam a despertar para o potencial das CV. O padrão W3C coloca os detentores de credenciais no centro, o que é semelhante à nossa experiência ao usar credenciais físicas: os usuários podem colocar suas credenciais verificáveis em suas próprias carteiras digitais e mostrá-las a qualquer pessoa a qualquer momento, sem a permissão do emissor das credenciais. Essa abordagem descentralizada também ajuda os usuários a gerenciar melhor e divulgar seletivamente suas próprias informações, o que melhora muito a proteção da privacidade dos dados.

Várias de nossas equipes se envolveram em projetos que incluem tecnologia de credenciais verificáveis nos últimos 6 meses. E de uma forma que não nos surpreende, os cenários variam entre países e departamentos governamentais. Nossa equipe explorou variadas combinações de identificadores descentralizados, credenciais verificáveis e apresentações verificáveis em múltiplos projetos. Este é um campo em desenvolvimento, e agora que temos mais experiência, queremos acompanhá-lo no Radar.

10. Design de teste de componentes ciente da acessibilidade

Avalie

Um dos muitos lugares no processo de entrega de software para considerar desde o início os requisitos de acessibilidade é durante o teste de componentes da web. Plug-ins de frameworks de teste como o chai-a11y-axe fornecem asserções em sua API para verificar o básico. Porém, além de usar o que os frameworks de teste têm a oferecer, o accessibility-aware component test design (design de testes cientes da acessibilidade de componentes) ajuda ainda mais a fornecer todos os elementos semânticos necessários para leitores de tela e outras tecnologias assistivas.

Em primeiro lugar, em vez de usar ids de teste ou classes para encontrar e selecionar os elementos que deseja validar, use um princípio de identificação de elementos por papéis ARIA ou outros atributos semânticos empregados por tecnologias assistivas. Algumas bibliotecas de teste, como a Testing Library, até recomendam isso em sua documentação. Em segundo lugar, não teste apenas as interações de clique; considere também os usuários que não podem usar um mouse ou ver a tela e considere adicionar testes adicionais para o teclado e outras interações.

11. Desenvolvimento de testes primeiro auxiliado por IA

Avalie

Como muitas outras na indústria de software, temos explorado as ferramentas de IA, em rápida evolução, que possam nos ajudar a escrever código. Temos visto muita gente alimentar o ChatGPT com uma implementação e pedir que gere testes para aquele código. Mas como acreditamos muito no desenvolvimento orientado a testes, e nem sempre queremos alimentar um modelo externo com nosso código de implementação potencialmente confidencial, uma de nossas experiências nesse espaço é uma técnica que chamamos Desenvolvimento de testes primeiro auxiliado por IA. Nessa abordagem, fazemos com que o ChatGPT gere testes para nós, e então uma pessoa desenvolvedora implementa a funcionalidade. Especificamente, primeiro descrevemos a stack de tecnologias e os padrões de design que estamos usando, em um "fragmento" de prompt que é reutilizável para muitos casos de uso. Então descrevemos a funcionalidade específica que desejamos implementar, incluindo os critérios de aceitação. Baseado nisso tudo, pedimos ao ChatGPT para gerar um plano de implementação para a funcionalidade, usando nosso estilo de arquitetura e nossa stack de tecnologias. Após verificar o plano de implementação, pedimos ao modelo que gere testes para nossos critérios de aceitação.

Essa abordagem tem funcionado surpreendentemente bem para nós: ela exigiu que a equipe desenvolvesse uma descrição sucinta de seu estilo de arquitetura e ajudou pessoas desenvolvedoras iniciantes e novos membros da equipe a programarem funcionalidade de maneira alinhada ao estilo existente da equipe. A maior desvantagem da abordagem é que, apesar de não fornecermos nosso código-fonte ao modelo, ainda assim o alimentamos com informações potencialmente sensíveis, como nossa stack de tecnologias e descrições de funcionalidades. As equipes devem se assegurar de trabalhar com suas assessoras legais, para evitar qualquer problema de propriedade intelectual, pelo menos até que uma versão "corporativa" dessas ferramentas de IA esteja disponível.

12. LLMs específicos de um domínio

Avalie

Apresentamos Modelos Grandes de Linguagem (LLMs) como BERT e ERNIE no Radar anterior; LLMs específicos de domínio, no entanto, são uma tendência emergente. O ajuste fino de LLMs de uso geral com dados específicos de domínio pode adaptá-los para várias tarefas, incluindo recuperação de informações, aumento do suporte ao cliente e criação de conteúdo. Essa prática tem mostrado resultados promissores em setores como jurídico e financeiro, conforme demonstrado por OpenNyAI para análise de documentos jurídicos. Com mais organizações experimentando LLMs e novos modelos como GPT4 sendo lançados, podemos esperar mais casos de uso específicos de domínio em um futuro próximo.

No entanto, existem desafios e armadilhas a serem considerados. Primeiro, os LLMs podem estar errados, por isso é essencial criar mecanismos em seu processo para garantir a precisão dos resultados. Em segundo lugar, os LLMs de terceiros podem reter e compartilhar novamente seus dados, representando um risco para informações autenticadas e confidenciais. As organizações devem revisar cuidadosamente os termos de uso e a confiabilidade dos provedores ou considerar o treinamento e a execução de LLMs em uma infraestrutura que controlem. Como acontece com qualquer nova tecnologia, as empresas devem agir com cuidado, entendendo as implicações e os riscos associados à adoção de um LLM.

13. Testes de acessibilidade guiados inteligentes

Avalie

Pode ser um pouco assustador tornar um aplicativo da web compatível com tecnologias assistivas quando você mesmo nunca as usa e sente que ainda não sabe coisa alguma a respeito de diretivas, como as Diretrizes de acessibilidade de conteúdo da Web (WCAG). Os testes de acessibilidade guiados inteligentes são uma categoria de ferramentas que ajudam a testar se você fez a coisa certa, sem precisar ser uma especialista em acessibilidade. Essas ferramentas são extensões do navegador que examinam seu site, resumem como a tecnologia assistiva o interpretaria e, em seguida, fazem uma série de perguntas para confirmar se a estrutura e os elementos que você criou são os pretendidos. Usamos o axe DevTools, o Accessibility Insights for Web ou o ARC Toolkit em alguns de nossos projetos.

14. Logseq como base de conhecimento da equipe

Avalie

O gerenciamento de conhecimento da equipe é um conceito familiar para equipes que usam ferramentas como wikis para armazenar informações e integrar novos membros. Algumas de nossas equipes agora preferem usar o Logseq como base de conhecimento da equipe. Logseq, um sistema de gerenciamento de conhecimento de código aberto, é alimentado por um banco de dados gráfico, que ajuda as usuárias a organizar conceitos, anotações e ideias, e pode ser adaptado para uso em equipe com armazenamento baseado em Git. Ele também permite que as equipes construam uma base de conhecimento democrática e acessível, proporcionando a cada membro da equipe uma jornada de aprendizado personalizada e proporcionando uma integração eficiente. No entanto, como acontece com qualquer ferramenta de gestão do conhecimento, as equipes precisarão aplicar uma boa curadoria e gerenciamento de sua base de conhecimento, para evitar a sobrecarga ou desorganização das informações.

Ainda que uma funcionalidade semelhante esteja disponível em ferramentas como o Obsidian, a principal diferença está no foco do Logseq no consumo, com links baseados em parágrafos que permitem aos membros da equipe encontrarem rapidamente o contexto relevante sem que precisem ler um artigo inteiro.

15. Engenharia de prompt

Avalie

Engenharia de prompt , se refere ao processo de projetar e refinar prompts para modelos de IA generativos, visando obter respostas de alta qualidade. Isso envolve a construção cuidadosa de prompts que sejam específicos, claros e relevantes para a aplicação ou tarefa desejadas, de forma a induzir respostas úteis do modelo. A “engenharia de prompt” tem por objetivo melhorar a capacidade dos modelos grandes de linguagem (LLMs) em tarefas como responder a perguntas, raciocínio aritmético ou em contexto específicos de um domínio. Para o desenvolvimento de software, a engenharia de prompt poderia ser usada para fazer um LLM escrever uma história, uma API ou um banco de testes baseado em uma conversa curta com uma das partes interessadas ou em algumas anotações. Desenvolver técnicas eficazes de estímulo está se tornando uma habilidade valiosa para trabalhar com sistemas de IA. Há debates sobre se a engenharia de prompt é uma ciência ou uma arte, e sobre se riscos de segurança, tais como “ataques de injeção de estímulos”, também devem ser levados em consideração.

16. Análise de conectividade ao testar infraestrutura

Avalie

Quando implantamos uma infraestrutura como código, notamos que tempo demais pode ser gasto diagnosticando e reparando problemas de produção resultantes da incapacidade de comunicação entre os sistemas. Como a topologia de rede entre eles pode ser complexa, nem todas as rotas podem ser transitáveis, mesmo que as portas e terminais individuais tenham sido configurados de maneira correta. As práticas de teste de infraestrutura geralmente incluem verificar se as portas certas estão abertas ou fechadas ou se um endpoint pode ser acessado, mas só recentemente começamos a fazer análise de conectividade ao testar infraestrutura. A análise geralmente envolve mais do que simples determinações de sim/não. Por exemplo, uma ferramenta pode percorrer e relatar várias rotas por meio de gateways de trânsito. Essa técnica é suportada por ferramentas em todos os principais provedores de nuvem. O Azure tem um serviço chamado Network Watcher que pode ser usado em scripts de testes automatizados e o GCP suporta testes de conectividade. Agora, na AWS, você pode testar a acessibilidade entre contas na mesma organização.

17. LLMs Auto-hospedados

Avalie

Grandes modelos de linguagens (LLMs) geralmente requerem infraestrutura de GPU significativa para operar. Agora estamos começando a ver ferramentas para outras plataformas, como o llama.cpp, que possibilitam a execução de LLMs em plataformas de hardware diferentes – incluindo Raspberry Pis, laptops e servidores comuns. Assim, os LLMs auto-hospedados agora são uma realidade. Atualmente, há vários LLMs de código aberto como o GPT-J, o GPT-JT e o LLaMA que podem ser auto-hospedados. Essa abordagem traz vários benefícios, como melhor controle no ajuste fino para o caso de uso específico, segurança e privacidade aprimoradas, bem como, obviamente, acesso offline. No entanto, você deve avaliar cuidadosamente os recursos dentro da organização e o custo de rodar tais LLMs antes de decidir auto-hospedar.

18. Rastrear a saúde em vez da dívida tecnológica

Avalie

Rastrear a dívida tecnológica é um tópico perene em organizações de desenvolvimento de software. O que é débito técnico e o que não é? Como se prioriza isso? E o mais importante, como se expressa o valor a ser pago aos seus investidores internos? Seguindo as diretrizes do Manifesto Ágil — “enquanto há valor no item à direita, valorizamos mais o item à esquerda” — gostamos da ideia de rastrear a saúde em vez da dívida. O pessoal da REA na Austrália compartilha um bom exemplo de como pode ocorrer esse rastreamento de integridade. Eles rastreiam as classificações do sistema nas categorias de desenvolvimento, operação e arquitetura.

Concentrar-se na saúde em vez da dívida é uma abordagem mais construtiva. Isso conecta uma equipe ao valor final da redução da dívida e os ajuda a priorizá-la. Cada parte da dívida técnica abordada deve, idealmente, ser conectável a uma das expectativas acordadas. As equipes devem tratar a classificação de integridade da mesma maneira que outros objetivos de nível de serviço (SLOs) e priorizar as melhorias sempre que saírem da “zona verde” para uma determinada categoria.

19. Confiança zero para pipelines de CI/CD

Avalie

Se não estiverem corretamente protegidas, a infraestrutura e as ferramentas que executam nossos pipelines de compilação e entrega podem se tornar uma grande fraqueza. Para compilar e implantar o software, esses pipelines precisam de acesso a dados e sistemas críticos, tais como código-fonte, credenciais e chaves secretas. Isso torna esses sistemas bastante convidativos para atores mal-intencionados. Nós então recomendamos fortemente a aplicação de modelos de segurança de arquitetura de confiança zero para pipelines de CI/CD e infraestrutura – confiando nelas o mínimo necessário. Isso abarca um conjunto de técnicas: se possível, autenticar seus pipelines junto ao seu provedor de serviços em nuvem via mecanismos de identidade de uma entidade federada como OIDC ("OpenID Connect"), em vez de dar a eles acesso direto a chaves secretas. Implementar o princípio do menor privilégio, minimizando o acesso de usuárias individuais ou de contas de executores, em vez de utilizar contas com acesso ilimitado. Usar os executores de forma efêmera, em vez de reutilizá-los, reduzindo o risco de exposição de segredos de atividades anteriores ou de executar atividades em executores comprometidos. Manter o software nos agentes e executores atualizado. Monitorar a integridade, a confidencialidade e a disponibilidade de seus sistemas de CI/CD da mesma forma que se monitora o software em produção.

