Proposta de Monitoramento e Observabilidade

Aplicativo de Chat em Flutter com Firestore e Login via SSO do Google

Equipe:

- Rhyan Britto
- Maria Jayara
- Érica Spadêto
- Adriano Pontes
- Lucas Benevides
- Marcelo Couto

1. Introdução

O aplicativo de chat foi desenvolvido com o intuito de oferecer comunicação ágil e segura aos usuários. Para isso, foram implementadas funcionalidades que possibilitam a visualização do email e nome do usuário, o logoff e a geração manual de um crash para teste do Crashlytics, garantindo que o monitoramento dos erros seja eficaz.

2. Requisitos

2.1. Requisitos Funcionais

- **Tela de Configurações:**
  Permite que o usuário consulte seu email e nome, realize o logoff e acione manualmente um crash para testes do Crashlytics.

- **Monitoramento de Erros e Crashes:**

    - Captura e registro de falhas e exceções por meio do Firebase Crashlytics.
    - Geração de relatórios detalhados com stack trace e notificação automática por e-mail à equipe, sempre que um novo crash ocorrer ou houver aumento significativo na taxa de erros.
- **Coleta de Eventos de Uso (Analytics):**
  Registro dos eventos:

    - enviar_mensagem
    - login
    - login_falha
    - login_sucesso
    - sair_aplicativo
    - tentativa_login
    - testar_crashlytics
- **Observabilidade do Firestore:**
  Monitoramento concentrado nos dados relativos aos alertas, com visualização centralizada via Grafana.

- **Geração de Alertas Automáticos:**
  Configuração dos seguintes alertas:

    - **Potencial Spam:** Acionado quando um usuário envia mais de 100 mensagens em menos de 1 minuto.
    - **Alto Volume de Mensagens:** Acionado quando o total de mensagens enviadas ultrapassa 500 em menos de 1 minuto.
    - **Palavrão Detectado:** Acionado quando um palavrão é identificado em alguma mensagem.

2.2. Requisitos Não Funcionais

- Garantir a segurança e integridade dos dados.
- Permitir a rápida identificação de comportamentos suspeitos.
- Manter uma implementação simples e focada nas métricas e alertas essenciais.

3. Escopo

Em Escopo:

- **Erros e Crashes:**
  Monitoramento via Firebase Crashlytics, com envio automático de notificações à equipe.

- **Eventos de Uso:**
  Coleta e análise dos eventos definidos (Analytics).

- **Alertas e Observabilidade:**
  Monitoramento e exibição de dados relacionados aos alertas configurados (potencial spam, alto volume de mensagens e palavrão detectado) através do Grafana, que também apresenta um indicador para usuários inativos.

Fora de Escopo:

- Monitoramento avançado de desempenho.
- Desenvolvimento de backend próprio (a solução utiliza os serviços do Firebase).

4. Objetivos de Observabilidade

- **Detecção e Correção de Erros:**
  Receber alertas e relatórios em tempo real para permitir a correção imediata dos problemas.

- **Monitoramento de Uso e Engajamento:**
  Analisar os eventos de uso para compreender o comportamento dos usuários.

- **Identificação de Comportamentos Suspeitos:**
  Detectar situações de spam, envio excessivo de mensagens e o uso de termos inapropriados.

- **Acompanhamento dos Alertas:**
  Monitorar os dados referentes aos alertas para possibilitar ações proativas pela equipe de suporte.

5. Estratégia de Monitoramento

A estratégia de monitoramento integra os seguintes processos:

- **Logs e Relatórios de Erros:**
  Utilização do Firebase Crashlytics para capturar e reportar crashes e exceções, com a possibilidade de acionar manualmente um crash via tela de configurações.

- **Coleta de Eventos:**
  Registro dos eventos de uso por meio do Firebase Analytics, permitindo a análise do comportamento dos usuários.

