Processo básico de Desenvolvimento de Software

Esse documento é a referência a processos básicos para o desenvolvimento da aplicação “My Health”, contém a descrição de padrões, ferramentas e boas práticas que serão utilizadas durante todo o ciclo de vida do projeto.

1. Padrões para o projeto

O aplicativo e serviços derivados como os microsserviços e o banco de dados seguem alguns padrões de desenvolvimento. Os padrões arquiteturais estão descritos no documento “Arquitetura de Software”, esta seção destina-se apenas aos padrões que envolvem o desenvolvimento e manutenção do software.

* 1. Linguagens

A aplicação móvel utiliza a linguagem de programação Kotlin, esse é um requisito importante devido a dependência com o kit de ferramentas JetPack Compose, a linguagem é interoperável com a linguagem Java, podendo inclusive utilizar suas bibliotecas padrão. Os microsserviços do back-end utilizam a linguagem Java através do framework Spring Boot, apesar de ser possível reutilizar a linguagem Kotlin, prefere-se priorizar Java para melhor distinguir os domínios da aplicação, como as linguagens interoperam entre si, componentes podem ser facilmente reutilizáveis. Por fim, a comunicação do banco de dados deve ser feita em SQL. Para maiores informações consultar a seção “3 Boas Práticas” desse documento.

* 1. Banco de Dados

O projeto vai utilizar dois tipos diferentes de bancos de dados relacionais, para a aplicação móvel, o SQLite será utilizado, pois fará o armazenamento local de apenas os dados do usuário ativo na aplicação. Nos microsserviços, o banco de dados será o PostGreSQL, devido ao maior volume de dados e aos acessos simultâneos entre os microsserviços. Para maiores informações consultar a seção “3 Boas Práticas” desse documento.

1. Ferramentas do projeto

O projeto utiliza diversas ferramentas para o desenvolvimento da aplicação e dos microsserviços, abaixo uma breve descrição dessas ferramentas:

* 1. Git

O sistema de gerenciamento de versões do projeto é feito pelo software Git, o repositório central é localizado em <https://github.com/d4rkwav3/projetoa3.2023.1/> , e cada desenvolvedor deve criar seu próprio ramo (branch) ao realizar qualquer tipo de alteração no projeto. Apenas após as alterações terem sido testadas, revistas e aprovadas pela equipe o ramo poderá ser fundido ao ramo main (principal).

* 1. JetPack Compose

Todo o desenvolvimento da interface de usuário para a aplicação móvel utilizar o kit de ferramentas JetPack Compose, disponível em repositórios Maven/Gradle, o antigo padrão XML não deve ser utilizado em nenhuma hipótese. Para maiores informações consultar o documento “Arquitetura de Software”.

* 1. Android Studio

Para o desenvolvimento da aplicação móvel, o desenvolvedor deve utilizar a IDE (integrated development environment) Android Studio, essa IDE possui diversas facilidades que agilizam o desenvolvimento da aplicação, entre elas o download e verificação automática de dependências, automatização de testes, padrões de projeto, checagem de qualidade de código, cálculo de cobertura dos testes, ambiente de emulação do sistema Android, pré-visualização de interfaces gráficas, entre outras.

* 1. Spring Boot

Os microsserviços de suporte ao aplicativo móvel devem ser desenvolvidos utilizando a framework Spring Boot, utilizando as dependências “Spring Web” para as requisições HTTP e “Spring JPA” para transações com o banco de dados.

1. Boas práticas

A seguir uma série de boas práticas que devem ser utilizadas extensivamente e sem exceções durante todo o ciclo de desenvolvimento da aplicação, separadas entre os ambientes de aplicação móvel, microsserviços e banco de dados.