Temos visto equipes esquecendo desses tipos de práticas, especialmente quando estão acostumados a trabalhar com uma infraestrutura autogerenciada de CI/CD em zonas internas da rede. E enquanto tais práticas são importantes nas redes internas, se tornam ainda mais cruciais quando se usa um serviço externo gerenciado, pois isso aumenta ainda mais a superfície de ataque e o raio de destruição.

20. Gerenciamento casual de webhooks

Evite

À medida que o trabalho remoto continua a aumentar, também aumenta a adoção de plataformas de colaboração por chat e ChatOps. Essas plataformas geralmente oferecem webhooks como uma forma simples de automatizar o envio de mensagens e notificações, mas notamos uma tendência preocupante: o gerenciamento casual de webhooks — onde eles são tratados como uma configuração em vez de um segredo ou uma credencial. Isso pode levar a ataques de phishing e a espaços internos comprometidos.

Os webhooks são credenciais que oferecem acesso privilegiado a um espaço interno e podem conter chaves de API com possibilidade de serem facilmente extraídas e utilizadas diretamente. Não tratá-los como segredos abre a possibilidade de ataques de phishing bem-sucedidos. Webhooks em repositórios Git podem ser facilmente extraídos e usados para enviar payloads fraudulentos, que a usuária talvez não tenha como autenticar. Para atenuar essa ameaça, as equipes que lidam com webhooks precisam mudar sua cultura e tratar os webhooks como credenciais confidenciais. As pessoas desenvolvedoras de software que criam integrações com plataformas ChatOps também devem estar atentas a esse risco e garantir que os webhooks sejam tratados com medidas de segurança adequadas.

21. Lambda pinball

Evite

Ainda que arquiteturas sem servidor possam ser extremamente úteis para resolver alguns problemas, elas vêm com um certo nível de complexidade, especialmente quando envolvem execução e fluxos de dados em vários Lambdas interdependentes — às vezes, isso pode resultar em uma arquitetura Lambda pinball. Nossas equipes relataram que manter e testar arquiteturas estilo Lambda pinball pode ser muito desafiador: entender a infraestrutura, implantação, diagnóstico e depuração pode se tornar difícil. A nível do código, o mapeamento simples entre os conceitos de domínio e os vários Lambdas envolvidos é praticamente impossível, tornando desafiadoras quaisquer alterações e adições. Embora acreditemos que ambientes sem servidor são a estrutura correta para alguns problemas e domínios, eles não são uma "bala de prata" para todos os problemas, e é por isso que você deve tentar evitar Lambda pinball. Um padrão que pode ajudar é fazer uma distinção entre interfaces públicas e publicadas, e aplicar delimitação de domínio, com interfaces publicadas entre eles.

22. Planejando para utilização total

Evite

Ainda que a prática de criar excesso de capacidade no processo de desenvolvimento seja bem conhecida na comunidade de gestão de produtos, ainda vemos muitas equipes planejando para utilização total de seus membros. Reservar alguma capacidade durante o planejamento do sprint geralmente leva a uma melhor previsibilidade e a uma melhor qualidade; promove a resiliência da equipe a eventos inesperados, como doenças, problemas em produção, solicitações inesperadas de produtos e dívidas técnicas, ao mesmo tempo em que permite atividades produtivas como formação da equipe e ideias que podem levar à inovação do produto. Trabalhar abaixo da utilização total significa que as equipes podem pensar mais sobre a robustez do software resultante e prestar mais atenção aos sinais corretos de observabilidade. Nossa experiência é que uma equipe completamente utilizada também leva a um colapso no rendimento, assim como uma rodovia totalmente utilizada gera tráfego lento e desanimador. Por exemplo, quando uma de nossas equipes apresentou problemas de suporte , foi observado um aumento de 25% na taxa de entrega e uma redução de 50% na volatilidade do tempo de cada ciclo ao planejar a velocidade de desenvolvimento baseando-se no uso de apenas dois, dos três pares de pessoas desenvolvedoras.

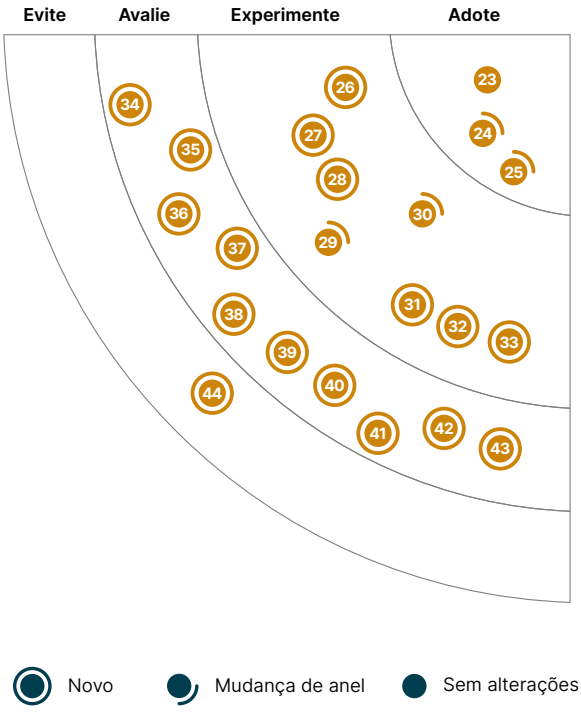
Plataformas

Adote
23. Contentful
24. GitHub Actions
25. K3s

Experimente
26. Apache Hudi
27. Arm na nuvem
28. Ax
29. DuckDB
30. Feature Store
31. RudderStack
32. Strapi
33. TypeDB

Avalie
34. Autoware
35. Cozo
36. Dapr
37. Immuta
38. Matter
39. Modal
40. Neon
41. OpenLineage
42. Passkeys
43. Spin

Evite
44. Denodo como ferramenta primária de transformação de dados



23. Contentful

Adote

Os sistemas de gerenciamento de conteúdo headless, tornaram-se um componente comum das plataformas digitais. O Contentful ainda é nossa escolha padrão neste espaço, mas novos participantes como o Strapi também nos impressionaram. Gostamos particularmente da abordagem API primeiro do Contentful e da implementação de CMS como código. Ele oferece suporte eficaz a primitivas de modelagem de conteúdo como código, bem como scripts de evolução de modelo de conteúdo, que permitem que ele seja tratado como outros esquemas de armazenamento de dados e que práticas como o design evolutivo de banco de dados sejam aplicadas ao desenvolvimento de CMS. Recentemente, o Contentful lançou um app framework para escrever aplicativos que facilitam a adaptação do Contentful a processos de negócios individuais, e a integração com outros serviços. Os aplicativos podem ser criados por e para uma organização específica, mas também está surgindo um mercado de aplicativos.

24. GitHub Actions

Adote

O GitHub Actions tornou-se um ponto de partida padrão para muitas equipes que precisam colocar CI/CD em funcionamento rapidamente em um ambiente totalmente novo. Entre outras coisas, ele pode lidar com fluxos de trabalho mais complexos e chamar outras ações dentro de ações compostas. Embora o ecossistema no GitHub Marketplace continue a crescer, ainda recomendamos cautela ao fornecer acesso no seu pipeline de desenvolvimento a GitHub Actions de terceiros. Recomendamos seguir os conselhos do GitHub acerca do reforço de segurança, para evitar o compartilhamento de dados secretos de maneira insegura. No entanto, a conveniência de criar seu fluxo de trabalho de compilação diretamente no GitHub ao lado de seu código-fonte, combinado com a capacidade de executar GitHub Actions localmente, usando ferramentas de código aberto como act, é uma opção atraente que simplificou a configuração e a integração de nossas equipes.

25. K3s

Adote

K3s continua sendo nossa distribuição padrão de Kubernetes para necessidades de computação de borda e ambientes com recursos limitados. É um Kubernetes leve e totalmente compatível, mas com sobrecarga operacional reduzida. Ele usa sqlite3 como back-end de armazenamento padrão em vez do etcd. Ocupa menos memória porque executa todos os componentes relevantes em um único processo. Usamos K3s em ambientes como sistemas de controle industrial e máquinas de ponto de venda e estamos muito satisfeitos com nossa decisão. Com o tempo de execução do K3s containerd agora com suporte a wasm, o K3s pode executar e gerenciar programas em WebAssembly diretamente, reduzindo ainda mais a sobrecarga de tempo de execução.

26. Apache Hudi

Experimente

Apache Hudi é uma plataforma de data lake de código aberto que oferece garantias transacionais ACID para o data lake. Nossas equipes tiveram uma ótima experiência usando o Hudi em um cenário de alto volume e alta taxa de transferência com inserções e upserts em tempo real. Gostamos especialmente da flexibilidade que a plataforma Hudi oferece para personalizar o algoritmo de compressão, o que auxilia a lidar com problemas de “small files”. Apache Hudi cai na mesma categoria da Delta Lake e da Apache Iceberg. Todas oferecem recursos semelhantes, mas cada uma difere nas implementações subjacentes e nas listas detalhadas de recursos.

27. Arm na nuvem

Experimente

Instâncias de processamento Arm na nuvem se tornaram cada vez mais populares durantes os útlmos anos, devido à sua eficiência de custo e energia, quando comparada às instâncias tradicionais baseadas em X86. Muito provedores de serviços na nuvem oferecem instâncias baseadas em Arm, incluindo a AWS, a Azure e a GCP. O custo benefício de rodar Arm na nuvem pode ser particularmente benéfico para negócios com grandes cargas de trabalho ou necessidade de escalar. Baseados na nossa experiência, recomendados instâncias de processamento Arm para todas as cargas de trabalho, a menos que existam dependências específicas de arquitetura. O ferramental para suportar múltiplas arquiteturas, como o multi-arch docker images, também simplificam o desenvolvimento e a implantação.

28. Ax

Experimente

Diante do desafio de explorar grandes espaços de configuração, em que pode-se levar um tempo significativo para avaliar uma determinada configuração, as equipes têm a possibilidade de recorrer à experimentação adaptativa, um processo iterativo, guiado por máquina, para encontrar soluções ideais de maneira eficiente em termos de recursos. Ax é uma plataforma para gerenciar e automatizar experimentos adaptativos, incluindo experimentos de aprendizado de máquina, testes A/B e simulações. Atualmente, ela suporta duas estratégias de otimização: otimização bayesiana usando BoTorch, que é construído sobre o PyTorch e contextual bandits. O Facebook, ao lançar a Ax e o BoTorch, descreveu casos de uso tais como o aumento da eficiência da infraestrutura de back-end, o ajuste de modelos de classificação e a otimização da pesquisa de hiperparâmetros para uma plataforma de aprendizado de máquina. Tivemos boas experiências usando Ax para uma variedade de casos de uso e, embora existam ferramentas para ajuste de hiperparâmetros, não temos conhecimento de uma plataforma que fornece funcionalidade em um escopo semelhante a Ax.

29. DuckDB

Experimente

O DuckDB é um banco de dados orientado a colunas para ciência de dados e cargas de trabalho analíticas. As analistas de dados geralmente carregam os dados localmente em ferramentas como pandas ou data.table para rapidamente analisar padrões e formular hipóteses antes de dimensionar a solução no servidor. Nós, no entanto, agora estamos usando o DuckDB para esses casos de uso, porque ele libera o potencial para fazer análises com massas de dados maiores que a memória disponível. O DuckDB suporta junções de intervalo (range joins), execução vetorizada e controle de concorrência multiversão (MVCC) para grandes transações, e nossas equipes estão muito satisfeitas com esses recursos.