- **Observabilidade do Firestore:**
  Foco na coleta dos dados relativos aos alertas, com esses dados sendo exibidos em dashboards configurados no Grafana.

- **Visualização e Indicadores:**
  O Grafana exibe dashboards centralizados com os dados dos alertas, incluindo um indicador informativo para usuários inativos, que permite identificar a falta de acesso sem disparar um alerta automático.

6. Ferramentas e Componentes Utilizados

- **Firebase Crashlytics:**
  Captura e monitoramento de erros e crashes no aplicativo.

- **Firebase Analytics:**
  Coleta dos eventos de uso, possibilitando análises detalhadas do comportamento dos usuários.

- **Firestore:**
  Banco de dados utilizado para armazenar os dados do aplicativo, com monitoramento focado nos alertas.

- **Grafana:**
  Plataforma de dashboards que exibe os dados de alertas e os indicadores do Firestore, facilitando a visualização e a tomada de decisão.

7. Métricas e Indicadores-Chave

- **Potencial Spam:**
  Número de ocorrências em que um usuário envia mais de 100 mensagens em menos de 1 minuto.

- **Alto Volume de Mensagens:**
  Número de ocorrências em que o total de mensagens enviadas em menos de 1 minuto ultrapassa 500.

- **Palavrão Detectado:**
  Número de alertas gerados pela identificação de palavrões em mensagens.

- **Usuários Inativos:**
  Indicador exibido no Grafana que monitora os usuários que não acessam o aplicativo por um período determinado (sem disparo automático de alerta).

8. Dashboards e Relatórios

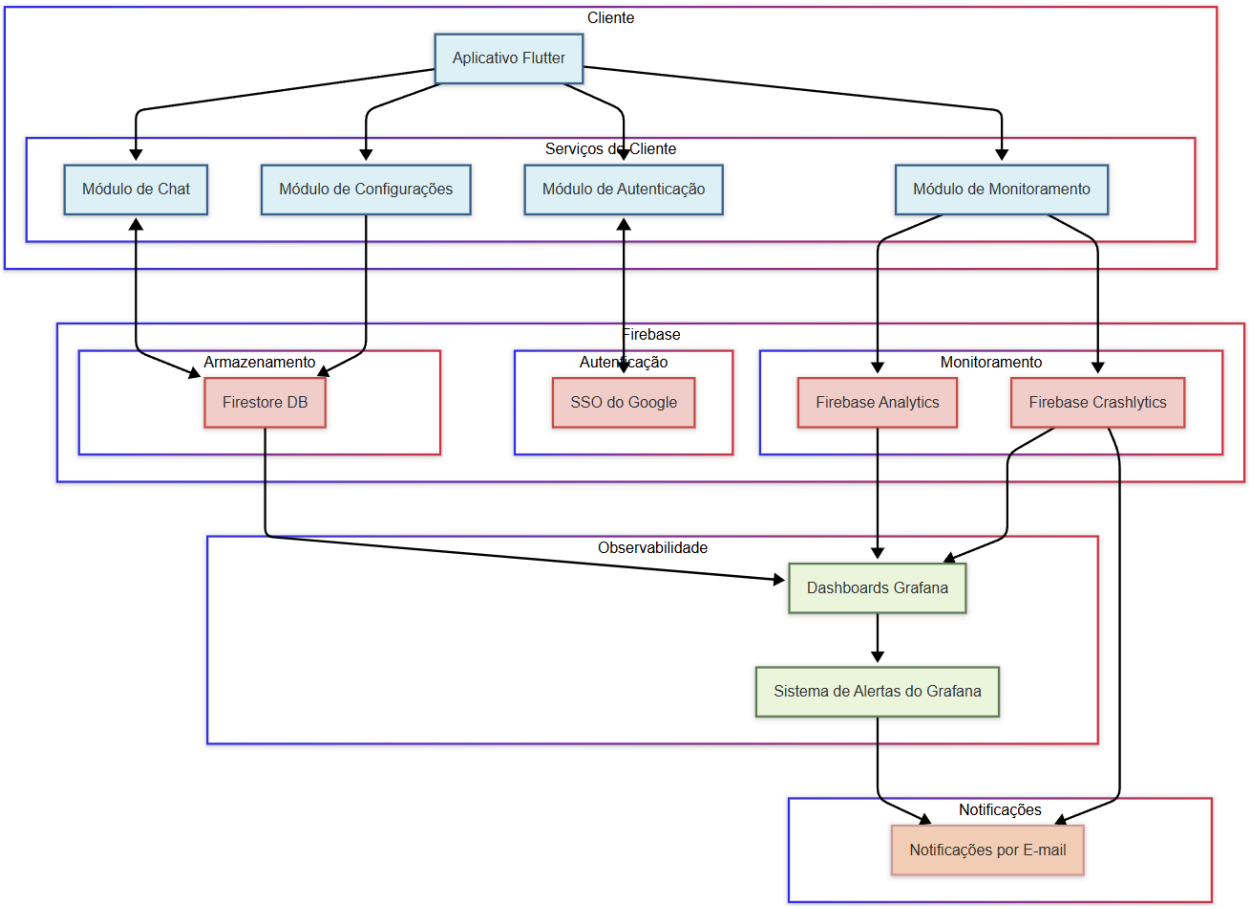
- **Grafana:**
  Configurado para exibir dashboards centralizados com os dados dos alertas provenientes do Firestore. Os dashboards incluem gráficos para:

    - Monitoramento de potenciais spams e alto volume de mensagens.

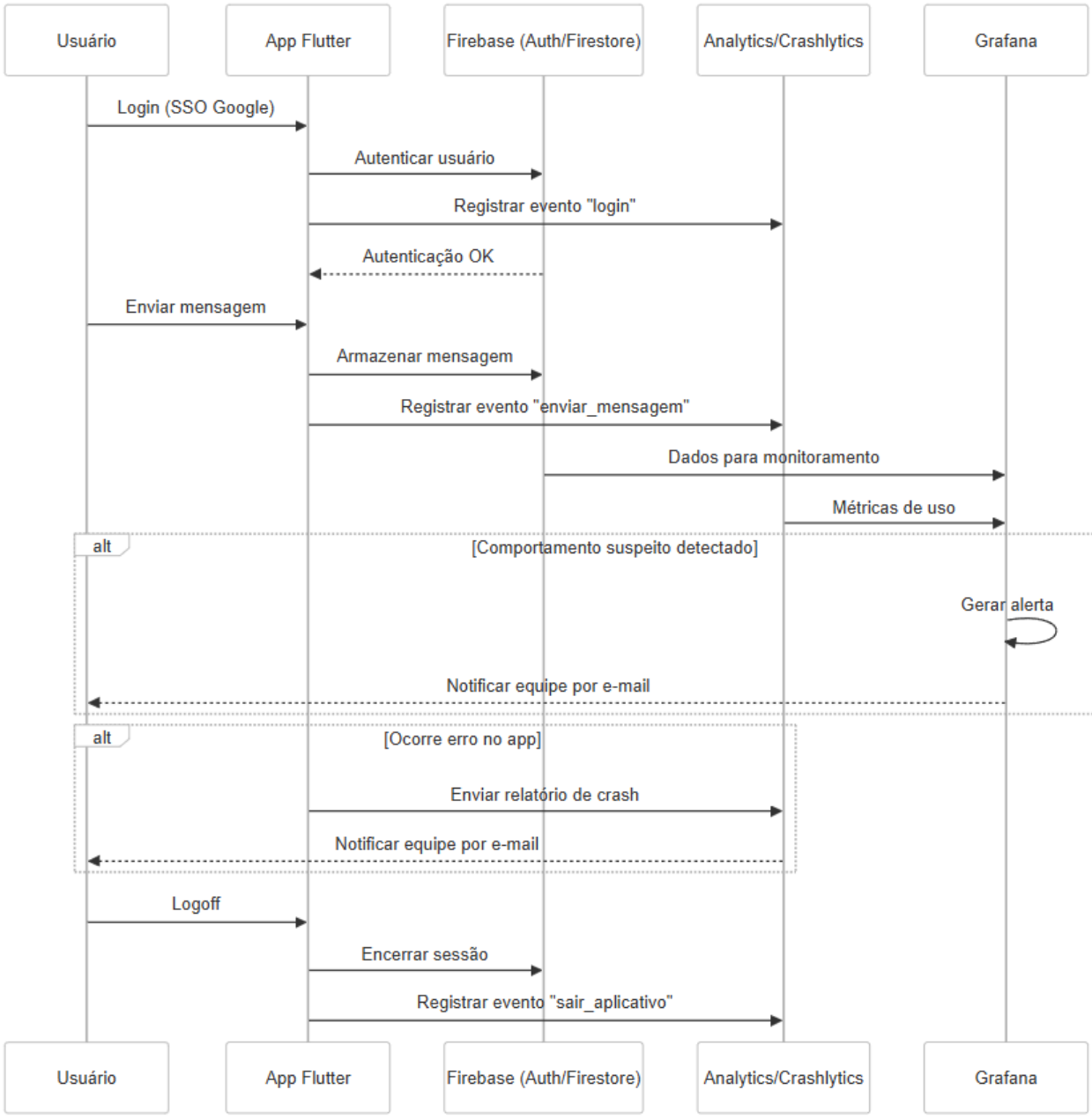
    - Indicadores de ocorrência de palavrões nas mensagens.
    - Indicador informativo para usuários inativos.
- **Firebase Consoles:**
  Utilizados para análises detalhadas dos eventos e dos crashes através dos consoles do Crashlytics e Analytics.