* 1. Aplicação móvel
     1. Convenções da linguagem Kotlin

As conversões descritas na documentação da linguagem Kotlin são o guia para o projeto da aplicação, pode ser acessada no link <https://kotlinlang.org/docs/coding-conventions.html>. É importante exaltar algumas dessas convenções nesse documento. A escrita do nome de variáveis deve ser feita em PascalCase, porém as classes devem começar com a primeira letra maiúscula, diferente das variáveis internas que começam com uma letra minúscula. Os nomes devem ser os mais descritivos e significativos que for possível, um uso mínimo de comentários deve ser feito a fim de explicar métodos preferencialmente acima da declaração do mesmo escrito o mais breve que for possível. A utilização da língua inglesa é obrigatória dos arquivos do projeto, com exceção de comentários que devem ser feitos em língua portuguesa do Brasil. Apesar da interoperabilidade entre kotlin e java, todos os arquivos do pacote do projeto móvel devem utilizar a sintaxe kotlin, ao se utilizar arquivos do back-end, eles devem ser reestruturados às convenções kotlin.

* + 1. Organização dos arquivos.

Os arquivos do projeto devem residir em diretórios que abriguem arquivos de funções semelhantes, por exemplo, os arquivos de classe de UI deve estar na pasta/pacote “UI”, as classes de modelo na pasta “Model”. Toda classe de função deve ter uma pasta própria.

* + 1. API Android Mínima

O uso da API mínima do android deve obrigatoriamente cobrir um mínimo de 90% dos dispositivos móveis, é proibido o uso de API mais recentes que diminua os aparelhos compatíveis para abaixo dessa porcentagem.

* 1. Microsserviços
     1. Convenções da linguagem Java

As conversões descritas na documentação da linguagem Java são o guia para o projeto da aplicação, pode ser acessada no link <https://www.oracle.com/technetwork/java/codeconventions-150003.pdf> . É importante exaltar algumas dessas convenções nesse documento. A escrita do nome de variáveis deve ser feita em PascalCase, porém as classes devem começar com a primeira letra maiúscula, diferente das variáveis internas que começam com uma letra minúscula. Os nomes devem ser os mais descritivos e significativos que for possível, um uso mínimo de comentários deve ser feito a fim de explicar métodos preferencialmente acima da declaração do mesmo escrito o mais breve que for possível. A utilização da língua inglesa é obrigatória dos arquivos do projeto, com exceção de comentários que devem ser feitos em língua portuguesa do Brasil. Apesar da interoperabilidade entre kotlin e java, todos os arquivos do pacote do projeto de back-end devem utilizar a sintaxe Java, ao se utilizar arquivos da aplicação móvel, eles devem ser reestruturados às convenções Java.

* + 1. Organização dos arquivos

Os arquivos do projeto devem residir em diretórios que abriguem arquivos de funções semelhantes, por exemplo, os arquivos de classe de UI deve estar na pasta/pacote “UI”, as classes de modelo na pasta “Model”. Toda classe de função deve ter uma pasta própria.

* + 1. Ambiente de execução

O ambiente de desenvolvimento é de preferência do desenvolvedor, porém o ambiente de execução dos microsserviços é obrigatoriamente o GNU/Linux Debian, em sua versão LTS (Long Term Support) mais recente. Toda atualização dos microsserviços deve obrigatoriamente ser testadas nesse ambiente, pois é o ambiente de produção.

* 1. Banco de Dados
     1. Banco de dados da aplicação móvel

A aplicação móvel pode criar e manipular um banco de dados SQLite local, todas as transações devem ser compatíveis com esse banco, o uso de dialetos específicos deve ser evitado, com o intuito de não gerar retrabalho (ver seção 3.3.3).

* + 1. Banco de dados geral

O banco de dados dos microsserviços PostGreSQL, todas as transações devem ser compatíveis com esse banco, o uso de dialetos específicos deve ser evitado, com o intuito de não gerar retrabalho (ver seção 3.3.3).

3.3.3 Dialetos SQL

O uso de dialetos SQL para os bancos de dados descritos acima deve ser evitado ao máximo a fim de permitir a reutilização de código e melhor manutenibilidade do projeto, exceções só devem ser consideradas casos alguma recurso equivalente não esteja disponível no padrão SQL ANSI e exija muitos recursos para uma implementação própria, cada caso deve ser revisto e aprovação antecipadamente pela liderança do projeto.