30. Feature Store

Experimente

Qualquer sistema computacional precisa representar de forma apropriada o domínio em que é empregado, e deve sempre ser baseado em metas e objetivos fundamentais. Projetos de aprendizagem de máquina (ML) não são diferentes. Engenharia de Características é um aspecto crucial da engenharia e do projeto de sistemas computacionais de ML. A Feature Store (Armazém de Características) é um conceito de arquitetura relacionado, voltado para facilitar a identificação, descoberta e monitoramento de características pertinentes a um dado domínio ou problema do negócio. Implementar esse conceito envolve uma combinação de projeto arquitetônico, engenharia de dados e gerenciamento de infraestrutura, para criar um sistema de ML escalável, eficiente e confiável. No que diz respeito a ferramentas, é possível encontrar desde plataformas de código aberto até ofertas totalmente administradas, mas elas são apenas parte desse conceito. No projeto de sistemas de ML de ponta a ponta, implementar um armazém de características ajuda nas seguintes áreas: a habilidade de (1) definir as características corretas; (2) melhorar a reusabilidade e tornar as características disponíveis de forma consistente, independente do tipo de modelo, o que também inclui criar pipelines de engenharia de características, para curar os dados descritos no armazém de características; (3) permitir a descoberta de características e (4) permitir oferecimento de características. Nossas equipes se valem de feature stores em produção para colher esses benefícios para sistemas de aprendizagem de máquina de ponta a ponta.

31. RudderStack

Experimente

O RudderStack é uma plataforma de dados de clientes (CDP - customer data platform) que facilita o armazenamento de dados em um armazém de dados (data warehouse) ou lago de dados (data lake). Essa abordagem, cada vez mais conhecida como CDP Headless (sem interface gráfica), separa os recursos do CDP de sua interface de usuário e enfatiza a configurabilidade por meio de APIs e do data warehouse/lake como armazenamento primário. Como esperado de um produto desta categoria, o RudderStack possui um rico repositório de integrações com produtos de terceiros (tanto como origem quanto como coletor) e a capacidade de ingerir eventos personalizados. O RudderStack tem uma oferta comercial e uma versão OSS auto-hospedada com limitações de funcionalidade.

32. Strapi

Experimente

Strapi é um sistema de gerenciamento de conteúdo (CMS) headless (onde o front-end e o back-end são independentes) de código aberto baseado em NodeJS, semelhante ao Contentful. Ele já existe há algum tempo e o usamos com sucesso em alguns projetos. O Strapi fornece APIs REST e GraphQL, possui documentação abrangente, apresenta API de modelo de dados fácil de usar, e oferece suporte à personalização da interface do usuário e da lógica.

33. TypeDB

Experimente

TypeDB é um banco de dados gráfico de conhecimento, projetado para trabalhar com relações de dados complexas que facilitam a consulta e análise de grandes conjuntos de dados. A linguagem de consulta TypeQL do TypeDB tem uma sintaxe semelhante a do SQL facilitando a curva de aprendizado para definição de esquema, consulta e exploração. TypeDB vem com uma variedade de ferramentas que auxiliam no trabalho com o banco de dados, incluindo uma interface de linha de comando e uma interface gráfica da usuária, TypeDB Studio, que fornece alguns recursos para trabalhar com o TypeDB, como gerenciamento de esquemas, consulta de dados, visualização de relacionamentos ou até mesmo colaboração com outras pessoas. Há uma boa quantidade de documentação disponível e uma comunidade ativa para suporte. Nossas equipes o usaram para criar gráficos de conhecimento de conceitos taxonômicos em diferentes bancos de dados e aproveitaram seus fortes recursos de inferência adicionando novas regras de inferência, aumentando a eficiência e reduzindo a carga de trabalho. Com sua experiência de desenvolvedora intuitiva e comunidade de suporte, o TypeDB é um bom candidato a ser considerado por qualquer equipe que busca criar soluções de dados que dependem do relacionamento de dados complexos, incluindo dados de linguagem natural, mecanismos de recomendação e gráficos de conhecimento.

34. Autoware

Avalie

O Autoware é um stack de software de direção autônoma de código aberto construída sobre o ROS (Robot Operating System), que pode ser usada para desenvolver e implantar sistemas avançados de assistência ao condutor (ADAS) em uma ampla gama de veículos, como carros e caminhões. A plataforma fornece um conjunto de ferramentas e algoritmos para vários aspectos da direção autônoma, como percepção, tomada de decisão e controle. Também possui um módulo de planejamento e controle que gera uma trajetória para o veículo com base em seu ambiente e objetivos. Ela incentiva inovações em tecnologia de direção autônoma livres. Estamos construindo protótipos usando Autoware para validar ideias de novos produtos e a consideramos útil.

35. Cozo

Avalie

O Cozo é um banco de dados relacional incorporável que usa o Datalog para consultas. Nós estamos intrigados com seu suporte a consultas de time-travel e modelagem de dados de grafo em esquema relacional. Gostamos bastante que ele delegue o armazenamento de dados a mecanismos populares existentes - incluindo SQLite, RocksDB, Sled e TiKV. Apesar de o Cozo estar ainda em seus estágios iniciais de desenvolvimento, achamos que vale a pena avaliar.

36. Dapr

Avalie

O Dapr, sigla de Distributed Application Runtime (Runtime de Aplicação Distribuída), ajuda pessoas desenvolvedoras a criarem microsserviços resilientes com ou sem estado, que rodam na nuvem. Algumas pessoas podem confundí-lo com um service mesh, porque ele usa uma arquitetura de sidecar rodando junto com a aplicação mas em um processo separado. O Dapr é mais orientado a aplicações, e se concentra em encapsular a tolerância a falhas e a conectividade exigidas para desenvolver aplicações distribuídas. Por exemplo, o Dapr fornece múltiplos componentes, de invocação de serviço e publicação/assinatura de mensagem a bloqueio distribuído, todos modelos comuns da comunicação distribuída. Uma de nossas equipes avaliou Dapr em um projeto recente; dada a experiência positiva que tiveram, as pessoas estão ansiosas para usá-lo em projetos futuros.

37. Immuta

Avalie

Immuta é uma plataforma de segurança de dados que permite proteger o acesso a seus dados, descobrir dados sensíveis automaticamente e auditar os dados sendo usados em uma organização. No passado, falamos sobre a importância da automação, das práticas de engenharia e de tratar a política de segurança como código quando pensamos sobre as preocupações de segurança. Segurança de dados não é diferente. Nossas equipes têm explorado a plataforma Immuta para gerenciar políticas de dados como código, permitindo um controle de acesso refinado, além do que o controle de acesso baseado em funções (RBAC). Políticas com controle de versão podem ser testadas e então provisionadas como parte de um pipeline de CI/CD. Em um ecossistema de dados descentralizados, como aquele facilitado por uma malha de dados (data mesh), ter papéis específicos a um domínio pode levar a proliferação de papéis ou grupos no sistema de identificação. A capacidade da plataforma Immuta para controle de acesso baseado em atributos (ABAC) reduz a concessão de acessos a uma equação matemática ligando um "atributo" em um usuário, a uma "tag" na fonte de dados. A plataforma ainda é nova, mas certamente merece ser destacada para as necessidades de segurança de dados.

38. Matter

Avalie

O Matter é um padrão aberto para tecnologia de casas inteligentes, lançado por Amazon, Apple, Google, Comcast e a Zigbee Alliance (agora chamada Connectivity Standards Alliance, ou CSA, Aliança por Padrões de Conectividade). Ele permite que aparelhos funcionem com qualquer ecossistema que tenha o certificado Matter, reduzindo assim a fragmentação e promovendo a interoperabilidade entre aparelhos e plataformas IoT de diferentes fornecedores. Ele está focado na padronização no nível da aplicação, no suporte ao Wi-Fi e Thread como meios de comunicação, e o apoio de grandes empresas de tecnologia o separa de outros protocolos, como o Zigbee. Apesar do número de aparelhos certificados com Matter ser ainda relativamente pequeno, sua crescente importância no espaço IoT o torna merecedor de avaliação por parte daqueles interessados em desenvolver soluções para casas inteligentes e IoT.

39. Modal

Avalie

Modal é uma plataforma como serviço (PaaS) que oferece processamento sob demanda sem a necessidade de usar a sua própria infraestrutura. A Modal permite a implantação de modelos de aprendizagem de máquina, tarefas massivas de processamento paralelo, filas de tarefas e apps web. Ela fornece uma abstração de contêiner que torna a mudança de implantação local para implantação em nuvem transparente, com recargas (reloads) "a quente", tanto localmente, quanto na nuvem. Ela remove implantações automaticamente, evitando a necessidade de limpeza manual. Mas pode também tornar as implantações persistentes.

A Modal foi escrita pela mesma equipe que desenvolveu o primeiro mecanismo de recomendação para o Spotify. Ela pode sustentar a stack AI/ML de ponta a ponta, e também oferece recursos GPU sob demanda, algo muito útil se você tem necessidades de processamento intensivos. Quer você esteja trabalhando em seu laptop, quer esteja trabalhando na nuvem, a Modal simplesmente funciona, fornecendo uma forma fácil e eficiente de implantar seus projetos.

40. Neon

Avalie

Neon é uma alternativa de código aberto ao AWS Aurora PostgreSQL. Os bancos de dados analíticos nativos de nuvem adotaram a técnica de separar o armazenamento dos nós de processamento para escalar elasticamente sob demanda. É difícil, porém, fazer o mesmo em um banco de dados transacional. O Neon consegue isso com seu novo mecanismo de armazenamento multitenant para PostgreSQL. Com alterações mínimas no código oficial do PostgreSQL, o Neon aproveita o AWS S3 para armazenamento de dados de longo prazo e escala elasticamente o processamento para cima ou para baixo (incluindo "scale-to-zero") para processar a informação. Essa arquitetura tem vários benefícios — incluindo cópias baratas e rápidas, sombreamento (copy-on-write) e branching. Estamos muito animados vendo essas inovações no PostgreSQL. Nossas equipes estão avaliando o Neon e recomendamos que você também o avalie.

41. OpenLineage

Avalie

O OpenLineage é um padrão aberto para coleta de metadados de linhagem para pipelines de dados, projetado para instrumentar trabalhos durante sua execução. Ele define um modelo genérico de execução, trabalho e entidades de conjunto de dados usando convenções de nomenclatura consistentes. O modelo de linhagem central é extensível ao definir facetas específicas para enriquecer essas entidades. O OpenLineage resolve o problema de interoperabilidade entre produtores e consumidores de dados de linhagem que, de outra forma, precisariam saber como se comunicar uns com os outros de várias maneiras. Embora exista o risco de ser só outro padrão sem adesão, o projeto ser da Linux Foundation AI & Data Foundation aumenta suas chances de ser adotado de maneira generalizada. Atualmente, o OpenLineage oferece suporte à coleta de dados para várias plataformas, como Spark, o Airflow e o dbt, embora as usuáriasprecisem configurar seus listeners. O suporte para consumidores de dados OpenLineage é mais limitado neste momento.

42. Passkeys

Avalie

O "fim das senhas" pode estar próximo, finalmente. Gerenciadas pela FIDO Alliance e apoiados pela Apple, Google e Microsoft, passkeys estão se aproximando da usabilidade convencional. Ao configurar um novo login com passkeys, um par de chaves é gerado: o site recebe a chave pública e a usuária fica com a chave privada. A verificação do login usa criptografia assimétrica. A usuária prova que está de posse da chave privada mas, ao contrário das senhas, ela nunca é enviada ao site. Nos dispositivos das usuárias, o acesso às senhas é protegido por biometria ou PIN.

As chaves podem ser armazenadas e sincronizadas nos ecossistemas das Big Techs, usando o iCloud Keychain da Apple, o Google Password Manager ou o Windows Hello. Na maioria dos casos, isso funciona apenas com versões recentes do sistema operacional e do navegador. O armazenamento de chaves no Windows Hello não é suportado no Windows 10. Felizmente, porém, o CTAP - Protocolo Cliente para Autenticador torna possível que as passkeys sejam mantidas em um dispositivo diferente daquele que cria a chave ou precisa dela para o login. Por exemplo, uma usuária cria uma chave de acesso para um site no Windows 10 e a armazena em um iPhone digitalizando um código QR. Como a chave é sincronizada via iCloud, a usuária pode fazer login no site a partir, digamos, de seu MacBook. As chaves também podem ser armazenadas em chaves de segurança de hardware, e o suporte para aplicativos nativos chegou no iOS e no Android. Apesar de alguns problemas de usabilidade – por exemplo, o Bluetooth precisa funcionar porque a proximidade do dispositivo é verificada quando um código QR é escaneado – vale a pena considerar passkeys. Sugerimos que você as experimente em passkeys.io para ter uma ideia de sua usabilidade.