9. Diagramas

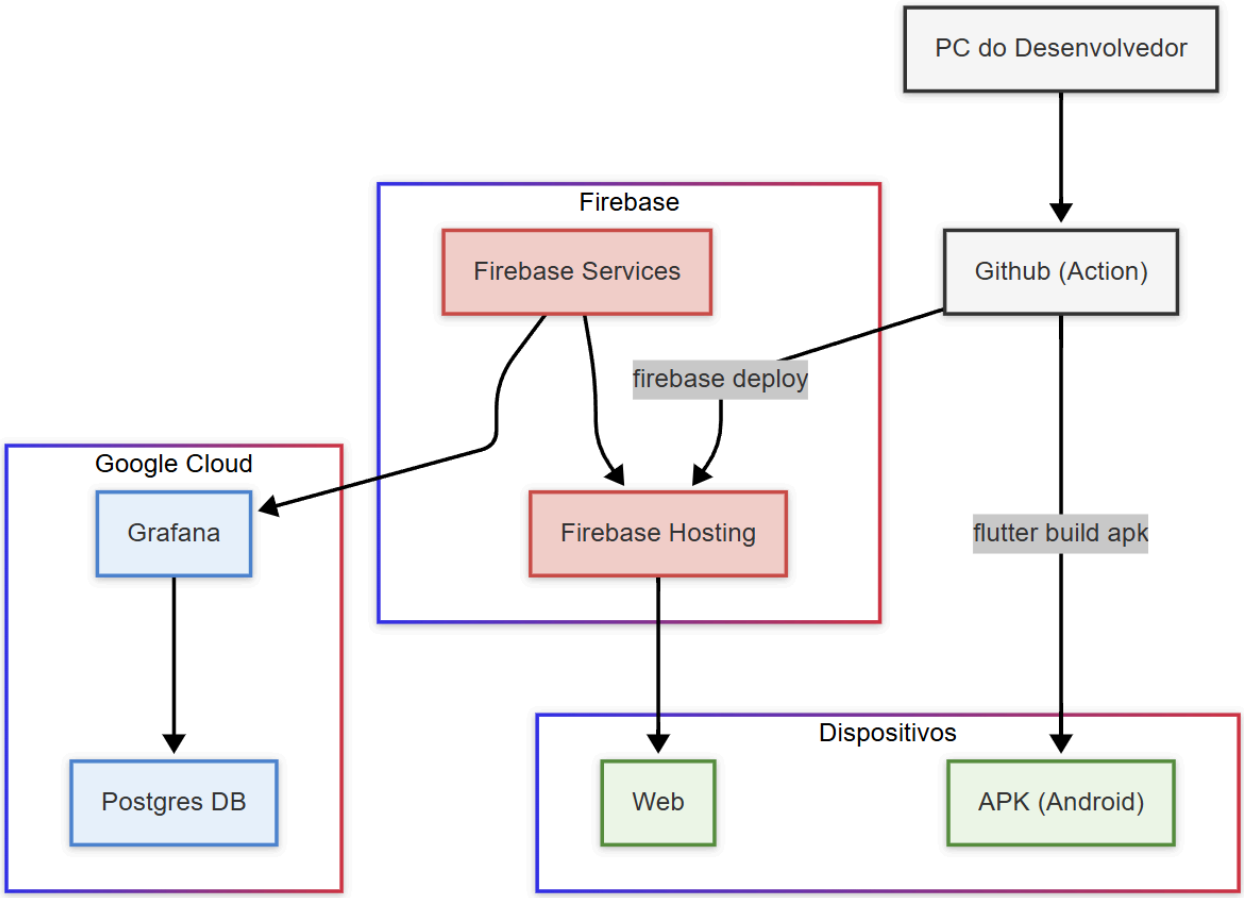
- Diagrama de componentes



- Diagrama de sequência



- Diagrama de deploy



10. Alertas e Notificações

- **Crashlytics:**
  Notifica automaticamente os membros da equipe via e-mail sempre que um novo crash é registrado ou há aumento significativo na taxa de erros.

- **Alertas Configurados (via Grafana):**

  - **Potencial Spam:** Alerta quando um usuário envia mais de 100 mensagens em menos de 1 minuto.
  - **Alto Volume de Mensagens:** Alerta quando o total de mensagens enviadas ultrapassa 500 em menos de 1 minuto.
  - **Palavrão Detectado:** Alerta quando um palavrão é identificado em alguma mensagem.

Observação: Os alertas são enviados diretamente para os e-mails dos membros do time, substituindo o endereço anteriormente utilizado.

11. Cronograma

1. **Desenvolvimento do Aplicativo:**
   Implementação das funcionalidades básicas, incluindo a tela de configurações com visualização do email, nome, logoff e acionamento manual de crash.

2. **Configuração do Monitoramento:**
   Integração do Firebase Crashlytics e Firebase Analytics, com registro dos eventos e testes de geração de crashes.

3. **Integração do Firestore com o Grafana:**
   Configuração dos dashboards e dos mecanismos de alerta focados nos dados dos alertas.

4. **Ajuste dos Parâmetros de Alerta:**
   Testes e definição dos limiares para os alertas de potencial spam, alto volume de mensagens e palavrão detectado.

5. **Validação e Implantação:**
   Revisão final e validação do sistema de monitoramento, garantindo que os dados estejam sendo capturados e exibidos conforme o esperado.

12. Conclusão

Este plano de observabilidade estabelece uma estratégia completa para o monitoramento do aplicativo de chat em Flutter. A integração entre Firebase Crashlytics, Firebase Analytics, Firestore e Grafana proporciona uma visão abrangente dos erros, dos eventos de uso e dos comportamentos suspeitos. Com alertas configurados para identificar potenciais spams, alto volume de mensagens e palavrões, além de um indicador para usuários inativos, a equipe de suporte pode agir de maneira rápida e assertiva, assegurando a estabilidade e a segurança do sistema.