43. Spin

Avalie

O Spin é uma plataforma de código aberto para desenvolver e executar microsserviços em WebAssembly (WASM). Nas edições anteriores do Radar, falávamos do WebAssembly no contexto dos navegadores, mas agora estamos vendo sua inserção no lado do servidor, devido às suas capacidades de fine-grained sandboxing, interoperabilidade entre linguagens e o recurso de hot reloading. A CLI do Spin, permite criar e distribuir rapidamente microsserviços WebAssembly para Rust, TypeScript, Python e TinyGo. Estamos animadas com Spin e recomendamos que você avalie essa plataforma com cuidado, à medida que for saindo das versões iniciais.

44. Denodo como ferramenta primária de transformação de dados

Evite

O Denodo é uma ferramenta de virtualização de dados, que tem por objetivo tornar mais fácil a exposição de dados de uma plataforma (de múltiplas fontes de dados subjacentes e através de uma variedade de interfaces), transformados de maneira segura e de uma forma amigável ao consumidor [de tais dados]. Transformações de dados dentro do Denodo podem ser definidas pela criação de bancos de dados virtuais e views, usando uma linguagem, similar ao SQL, chamada VQL, que é executada quando um usuário consulta o banco de dados virtual. Por trás das cortinas, o Denodo pode delegar as consultas ao banco de dados virtual para um ou vários dos bancos de dados subjacentes.

Apesar do Denodo facilitar a exposição inicial de dados em formato amigável ao consumidor, o desempenho degrada conforme níveis de views e bancos de dados virtuais são construídos uns sobre os outros, e consultas com múltiplos “joins” começam a impactar os bancos de dados subjacentes. Esses problemas são possíveis de resolver, mas exigem um conhecimento profundo do comportamento do produto e das opções de ajuste fino afinamento do desempenho. Devido a essas desvantagens, e dado seu suporte limitado a testes unitáriosde unidade, recomendamos não utilizar o Denodo como ferramenta primária de transformação de dados , usando em vez dele ferramentas como o Spark ou SQL (com o dbt) para suas necessidades de transformação de dados..

Ferramentas

Adote

45. DVC

Experimente

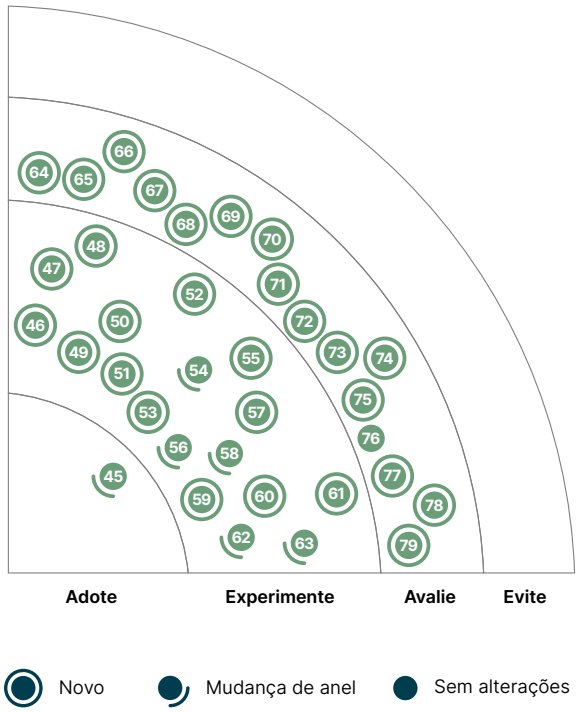
46. Akeyless
47. Apicurio Registry
48. EventCatalog
49. FOSSA
50. Gitleaks
51. Helmfile
52. Verificador de acessibilidade de acesso igualitário da IBM
53. Ktlint
54. Kubeflow
55. Mend SCA
56. Mozilla SOPS
57. Ruff
58. Soda Core
59. Steampipe
60. Terraform Cloud Operator
61. TruffleHog
62. Typesense
63. Vite

Avalie

64. axe Linter
65. ChatGPT
66. DataFusion
67. Deepchecks
68. Ferramentas de tradução de Tokens de design
69. Devbox
70. Evidently
71. Giskard
72. GitHub Copilot
73. iamlive
74. Kepler
75. Operador de segredos externos do Kubernetes
76. Kubeshark
77. Obsidian
78. Ory Kratos
79. Philips's self-hosted GitHub runner

Evite

—



45. DVC

Adote

DVC continua a ser nossa ferramenta preferida para gerenciar experimentos em projetos de ciência de dados. O fato de ser baseado no Git faz com que seja um território conhecido para as pessoas desenvolvedoras trazerem práticas de engenharia para o ecossistema de ciência de dados. A visão opinativa do DVC de um ponto de verificação do modelo encapsula cuidadosamente um conjunto de dados de treinamento, um conjunto de dados de teste, hiperparâmetros do modelo e o código. Ao tornar a reprodutibilidade uma preocupação de primeira classe, permite que a equipe viaje no tempo em várias versões do modelo. Nossas equipes usaram DVC com sucesso em produção para permitir entrega contínua para ML (CD4ML); pode ser conectada a qualquer tipo de armazenamento (incluindo AWS S3, Google Cloud Storage, MinIO e Google Drive). No entanto, com conjuntos de dados cada vez maiores, a criação de snapshots com base no sistema de arquivos pode se tornar particularmente cara. Quando os dados subjacentes mudam com rapidez, o DVC sobre um bom armazenamento com controle de versão permite rastrear os desvios do modelo ao longo de um período de tempo. Nossas equipes usaram efetivamente o DVC em formatos de armazenamento de dados como Delta Lake que otimiza o controle de versão (COW). A maioria de nossas equipes de ciência de dados configura DVC como uma tarefa do dia zero enquanto fazem o bootstrap de um projeto; por esse motivo, estamos felizes em movê-lo para Adote.

46. Akeyless

Experimente

Conforme mais organizações adotam a computação em nuvem, muitas começam a integrar simultaneamente vários provedores de nuvem, para maximizar a flexibilidade e minimizar a dependência do fornecedor. Contudo, gerenciar chaves e controles de acesso em diversos provedores de nuvem pode ser um desafio significativo, levando a uma maior complexidade e riscos de segurança. Akeyless é uma plataforma centralizada baseada na nuvem, que fornece gerenciamento unificado de segredos com uma variedade de vantagens para gerenciar segredos e dados confidenciais. Integra-se perfeitamente com diferentes provedores, simplificando a gestão de segredos e controles de acesso para monitorar e controlar quem tem acesso a dados sensíveis; com criptografia, controles de acesso, autenticação multifator e outros mecanismos de segurança, garante que apenas usuárias autorizadas possam acessar dados confidenciais. Além disso, fornece uma interface intuitiva para administração e monitoramento, proporcionando uma experiência de desenvolvedora e administração menos complexa e mais escalável.

47. Apicurio Registry

Experimente

Dentro de qualquer organização, produtores e consumidores de APIs precisam estar sincronizados acerca dos esquemas que serão usados para a comunicação entre eles. Especialmente à medida que o número de APIs e produtores e consumidores relacionados cresce na organização, o que provavelmente começa com a simples transmissão de esquemas entre as equipes pode vir a enfrentar desafios para escalar a solução. Diante desse problema, algumas de nossas equipes recorreram à Apicurio Registry, um registro centralizado de código aberto para vários tipos de esquemas e artefatos de API, incluindo especificações OpenAPI e esquemas Protobuf e Avro. A Apicurio Registry permite que as usuárias interajam com ela por meio de uma interface de usuária, assim como por uma API REST e um plugin Maven. Ela também oferece a opção de impor restrições de evolução do esquema, como compatibilidade com versões anteriores. Além disso, quando se trata de trabalhar com clientes Kafka, a Apicurio Registry é compatível com o Confluent Schema Registry. Embora nossas equipes tenham achado a documentação do Confluent Schema Registry mais útil, a Apicurio Registry atende às suas necessidades de uma fonte confiável para vários esquemas.

48. EventCatalog

Experimente

Hoje em dia, empresas frequentemente usam a transmissão de eventos como a fonte da verdade e como mecanismo de compatilhamento de informações em arquiteturas de microserviços. Isso cria a necessidade de padronizar tipos de eventos e compartilhar esses padrões com toda a empresa. Registros de schemas de eventos são normalmente implantados, mas as ofertas existentes tendem a ser especializadas em um único agente intermediário, tais como o Apache Kafka ou o Azure Event Hub. Eles também não chegam a proporcionar uma documentação robusta sobre os tipos de evento além de simples definições de schema. O EventCatalog é um projeto de código aberto que oferece algo que vemos muitas empresas desenvolvendo por si mesmas: um repositório amplamente acessível de documentação de eventos e schemas. Estes últimos descrevem o papel do evento no negócio, onde ele se encaixa no modelo de domínio do negócio, e quais serviços o assinam e o publicam. Se você está procurando uma forma de publicar documentação de eventos na sua organização, essa ferramenta pode poupar você do trabalho de desenvolver seu próprio sistema interno para isso.

49. FOSSA

Experimente

O FOSSA é uma ferramenta de conformidade de código aberto, que ajuda pessoas desenvolvedoras e equipes a determinarem de quais componentes de código aberto seu código-fonte depende e sob quais licenças estes componentes foram lançados. Essa informação é essencial para garantir a conformidade com várias licenças de código aberto e manter a Lista de Materiais de Software. O FOSSA se integra com ferramentas de gerenciamento de dependências de várias stacks de tecnologia, para identificar quais componentes de código aberto são usados em um projeto. A ferramenta também destaca qualquer problema de licenças, baseado nas políticas da empresa, e gera relatórios sobre isso. Alguns recursos fundamentais do FOSSA incluem sua capacidade de se integrar a fluxos de trabalho de desenvolvimento, tal como CI, e executar monitoramento de conformidade em tempo real. Muitos de nossos clientes e equipes consideraram o FOSSA uma ferramenta valiosa e eficaz.

50. Gitleaks

Experimente

O Gitleaks é uma ferramenta SAST (teste de segurança estático de aplicação) de linha de comando e código aberto, usada para detectar e prevenir que dados secretos (tais como senhas, chaves de APIs e tokens) sejam incluídos no código de maneira insegura (hardcoded) em repositórios Git. Ele pode ser usado como um hook de pré-commit no Git ou no pipeline CI/CD. Nossas equipes consideraram Gitleaks mais responsivo que outras ferramentas de varredura de dados secretos. Gitleaks usa expressões regulares e codificação de strings por entropia para detectar segredos. Em nossa experiência, a flexibilidade para fornecer uma regex personalizada junto com codificação por entropia permitiu às equipes categorizar melhor os dados secretos, com base em suas necessidades. Por exemplo, em vez de categorizar todas as chaves de API como “generic-api-key,” ele permitiu categorizações específicas, tal como “cloud provider key”.

51. Helmfile

Experimente

O Helmfile é uma ferramenta de linha de comando, de código aberto, e uma especificação declarativa para gerenciar e instalar uma coleção de charts Helm. Você pode usá-la para ajudar no controle de versão dos arquivos de valores do Helm, dos charts usados e de outras substituições. Ele permite fluxos de trabalho CI/CD com charts Helm e ajuda a criar ambientes reproduzíveis. Usamos o Helmfile para gerenciar implantações complexas com várias dezenas de charts Helm, e descobrimos que ele simplifica o fluxo de trabalho da implantação.

52. IBM Equal Access Accessibility Checker

Experimente

Torna-se mais barato corrigir defeitos quando eles são detectados mais cedo. É por isso que sempre tentamos obter o feedback mais rápido possível para as pessoas desenvolvedoras, na forma de análise estática, testes de unidade ou testes de ponta a ponta executados no ambiente local. Acessibilidade não é exceção e é por isso que apresentamos ferramentas como Lighthouse, axe-core e axe Linter no passado. Quando se trata de testar automaticamente páginas da web que já estão implementadas em produção, uma de nossas equipes optou por usar o IBM Equal Access Accessibility Checker em uma comparação direta. Embora ainda estejamos avaliando os resultados, podemos dizer que ele oferece uma maneira eficiente de testar as páginas depois de implantadas. Enfatizamos que isso deve ser usado para aumentar, não substituir, os testes automatizados iniciais vfeitos pela pessoa desenvolvedora. A ferramenta é distribuída sob uma licença Creative Commons, e é gratuita para uso sob aquelas restrições.

53. Ktlint

Experimente

O ecossistema Kotlin continua a evoluir e as nossas equipes relatam experiências positivas com o Ktlint, um linter e formatador simples e fácil de configurar para código Kotlin. Gostamos da formatação de código opinativa e automatizada, pois permite que os desenvolvedores se concentrem mais no que o código faz do que em como se parece; essa ferramenta permite que as equipes de desenvolvimento mantenham consistência e legibilidade em suas bases de código com eficiência, reduzindo a probabilidade de merges confusos devido a problemas de formatação. O Ktlint pode ser facilmente configurado para executar por hooks de pré-commit, afetando apenas os arquivos alterados e resultando em processos de integração mais rápidos.

54. Kubeflow

Experimente

O Kubeflow é uma plataforma de aprendizado de máquina (ML) nativa do Kubernetes, que simplifica a criação, o treinamento e a implantação de ciclos de vida de modelos para infraestruturas diversas. Nós utilizamos extensivamente o Pipelines para codificar fluxos de trabalho de ML para vários modelos, cobrindo casos de uso de experimentação, treinamento e entrega. Além do Pipelines, Kubeflow vem com múltiplos componentes. Entre estes, achamos o ajuste de hiper-parâmetros com o Katib e multi-tenancy bastante úteis.

55. Mend SCA

Experimente

Mend SCA (análise de composição de software), antes chamado Whitesource, ajuda a detectar dependências de software de código aberto, identificando se estão atualizados, contêm falhas de segurança ou apresentam requisitos de licenciamento. Nossas equipes tiveram uma boa experiência com a integração do Mend SCA no caminho para a produção. Desde a integração do IDE, gerando um PR automático com base em um problema identificado, até a integração no pipeline CI/CD, essa ferramenta oferece uma ótima experiência para a pessoa desenvolvedora. Outras ferramentas SCA populares, como Snyk, são comparáveis e também valem a pena explorar para suas necessidades de segurança.

56. Mozilla SOPS

Experimente

Nosso conselho, quando se trata de gerenciamento de segredos, sempre foi dissociá-los do código-fonte. Entretanto, equipes muitas vezes se veem confrontadas com uma escolha entre a automação total (no espírito da infraestrutura como código) versus algumas etapas manuais (usando ferramentas como vaults) para gerenciar, criar e rotacionar segredos. Por exemplo, nossas equipes usam o SOPS para gerenciar credenciais de inicialização da infraestrutura. Em algumas situações, entretanto, é impossível remover segredos de repositórios de código legado. Para tais necessidades, achamos o Mozilla SOPS uma boa escolha para encriptar segredos em arquivos de texto. O SOPS se integra com as keystores gerenciadas em nuvem, tais como a AWS e GCP Key Management Service (KMS) ou a Azure Key Vault, como fontes de chaves de encriptação. Ele também funciona em várias plataformas e suporta chaves PGP.

57. Ruff

Experimente

Ruff é um novo linter para Python. Para nós, a questão não é se vamos ou não usar um linter, mas qual linter vamos usar. E há muitas alternativas para Python. Ruff se destaca por duas razões: a experiência inicial e a velocidade. Ele tem mais de 500 regras embutidas, e substitui imediatamente o Flake8, incluindo muitos dos plugins deste último. As alegações da equipe por trás do Ruff sobre seu desempenho foram confirmadas por nossa experiência. Ele é pelo menos uma ordem de magnitude mais rápido que outros linters, o que é um imenso benefício, por ajudar a reduzir o tempo de compilação em grandes bases de código.

58. Soda Core

Experimente

O Soda Core é uma ferramenta de código aberto para qualidade de dados e observabilidade. Nossas equipes a usaram para validar os dados assim que chegam a um sistema, antes e depois de transformações, e configurar verificações automatizadas de monitoramento de anomalias. Estamos felizes com a SodaCL, a DSL para escrever verificações de dados na Soda Core, pois ajuda as pessoas da equipe, além das pessoas engenheiras de dados, a escrever as verificações de qualidade. No geral, nossa experiência usando a Soda Core para encontrar e resolver problemas de dados em grande escala tem sido positiva.

59. Steampipe

Experimente

Steampipe é uma ferramenta de código aberto que permite consultar instantaneamente serviços de nuvem como AWS, Azure e GCP utilizando SQL. Com mais de 100 plugins e suporte integrado para criação de painéis, o Steampipe torna trivial conectar os dados ativos de configuração na nuvem com conjuntos de dados internos ou externos e criar painéis de segurança ou conformidade. Gostamos de trabalhar com Steampipe e criamos vários desses painéis com configurações de nuvem AWS.

60. Terraform Cloud Operator

Experimente

Mais e mais equipes estão usando o padrão Kubernetes Operators para gerenciar seus clusters Kubernetes. Costumávamos recomendar a Crossplane para isso, e agora temos uma ferramenta alternativa, a Terraform Cloud Operator para Kubernetes. Essa ferramenta integra o Terraform Cloud e o Kubernetes, estendendo o plano de controle do Kubernetes para permitir o gerenciamento do ciclo de vida da nuvem e das infraestruturas locais por meio de manifestos do Kubernetes. Nossa equipe a usa para provisionar recursos de namespaces Kubernetes e RoleBindings para instâncias de banco de dados em nuvem e outros recursos SaaS. Gostamos bastante porque ela aproveita o módulo Terraform, que é uma camada de abstração mais familiar para operarmos os recursos da nuvem.

61. TruffleHog

Experimente

TruffleHog é uma ferramenta SAST (teste de segurança de aplicativo estático) de código aberto, para detectar segredos em várias fontes. Embora os repositórios GitHub e GitLab sejam os casos de uso mais populares, ela também podem ser usada para escanear buckets de armazenamento em nuvem como S3 e GCS, arquivos e diretórios locais e logs do CircleCI. As pessoas desenvolvedoras podem configurar o TruffleHog como um hook de pré-commit ou verificar o histórico de repositórios existentes em toda uma organização no GitHub, para detectar segredos. A ferramenta suporta a detecção de padrões regex personalizados que foram considerados bastante úteis, mesmo em seu estágio alfa atual. O TruffleHog também possui uma versão corporativa, mas nossas pessoas desenvolvedoras acharam a versão de código aberto fácil de configurar e suficiente para os casos de uso mais comuns. A ferramenta tem uma comunidade muito ativa que adiciona recursos regularmente.

62. Typesense

Experimente

Typesense é um mecanismo de busca de código aberto tolerante a erros de digitação, otimizada para fornecer experiências de busca de baixa latência e alto desempenho. Se você está desenvolvendo uma aplicação de busca sensível à latência e com um índice de busca que possa caber na memória, o Typesense é uma alternativa poderosa. Nossas equipes usam o Typesense em clusters de múltiplos nós e alta disponibilidade, para distribuir a carga de trabalho e assegurar que infraestruturas de busca críticas sejam resilientes. Elas têm tido uma boa experiência com o Typesense em produção, e por essa razão o movemos para Experimente.

63. Vite

Experimente

Vite, uma ferramenta para construção de front-end, continuou a amadurecer e crescer em popularidade desde que a apresentamos no anel Avalie do último Radar. Está rapidamente se tornando a escolha padrão entre nossas equipes ao iniciar um novo projeto de front-end. Vite fornece um conjunto de padrões para construir, agrupar e gerenciar dependências em aplicativos que dependem de módulos ES no navegador. Por aproveitar a velocidade nativa do esbuild e do bundler Rollup, o Vite melhora significativamente a experiência da pessoa desenvolvedora front-end. Além disso, quando usada com React, Vite oferece uma alternativa atraente para o robusto, mas quase extinto Create React App. Vite depende de módulos ES e, ao contrário da maioria das ferramentas mais antigas, não oferece shimming ou polyfills, o que significa que você precisa de uma estratégia diferente para navegadores mais antigos que não oferecem suporte a módulos ES. Nos casos em que navegadores mais antigos precisam ser suportados, algumas de nossas equipes importam os polyfills no nível do módulo para que Vite possa ser usada de forma consistente em todos os ambientes.

64. axe Linter

Avalie

Está se tornando cada vez mais fácil para as pessoas desenvolvedoras detectar problemas de acessibilidade no início do processo de desenvolvimento. Enquanto ferramentas como o axe-core verificam o código em busca de problemas de acessibilidade em seus pipelines, a extensão do VSCode axe Linter ajuda a encontrá-los mesmo antes disso, enquanto o código é escrito. A grande maioria dos problemas de acessibilidade se enquadra em categorias que poderiam ser evitadas por meio de testes automatizados e o uso de analisadores de código com feedback imediato como este.

65. ChatGPT

Avalie

ChatGPT é uma ferramenta interessante, com potencial para ser útil em vários aspectos do processo de construção de software. Como um grande modelo de linguagem (LLM, sigla em inglês para “Large Language Model”) que “leu” bilhões de páginas da web, o ChatGPT pode fornecer perspectivas adicionais e auxiliar em diferentes tarefas, desde a geração de ideias e requisitos até a criação de código e testes. Sua capacidade de trabalhar em várias partes do ciclo de vida do software o torna uma ferramenta versátil, que pode melhorar a eficiência e reduzir erros no processo de desenvolvimento. O GPT4, o LLM que impulsiona o ChatGPT, agora também tem a capacidade de se integrar a ferramentas externas, como repositórios de gerenciamento de conhecimento, ambientes de programação em área restrita ou busca na web. Por enquanto, achamos que o ChatGPT é melhor usado como entrada para um processo, para ajudar com o primeiro rascunho de uma história ou criando o código padrão repetitivo de uma tarefa de programação, em vez de uma ferramenta que produz resultados “completos”.

Há preocupações acerca de propriedade intelectual e privacidade de dados com essas ferramentas de IA, incluindo algumas questões legais não resolvidas, por isso recomendamos que as organizações busquem o aconselhamento de suas equipes jurídicas antes de usar a ferramenta. Algumas de nossas clientes já começaram a experimentar o ChatGPT em vários estágios do ciclo de vida do software e incentivamos outros a explorar a ferramenta e avaliar seus benefícios potenciais. Esperamos que, como o GitHub Copilot, uma oferta “para negócios” esteja disponível em breve, o que pode aliviar as preocupações com a propriedade intelectual.

66. DataFusion

Avalie

O DataFusion faz parte da exploração, pela comunidade de dados, do desempenho, da segurança da memória e da concorrência do Rust aplicados ao processamento de dados. Ele tem semelhanças com o Polars, isto é, uma API de DataFrame conhecida em Rust (com binding para Python), o uso do Apache Arrow por debaixo dos panos e suporte a SQL. Embora tenha sido projetado principalmente para execução de processo único, o suporte ao processamento distribuído está sendo desenvolvido no Ballista. Acreditamos que as bibliotecas Rust para processamento de dados são um espaço em evolução que vale a pena seguir e explorar, e o DataFusion faz parte disso.

67. Deepchecks

Avalie

Conforme a aprendizagem de máquina se aproxima do uso comum, as práticas de testes automatizados de modelos, validação de dados de treinamento e observação do desempenho dos modelos em produção estão amadurecendo. Cada vez mais, essas verificações automáticas têm sido incorporadas em pipelines de entrega contínua ou usadas com modelos em produção, para detectar desvios e desempenho do modelo. Um conjunto de ferramentas com capacidades similares ou sobrepostas emergiu para lidar com as várias etapas desse processo. (A Giskard e a Evidently também aparecem neste volume). A Deepchecks é outra dessas ferramentas, e está disponível como uma biblioteca Python de código aberto, podendo ser invocada de um pipeline de codificação através de um grande conjunto de APIs. Um recurso singular dessa ferramenta é sua capacidade de lidar com dados tabulares e imagens, com um módulo para linguagem de dados na versão alfa. No momento, nenhuma ferramenta sozinha é capaz de oferecer a variedade de testes e barreiras em todo o pipeline de ML. Recomendamos avaliar Deepchecks para o seu nicho de aplicação específico.

68. Ferramentas de tradução de tokens de design

Avalie

Tokens de design são um mecanismo útil para definir elementos padronizados em sistemas de design. Mas manter aqueles elementos de design consistentes através de mídias como aplicativos móveis ou frameworks web é uma tarefa cada vez mais colossal. As ferramentas de tradução de tokens de design simplificam esse problema, organizando e automatizando a transformação da descrição do token (em YAML ou JSON) no código que efetivamente controla a renderização em uma dada mídia, como CSS, componentes React ou HTML. Style Dictionary é um exemplo de código aberto amplamente utilizado, e que se integra bem em pipelines de compilação automatizados. Mas há também alternativas comerciais, como o Specify.

69. Devbox

Avalie

Devbox fornece uma interface acessível para a criação de ambientes de desenvolvimento reproduzíveis e específicos de cada projeto, aproveitando os recursos do gerenciador de pacotes Nix. Nossas equipes o usam para eliminar incompatibilidades de versão e configuração em seus ambientes de desenvolvimento, e o adoram pela facilidade de uso. Devbox suporta shell hooks, scripts personalizados e geração de devcontainer.json para integração ao VSCode.

70. Evidently

Avalie

Evidently é uma ferramenta Python de código aberto projetada para ajudar a monitorar a construção de modelos de aprendizagem de máquina, para garantir sua qualidade e uma operação estável em produção. Ela pode ser usada em vários estágios do ciclo de vida do modelo: como um painel para revisar o modelo em um bloco de notas, como parte do pipeline ou como um serviço de monitoramento após a implantação. Especialmente focada na detecção de desvios do modelo, a Evidently também oferece recursos como qualidade do modelo, inspeção da qualidade dos dados e detecção de desvios do alvo. Além disso, tem muitas métricas embutidas, visualizações associadas e testes que são facilmente combinados em um relatório, painel ou em um pipeline orientada a testes.

71. Giskard

Avalie

Giskard é uma ferramenta de código aberto projetada para ajudar organizações a criar modelos de IA mais robustos e éticos, fornecendo capacidades de garantia de qualidade com foco na explicabilidade e na equidade. Ela facilita a cooperação entre partes interessadas técnicas e não-técnicas, permitindo que avaliem os modelos de forma colaborativa e estabeleçam critérios de aceitação baseados na evitação de vieses e outras métricas de qualidade essenciais. A Giskard garante que os resultados produzidos pelo modelo estejam melhor alinhados aos objetivos do negócio e ajuda a resolver problemas de qualidade antes da implantação em produção.

72. GitHub Copilot

Avalie

O GitHub Copilot é uma Inteligência Artificial (IA) assistente de programação, criado em uma colaboração entre a Microsoft e a OpenAI. Ele usa modelos de aprendizado de máquina (ML) para gerar sugestões baseadas no contexto no qual a pessoa desenvolvedora está trabalhando. Entre seus recursos está uma forte integração com a IDE, e ele usa uma base de código existente e um editor de contexto para criar as sugestões. Apesar de ter sido chamado de "seu par IA de programação", nós não chamamos o que ele faz de "pareamento" – nós provavelmente o descreveríamos como uma espécie Stack Overflow superalimentado e sensível ao contexto. Quando consegue prever corretamente o que uma pessoa desenvolvedora está tentando fazer, o Copilot pode ser uma ferramenta poderosa para ajudar a fazer as coisas. Como todas as IAs baseadas em grandes modelos de linguagens (LLMs), entretanto, ele tem uma tendência de ludibriar ao sugerir o uso de APIs plausíveis, mas inexistentes, e pode introduzir bugs através de algoritmos sutilmente defeituosos. Tivemos sucesso em gerar código a nível de linha, bloco e método, bem como na criação de testes e configurações de infraestrutura. Um detalhe interessante, ele funciona melhor se você utilizar boas práticas de nomenclatura, encorajando a construção de código mais legível.

As habilidades das ferramentas de IA estão progredindo rapidamente, e achamos sensato que as organizações as experimentem. Alguns discursos de vendas do Copilot alegaram ganhos de eficiência muito altos, mas nós continuamos céticas: afinal, escrever código não é a única coisa que pessoas desenvolvedoras fazem, e mais, é notoriamente difícil medir a produtividade de desenvolvedoras. Dito isso, o Copilot é uma ferramenta muito barata; se ele oferecer qualquer ganho de produtividade, terá justificado a aquisição. O Copilot X – em pré-venda no momento em que essa nota é escrita – oferece funcionalidade adicional e integração com o fluxo de trabalho de criação de software. O Copilot tem uma versão "corporativa" que oferece maior transparência sobre as questões de propriedade intelectual, bem como a possibilidade de gerenciar os recursos da ferramenta de forma centralizada para toda a organização. Achamos esses recursos críticos para adoção corporativa.

73. iamlive

Avalie

Criar precisamente as mínimas políticas IAM viáveis que desejamos na AWS, de acordo com o princípio do menor privilégio, pode ser uma longa jornada de tentativa e erro. iamlive pode encurtar consideravelmente tal jornada. Ele monitora as chamadas da CLI da AWS feitas desde uma máquina e determina as políticas necessárias para executar essas chamadas. A ferramenta gera um documento de política com declarações, ações, princípios e recursos, que pode ser usado como um bom ponto de partida. Achamos a ferramenta particularmente útil para criar as políticas necessárias em pipelines de CI/CD que provisionam infraestrutura, reduzindo as idas e vindas habituais após uma execução do Terraform ter falhado porque as políticas de permissões para aquele papel no IAM eram insuficientes.

74. Kepler

Avalie

Medir o consumo de energia é um passo importante para que equipes reduzam a pegada de carbono de seu software. O Cloud Carbon Footprint (CCF) estima o consumo de energia baseada nos dados de cobrança e utilização obtidos da APIs da nuvem. Kepler — acrônimo de Kubernetes-based Efficient Power Level Exporter (Exportador de Nível de Energia Eficiente baseado em Kubernetes) — vai um passo além: ele usa contadores de software via RAPL, ACPI e nvml para medir o consumo de energia por recursos de hardware, e usa uma abordagem baseada na eBPF para atribuir consumo de energia a processos, containers e pods Kubernetes. O consumo é então convertido em estimativas de uso de energia através de um modelo personalizado de aprendizado de máquina (ML) e de dados de referência de SPEC Power. Por fim, o relatório de consumo de energia a nível de pods fica disponível como métricas do Prometheus. Em casos onde o Kubernetes está rodando em máquinas virtuais, por exemplo, quando não está usando instâncias de máquinas concretas, o Kepler usa cgroups para estimar o consumo de energia. Temos grande experiência com o CCF, e podemos atestar sua utilidade, mas a abordagem do projeto Kepler nos deixou intrigados.

75. Operador de segredos externos do Kubernetes

Avalie

O Operador de segredos externos do Kubernetes permite que provedores de segredos externos sejam integrados ao Kubernetes. Ele lê a API do provedor externo e injeta o resultado em um segredo do Kubernetes. O operador trabalha com uma grande variedade de ferramentas de gerenciamento de segredos, incluindo algumas que apresentamos em edições anteriores do Radar. Nossas equipes descobriram que ele simplificou o uso de segredos ao trabalhar com o Kubernetes, permitindo o uso de um único ponto de armazenamento para todo o projeto.

76. Kubeshark

Avalie

Kubeshark é um visualizador de tráfego de API para o Kubernetes. Até novembro de 2022 ela era conhecida como Mizu. Diferente de outras ferramentas, Kubeshark não exige instrumentalização ou mudanças no código. Ela roda como um DaemonSet para injetar um contêiner no nível do nó do seu cluster Kubernetes e executa operações semelhantes a tcpdump. Nós a achamos útil como ferramenta de depuração, pois ela consegue observar todas as comunicações de APIs em múltiplos protocolos (REST, gRPC, Kafka, AMQP e Redis) em tempo real.

77. Obsidian

Avalie

A gestão do conhecimento é fundamental para as pessoas que trabalham com tecnologia, pois precisamos estar constantemente aprendendo e nos mantendo atualizadas com os mais recentes desenvolvimentos tecnológicos. Recentemente, ferramentas como Obsidian e o Logseq surgiram na categoria de ferramentas de anotações que suportam notas interligadas, para formar um grafo de conhecimento, enquanto as armazenam em arquivos markdown em um diretório local, permitindo assim que as usuárias possuam seus dados. Essas ferramentas ajudam as usuárias a organizar e vincular suas anotações de maneira flexível e não linear.

Obsidian tem um rico repositório de plugins da comunidade. Alguns que chamaram nossa atenção, em especial, são o Canvas, semelhante a uma versão local do Miro ou Mural, e o Dataview, que efetivamente trata suas notas como um banco de dados e fornece uma linguagem de consulta para filtrar, classificar e extrair dados de suas notas markdown.

78. Ory Kratos

Avalie

Já avaliamos o Ory Hydra como uma solução OAuth2 auto-hospedada, e o feedback da equipe tem sido bom. Desta vez, nos voltamos para Ory Kratos, um sistema de gerenciamento de usuário e identidade, voltado primeiramente a API, amigável a pessoa desenvolvedora e fácil de personalizar. Ela já fornece funções comuns que queremos alcançar em um sistema de gerenciamento de identidade, incluindo login e registro de autoatendimento, autenticação multifator (MFA/2FA), verificação e recuperação de conta. Como a Hydra, a Kratos é headless e exige que as pessoas desenvolvedoras construam a interface da usuária, o que dá mais flexibilidade à equipe. As desenvolvedoras também podem personalizar o esquema de identidade para atender a diferentes contextos de negócios. Kratos não tem nenhuma dependência externa além do banco de dados e é fácil de implantar e dimensionar em diferentes ambientes de nuvem. Se você precisa construir um sistema de gerenciamento de usuários, recomendamos que experimente Kratos.

79. Philips's self-hosted GitHub runner

Avalie

Apesar dos executores do GitHub Actions atenderem a uma enorme variedade dos runtimes mais comuns, algumas vezes é necessário algo mais específico para um caso de uso individual, isto é, o runtime de uma linguagem menos comum ou uma configuração de hardware singular. É nesse momento que um executor auto-hospedado é necessário. Philips's self-hosted GitHub runner é um módulo de Terraform que permite criar executores personalizados em instâncias spot do AWS EC2. Como um pouco do gerenciamento de ciclo de vida do GitHub Actions se perde quando você mesmo hospeda seus executores, o módulo também cria um conjunto de Lambdas para esse fim. Eles fazem o serviço pesado de escalar os executores para aumentar ou abaixar o processamento, conforme necessário. Isso ajuda a administrar os custos, e permite tornar os executores efêmeros, uma boa prática para melhorar a reprodutibilidade e a segurança. Se você precisa hospedar seus executores, há muitas coisas que podem ser esquecidas ao se desenvolver esse serviço do zero. Em vez disso, procure ferramentas como essa.

Linguagens e Frameworks

Adote

80. Gradle Kotlin DSL
81. PyTorch

Experimente

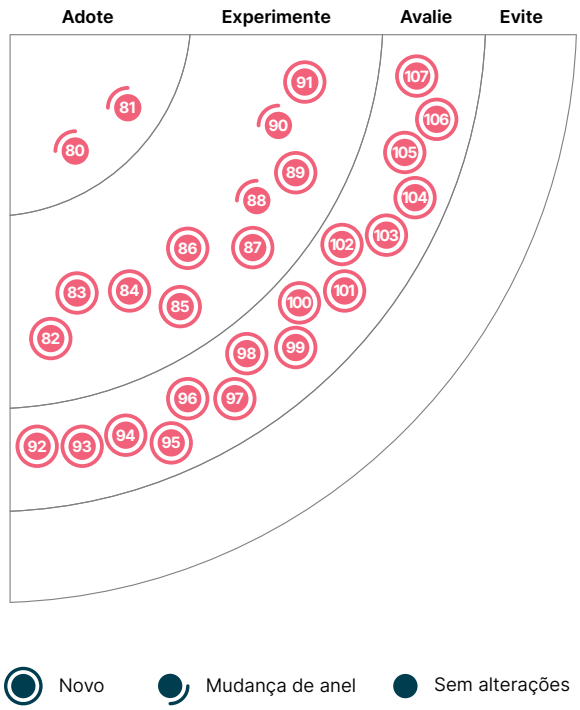
82. dbt-unit-testing
83. Jetpack CameraViewfinder
84. Jetpack DataStore
85. Mikro ORM
86. Preferência de linguagem por app
87. Quarto
88. River
89. Stencil
90. Synthetic Data Vault
91. Vitest

Avalie

92. .NET 7 Native AOT
93. .NET MAUI
94. dbt-expectations
95. Directus
96. Ferrocene
97. Flutter para dispositivos embarcados
98. Fugue
99. Galacean Engine
100. LangChain
101. mljar-supervised
102. nanoGPT
103. pandera
104. Qwik
105. SolidJS
106. Turborepo
107. WebXR Device API

Evite

—



80. Gradle Kotlin DSL

Adote

Nossas equipes agora veem Gradle Kotlin DSL como padrão para iniciar novos projetos usando Gradle, preferindo-o ao Groovy. As equipes que já usam Groovy devem considerar a migração. Kotlin fornece melhor suporte para refatoração e edição mais simples em IDEs e nossas equipes relatam que produz código mais fácil de ler e manter. Dado que alguns IDEs agora oferecem suporte à migração, deve ser relativamente rápido experimentar a substituição do Groovy existente. Em algumas situações, Kotlin pode ser mais lento que Groovy; no entanto, para muitos projetos, é improvável que isso afete a equipe.

81. PyTorch

Adote

PyTorch continua a ser nossa escolha de framework de aprendizado de máquina (ML). A maioria de nossas equipes prefere o PyTorch ao TensorFlow. O PyTorch expõe o funcionamento interno do ML, algo que o TensorFlow oculta, facilitando a depuração. Com grafos computacionais dinâmicos, a otimização do modelo é muito mais fácil se comparada a qualquer outro framework de ML. A ampla disponibilidade de trabalhos de pesquisa sobre modelos de última geração (SOTA - State-of-the-Art) e sobre a facilidade de implementação fazem o PyTorch se destacar. Quando se trata de ML com grafos, o PyTorch Geometric é um ecossistema mais maduro e nossas equipes tiveram ótimas experiências com ele. O PyTorch gradualmente também preencheu lacunas no tocante a implantação e dimensionamento de modelos; nossas equipes usaram o TorchServe para oferecer modelos pré-treinados com sucesso em produção, por exemplo. Com muitas equipes adotando o PyTorch como padrão para suas necessidades de aprendizagem profunda de ponta a ponta, recomendamos alegremente a adoção do PyTorch.

82. dbt-unit-testing

Experimente

dbt-unit-testing é um dbt que permite escrever testes de unidade para um modelo e sua lógica, simulando suas dependências. Isso traz o rigor de engenharia do feedback rápido de desenvolvimento, para o ecossistema de dados. Nossas equipes usam este pacote com o Snowflake, para praticar o desenvolvimento orientado a testes (TDD), embora só seja viável para transformações simples. A biblioteca certamente tem arestas quando se trata de depurar execuções de testes que falharam, mas a capacidade de escrever testes de unidade em transformadores à medida que desenvolvemos o modelo resultou em uma experiência elegante para as pessoas desenvolvedoras.

83. Jetpack CameraViewfinder

Experimente

Quando acrescentam funcionalidades de câmera aos apps Android, as pessoas desenvolvedoras precisam estar atentas a certas armadilhas conhecidas. A recente API Jetpack CameraViewfinder melhora significativamente a experiência da desenvolvedora nessa área. Internamente ela usa uma TextureView ou uma SurfaceView, para exibir o feed da câmera e aplica transformações para mostrar o visor corretamente, corrigindo o aspecto, a escala e a rotação onde for necessário. Também são fornecidos layouts otimizados para aparelhos dobráveis. Apesar de não ser um recurso principal ou essencial, destacamos isso aqui para assegurar que as equipes estejam cientes de sua existência.

84. Jetpack DataStore

Experimente

Jetpack DataStore é uma nova solução de armazenamento de dados para armazenar dados de forma assíncrona, consistente e transacional. Ele tem duas implementações: Preferences DataStore, para pares chave-valor sem tipo definido e Proto DataStore, para tipos de dados complexos usando Protobufs. Por padrão, ele é usado com corrotinas do Kotlin e Flow, mas suporte adicional para RXJava 2 e 3 está disponível. A documentação recomenda que você considere migrar para o DataStore se estiver usando SharedPreferences no momento, e nós concordamos com essa recomendação.

85. Mikro ORM

Experimente

Mikro ORM é um framework de mapeamento objeto-relacional (ORM) que possui uma interessante abordagem centrada em TypeScript. Aproveitando TypeScript em toda a stack, ele oferece uma experiência de desenvolvimento consistente do navegador ao back-end, tornando mais fácil para as pessoas desenvolvedoras escrever e manter o código. Notavelmente, o desempenho do Mikro ORM é excelente, permitindo uma execução rápida de consultas e minimizando a latência. Mas ainda que o Mikro ORM ofereça recursos atraentes, é essencial ter em mente as advertências gerais associadas aos mapeadores objeto-relacionais. As estruturas ORM são geralmente complexas e oferecem vazamento de abstração em um armazenamento de dados relacional, e, portanto, usar uma é sempre um equilíbrio de compensações.

86. Preferência de linguagem por app

Experimente

Muita gente fala mais de uma língua, e utiliza línguas diferentes em diferentes contextos. Aparelhos e plataformas capazes de executar apps em geral pedem para pessoas usuárias que selecionem um idioma para o sistema, e então fazem os apps seguirem aquela decisão. Para celulares, em especial, as usuárias podem preferir que certos apps rodem em um idioma diferente do padrão do sistema; a Apple introduziu há algum tempo o ajuste de idioma por app no iOS. As pessoas desenvolvedoras de apps Android, entretanto, precisavam implementar uma solução própria em seus apps, se quisessem oferecer essa opção – até agora. O Android 13 introduziu uma nova configuração do sistema, preferência de linguagem por app, e uma API pública, facilitando que desenvolvedores ofereçam esse recurso. Para compatibilidade com versões anteriores, APIs equivalentes estão disponíveis na AndroidX, via AppCompatDelegate. Encorajamos as pessoas desenvolvedoras a substituir suas soluções personalizadas e usar esse recurso em seus apps.

87. Quarto

Experimente

Quarto é um sistema de código aberto para publicação científica e técnica. Com ele, podemos criar notebooks computacionais que permitem escrever documentos em markdown, incorporar código e enviar a saída desse código para o documento final. Ele pode ser usado para criar relatórios de análise de dados reproduzíveis e personalizáveis, que podem ser facilmente compartilhados em uma variedade de formatos. Nossas equipes de ciência de dados usaram Quarto para compartilhar relatórios de análise de dados contendo visualizações (plotadas) e tabelas. Elas gostaram de poder usar R e Python para gerar esses relatórios dinâmicos e então exportá-los em HTML para compartilhar com todas as partes envolvidas nos projetos. Se você está buscando compartilhar suas pesquisas e análises dentro ou fora de sua organização, recomendamos avaliar Quarto.

88. River

Experimente

No centro de muitas abordagens à aprendizagem de máquina está a criação de um modelo a partir de um conjunto de dados de treinamento. Após o modelo ser criado, ele pode ser utilizado repetidas vezes. Entretanto, o mundo não é estacionário, e frequentemente o modelo precisa mudar em função do surgimento de novos dados. Simplesmente refazer a etapa de criação do modelo pode ser um processo lento e caro. A aprendizagem incremental visa resolver esse problema, tornando possível aprender a partir de fluxos de dados de forma incremental para reagir mais rápido a mudanças. Adicionalmente, os requisitos de memória e processamento são mais baixos e mais previsíveis. Nossa experiência prática com River continua sendo positiva. O Vowpal Wabbit, que pode ser uma alternativa, tem uma curva de aprendizagem muito mais íngreme, e a API similar à Scikit oferecida por River o torna mais acessível a cientistas de dados.

89. Stencil

Experimente

Stencil é uma biblioteca que permite às pessoas desenvolvedoras criar componentes da Web reutilizáveis usando ferramentas bem estabelecidas, como o TypeScript, o JSX e o JSDoc. De acordo com as experiências de nossas equipes, Stencil é uma escolha muito boa para criar sistemas de design independentes de plataforma. Para os poucos navegadores que não oferecem suporte a recursos de navegadores modernos, o Stencil garante a compatibilidade ao fazer polyfill de recursos não compatíveis e APIs sob demanda.

90. Synthetic Data Vault

Experimente

Synthetic Data Vault (SDV) é um ecossistema de bibliotecas para geração de dados sintéticos, que podem aprender a distribuição de um conjunto de dados para gerar dados sintéticos com o mesmo formato e propriedades estatísticas da fonte. No passado, falamos sobre as desvantagens de usar dados de produção em ambientes de teste. No entanto, as nuances da distribuição dos dados em produção dificilmente podem ser replicadas manualmente, resultando em defeitos e surpresas. Tivemos boas experiências usando SDV para gerar grandes volumes de dados para testes de desempenho. O SDV se sai bem com a modelagem de uma tabela única. No entanto, o tempo de geração de dados aumenta consideravelmente conforme o número de tabelas com restrições de chave estrangeira aumenta. Apesar disso, SDV é uma ótima promessa para testes de desempenho local. É uma boa ferramenta para geração de dados sintéticos e vale a pena considerar para suas necessidades de teste.

91. Vitest

Experimente

O Vitest é um framework de teste de unidade para JavaScript. Até agora, muitas equipes confiavam no Jest mas Jest não funciona bem com Vite, uma ferramenta moderna de construção de front-end. Usar Jest e Vite juntos forçou as equipes a criar dois pipelines - um para compilação e desenvolvimento e outro para testes de unidade - o que exigia uma configuração duplicada e tediosa dos pipelines. Esses problemas são resolvidos com Vitest. Ele é projetado especificamente para Vite e usa-o como um bundler. Como recurso adicional, o Vitest possui APIs compatíveis com Jest, o que torna possível usá-lo como substituto do Jest em várias configurações de compilação. No entanto, usar Vite e Vitest juntos fornece uma melhor experiência à pessoa desenvolvedora e, embora Vitest seja rápido, em nossa experiência, não é necessariamente mais rápido do que usar o Jest.

92. .NET 7 Native AOT

Avalie

O .NET 7 Native AOT é um grande passo em uma longa série de abordagens para implantação de aplicativos .NET de forma nativa. Ele elimina completamente o IL (Intermediate Language) e o JIT (Just-In-Time) no runtime. Introduzida no .NET 7, essa melhoria é particularmente significativa para a execução de aplicativos .NET em funções sem servidor. Essa nova opção de implantação elimina o problema de inicialização a frio, que tem sido persistente para o .NET em plataformas sem servidor como a AWS Lambda e a Azure Functions. Com o Native AOT, você pode gerar um binário implantável menor do que com os métodos anteriores, resultando em tempos de inicialização a frio mais rápidos. A AWS adotou oficialmente o Native AOT, suportando-o com suas ferramentas Amazon Lambda. Essa nova opção de implantação equipara o .NET 7 ao TypeScript/JavaScript em termos de tempo de inicialização a frio, tornando-o uma opção viável para organizações com uma infraestrutura amplamente orientada para .NET.

93. .NET MAUI

Avalie

.NET MAUI é um nova framework multiplataforma para a criação de aplicativos móveis e desktop nativos com C# e XAML. Permite o desenvolvimento de aplicativos que podem ser executados no Android, no iOS, no macOS e no Windows, a partir de uma única base de código compartilhada. No entanto, como é uma nova tecnologia, o ecossistema em torno do MAUI não é tão desenvolvido quanto o do React Native ou de qualquer outro sistema multiplataforma, e suporta apenas C#. Além disso, o MAUI pode enfrentar os desafios encontrados por organizações que usaram Xamarin no passado, incluindo ferramentas multiplataforma deficitárias, problemas de integração em plataformas móveis, disponibilidade de pessoas desenvolvedoras e um ecossistema imaturo.

Embora a Microsoft tenha anunciado seu compromisso com o MAUI como um framework de código aberto com ênfase no desenvolvimento para plataformas móveis, seu sucesso ainda precisa ser comprovado. Se você já estiver usando o Xamarin, convém avaliar o MAUI como uma possível atualização; no entanto, se C# ou Xamarin ainda não fizerem parte do seu conjunto de ferramentas, convém abordar o MAUI com cautela até que a tecnologia seja adotada mais amplamente e aprovada pelo mercado.

94. dbt-expectations

Avalie

O dbt-expectations é um pacote de extensão para o dbt, inspirado no Great Expectations. A qualidade dos dados é um princípio importante da governança de dados. Então, quando falamos de governança de dados automatizada, é importante instalar controles internos que apontem anomalias ou problemas de qualidade nos pipelines de dados. Da mesma forma que os testes de unidade em um pipeline de compilação, o dbt-expectations cria asserções durante a execução de um pipeline de dados. No mundo dbt, é possível rodar testes de qualidade de dados, ao estilo Great Expectations, em seu data warehouse de forma direta, de dentro do dbt. Nossas equipes têm explorado a ferramenta, e faz sentido destacá-la aqui.

95. Directus

Avalie

Usamos Directus como sistema de gerenciamento de conteúdo (CMS) headless. Apesar de termos várias opções entre produtos de CMS headless, precisávamos de uma solução auto-hospedada com fluxos abundantes de gestão de ativos digitais e de produção de conteúdo. Nessa avaliação consideramos o Directus bem adequado às nossas necessidades; gostamos muito do processamento de dados acionado por eventos e da automação via flows.

96. Ferrocene

Avalie

A linguagem Rust vem ganhando popularidade nos últimos anos por seus recursos de segurança, desempenho e concorrência. No entanto, faltam conjuntos de ferramentas Rust certificadas para aplicações em mercados críticos de segurança, como o automotivo. Essa lacuna está sendo endereçada pelo Ferrocene, um conjunto de ferramentas de compilação para Rust. O Ferrocene promete ser compatível com o padrão de segurança funcional ISO 26262 para os sistemas eletrônicos em veículos rodoviários; um esforço para qualificar a linguagem e o conjunto de ferramentas para uso nesses domínios já está em andamento. Estamos entusiasmadas com seu progresso, e a disponibilidade de tais ferramentas para conformidade de segurança certamente acelerará a adoção do Rust na indústria automotiva.

97. Flutter para dispositivos embarcados

Avalie

Flutter para dispositivos embarcados torna relativamente fácil criar e manter uma interface de usuário moderna, semelhante a dos aplicativos móveis, mas que atenda sistemas embarcados, tais como interface humano-máquina (IHM) em carros, geladeiras e outros aparelhos de consumo. Isso é possível com Flutter, agora com suporte a embedders personalizados, o que permite a portabilidade para diferentes plataformas. Os aplicativos são escritos na linguagem de programação Dart usando o SDK e o ecossistema do Flutter. Temos desenvolvido protótipos com ele - nossas pessoas desenvolvedoras adoram a experiência de desenvolvimento e as nossas clientes gostam da agilidade, velocidade e experiência de usuária moderna que ela traz.

98. Fugue

Avalie

Em se tratando de engenharia de dados, estamos vendo uma variedade desconcertante de ferramentas e tecnologias. Especialmente para engenheiras menos experientes, pode fazer sentido trabalhar com uma camada de abstração para acessar as ferramentas, focar na tarefa em mãos sem precisar aprender várias APIs específicas de determinadas tecnologias e ter a opção de alternar tecnologias subjacentes sem muito esforço. Fugue é essa camada de abstração. Fornecendo uma interface unificada para computação distribuída que possibilita a execução de código Python, pandas e SQL em Spark, na Dask, na Ray e no DuckDB com o mínimo de necessidade de reescrita. No entanto, se sua equipe já se decidiu sobre um conjunto de tecnologias, está familiarizada com as APIs e se aprofundou nos ajustes de seus sistemas de back-end, dada a nossa experiência, essa camada de abstração fornece menos valor.

99. Galacean Engine

Avalie

O Galacean Engine é um mecanismo interativo para web e dispositivos móveis, projetado para fornecer uma maneira perfeita de renderizar arquitetura e animação baseadas em componentes, de uma maneira amigável para os dispositivos móveis. Com foco na renderização leve e de alto desempenho, tornou-se uma escolha cada vez mais popular entre pessoas desenvolvedoras que criam jogos envolventes para dispositivos móveis. É um mecanismo baseado em TypeScript que as pessoas desenvolvedoras relatam superar as alternativas.

100. LangChain

Avalie

A LangChain é um framework para desenvolvimento de aplicação usando Grandes Modelos de Linguagem (LLMs). Esses modelos desencadearam uma corrida pela incorporação de IA generativa em inúmeros casos de uso. Entretanto, o uso desses LLMs isoladamente pode não ser o suficiente — é preciso combiná-los com seus outros ativos diferenciados para chegar a um produto impactante. LangChain ocupa esse nicho, com alguns recursos incluindo gerenciamento de prompt, encadeamento, geração de dados aumentados (data augmentation) e um rico conjunto de agentes para determinar as ações a serem executadas e em qual ordem. Esperamos que mais ferramentas e frameworks evoluam com os LLMs, e recomendamos avaliar LangChain.

101. mljar-supervised

Avalie

O mljar-supervised é um pacote de AutoML em Python que auxilia na compreensão e explicação de dados tabulares. Nossas equipes de ciência de dados estão entusiasmadas com isso e o utilizam para automatizar a análise exploratória de dados. Ele abstrai a maneira comum de pré-processar os dados, construir os modelos de aprendizado de máquina (ML) e realizar o ajuste de hiperparâmetros para encontrar o melhor modelo. Explicabilidade e transparência são princípios importantes, e é aí que o mljar-supervised se destaca. Ele permite ver exatamente como a pipeline de ML é construída, com um relatório formatado e detalhado para cada modelo. É definitivamente um pacote AutoML interessante, que vale a pena avaliar para suas necessidades de ML.

102. nanoGPT

Avalie

nanoGPT é um framework para treinar e afinar transformadores generativos pré-treinados (GPT) de tamanho médio. O autor, Andrej Karpathy, se baseia nos artigos Attention is All You Need e OpenAI's GPT-3 para criar um GPT do zero usando o PyTorch. Com todo o barulho em torno da IA generativa, queríamos destacar o nanoGPT por sua simplicidade e sua preocupação em articular sem ambiguidade os blocos constitutivos da arquitetura GPT.

103. pandera

Avalie

Em edições anteriores do Radar, apresentamos plataformas de validação e teste de dados como a Great Expectations, que podem ser usadas para validar suposições e testar a qualidade de dados de entrada empregados em treinamento ou classificação. Às vezes, porém, tudo o que você precisa é de uma biblioteca de código simples para implementar testes e verificações de qualidade diretamente nos pipelines. A pandera é uma biblioteca Python para testar e validar dados em uma ampla variedade de tipos de frames, como pandas, Dask ou PySpark. pandera pode implementar afirmações simples sobre campos ou testes de hipóteses com base em modelos estatísticos. A grande variedade de bibliotecas de frames suportadas significa que os testes podem ser escritos uma vez e depois aplicados a uma variedade de formatos de dados subjacentes. pandera também pode ser usada para gerar dados sintéticos para testar modelos de ML.

104. Qwik

Avalie

Um dos desafios de criar uma experiência rica e interativa baseada em navegador é minimizar o tempo desde a primeira solicitação até a total interatividade da usuária. Ao inicializar, o aplicativo pode precisar baixar grandes quantidades de JavaScript para o navegador ou executar um longo processo para restaurar o estado do aplicativo no servidor. Qwik é um novo framework de frontend que serializa o estado do aplicativo, para que possa ser renderizado no servidor sem necessidade de "reidratação" ou repetição da lógica do aplicativo. Isso é obtido através da possibilidade de retomada (resumability), que envolve pausar a execução no servidor para retomá-la no cliente. E na mesma linha de outras frameworks de frontend mais recentes, como Astro ou a Svelte, Qwik também acelera o tempo de carregamento inicial da página, minimizando a quantidade de JavaScript ao carregar. No caso do Qwik, o download inicial do aplicativo é principalmente HTML, com a maioria do JavaScript carregado dinamicamente sob demanda de um cache local, se possível.

105. SolidJS

Avalie

O SolidJS é uma biblioteca JavaScript declarativa para criar interfaces de usuária. No ano passado, vimos um aumento na visibilidade e popularidade do SolidJS entre as pessoas desenvolvedoras, principalmente entre aquelas interessadas em criar interações de usuária mais ricas. Essa biblioteca compila seus modelos para nós reais do DOM (em vez de usar o vDOM) e os atualiza com reações precisas, que reduzem as atualizações desnecessárias do DOM e resultam em um desempenho mais rápido e uma melhor experiência da usuária. Ele tem uma API simples e um ótimo suporte a TypeScript, o que pode ajudar a detectar erros durante o desenvolvimento. Outro benefício do SolidJS é o tamanho pequeno do pacote, que é ideal para desenvolver aplicativos web rápidos e leves, além de beneficiar a abordagem "dispositivo móvel primeiro". O SolidJS é um framework relativamente novo, assim não possui uma comunidade ou ecossistema tão grande quanto outros frameworks. No entanto, a julgar pelo número crescente de bibliotecas e ferramentas úteis, parece estar crescendo em popularidade. Seu sistema de atualização reativa, modelo de componente funcional e sistema de modelagem tornam o SolidJS uma escolha atraente para avaliar e estamos vendo o interesse de várias equipes e comunidades nele.

106. Turborepo

Avalie

Um dos temas que parece despertar um interesse perene em nossas discussões é a questão dos monorepos. Alguns locais os adotaram para toda a organização, enquanto outros aplicaram o conceito em certos aplicativos restritos, como aplicativos móveis ou de desenvolvimento combinado de UI/BFF. Independente de quando ou onde é apropriado usar os monorepos, o setor parece estar revisitando ferramentas que podem gerenciar com eficácia grandes bases de código e prepará-las com eficiência em unidades implantáveis. Turborepo é uma ferramenta relativamente nova nesta categoria, que oferece uma alternativa ao Nx ou ao Lerna para grandes bases de código JavaScript ou TypeScript. Um dos desafios com grandes repositórios é executar compilações com rapidez suficiente para não interromper o fluxo da pessoa desenvolvedora ou reduzir a eficiência. Turborepo é escrito em Rust, o que lhe dá grande desempenho; ele também compila de forma incremental e armazena em cache as etapas intermediárias, para acelerar ainda mais as coisas. No entanto, requer mudanças no fluxo de trabalho da pessoa desenvolvedora que levam tempo para aprender e provavelmente é mais adequado para grandes bases de código com várias compilações independentes, onde uma abordagem diferente é necessária. Descobrimos que a documentação é escassa, levando algumas equipes a, por ora, se aterem a ferramentas mais bem estabelecidas. No entanto, vale a pena avaliar e ver se o Turborepo e seu novo companheiro, o Turbopack (atualmente em versão beta), continuam a evoluir.

107. WebXR Device API

Avalie

Quando começamos a trabalhar na API experimental WebVR, ficou claro que faria mais sentido ter uma API combinada para VR (Realidade Virtual) e AR (Realidade Aumentada). Em vez de mudar drasticamente a API WebVR, uma nova especificação foi criada: WebXR. Fundamentalmente, é a WebXR Device API que fornece recursos cruciais para escrever aplicações VR e AR para navegadores. A API é extensiva e, no momento em que escrevemos, não é completamente suportada por todos os navegadores. Nossas equipes usaram a WebXR em várias ocasiões, e vimos os benefícios descritos pelo Immersive Web Working Group. Para protótipos, nós gostamos particularmente da experiência estar disponível imediatamente em um navegador web. A equipe de desenvolvimento não precisa passar pelo processo de aprovação em uma loja de apps, e os usuários podem brincar com a experiência sem precisar instalar uma aplicação. Dado o status da API e o fato dela estar oculta sob feature toggle em alguns navegadores, não a vimos sendo usada além de provas de conceito e protótipos.

Quer continuar se atualizando com artigos e informações relacionadas ao Radar?

Siga nossos perfis nas redes sociais e inscreva-se gratuitamente para se tornar assinante.

Assine

A Thoughtworks é uma consultoria global de tecnologia que integra estratégia, design e engenharia de software para alavancar a inovação digital. Somos mais de 12,5 mil pessoas distribuídas entre 50 escritórios e em 18 países. Há mais de 25 anos, trabalhamos junto a nossas clientes para criar impacto extraordinário, usando a tecnologia como diferenciador para ajudá-las a resolver problemas de negócio complexos.

/thoughtworks