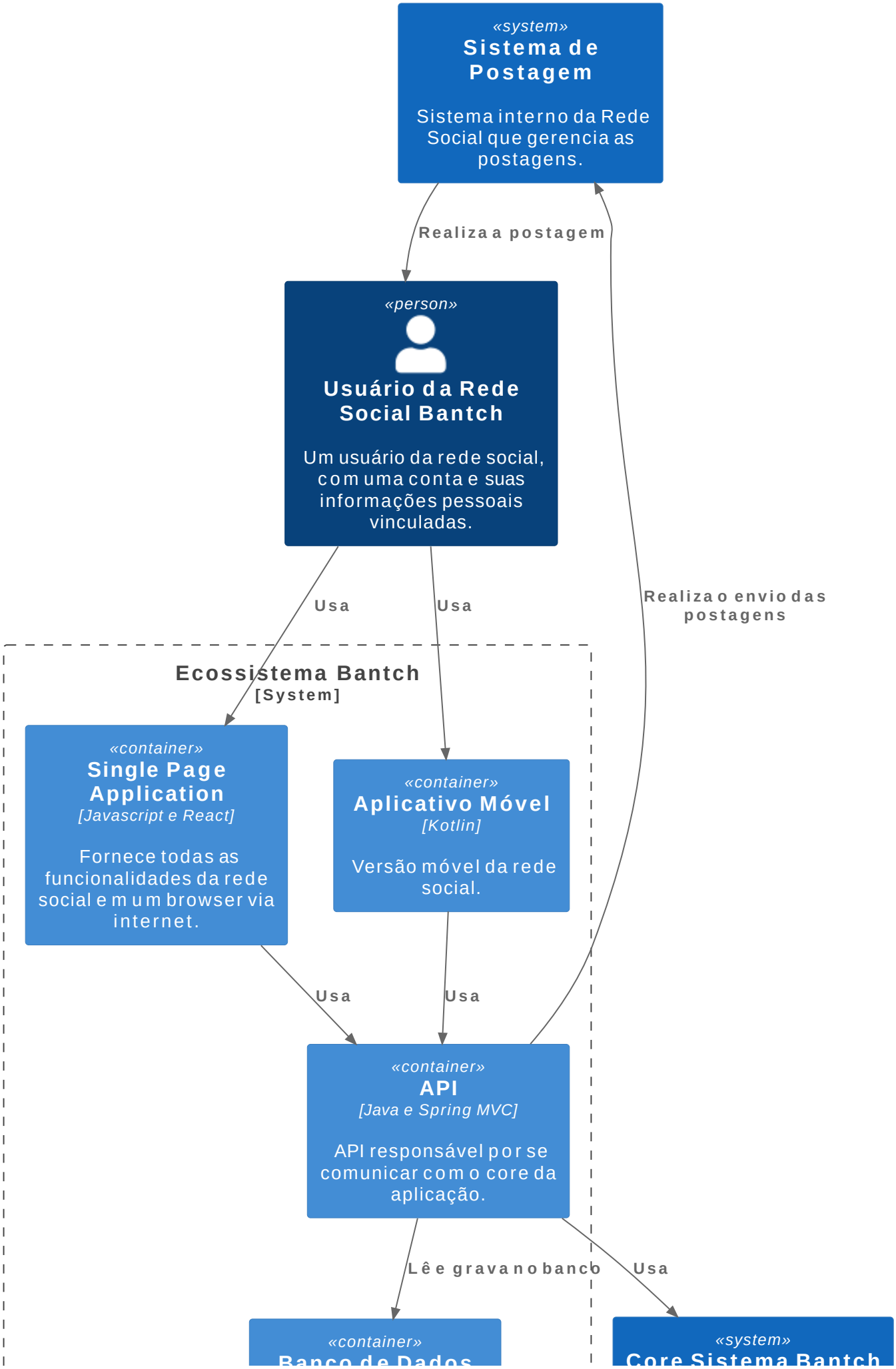
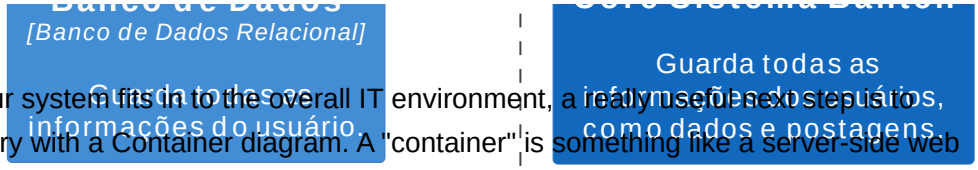
Bantch


«system»
Sistema de Postagem
Sistema interno da Rede Social que gerencia as postagens.
Realiza a postagem
«person»
Usuário da Rede Social Bantch
Um usuário da rede social, com uma conta e suas informações pessoais vinculadas.
Usa
Usa
Realiza o envio das postagens
Ecossistema Bantch
[System]
«container»
Single Page Application
[Javascript e React]
Fornece todas as funcionalidades da rede social em um browser via internet.
«container»
Aplicativo Móvel
[Kotlin]
Versão móvel da rede social.
Usa
Usa
«container»
API
[Java e Spring MVC]
API responsável por se comunicar com o core da aplicação.
Lê e grava no banco
Usa
«container»
Banco de Dados
«system»
Core Sistema Bantch

Level 2: Container diagram



Once you understand how your system fits in to the overall IT environment, a really useful next step is to zoom-in to the system boundary with a Container diagram. A "container" is something like a server-side web application, single-page application, desktop application, mobile app, database schema, file system, etc. Essentially, a container is a separately runnable/deployable unit (e.g. a separate process space) that executes code or stores data.

The Container diagram shows the high-level shape of the software architecture and how responsibilities are distributed across it. It also shows the major technology choices and how the containers communicate with one another. It's a simple, high-level technology focussed diagram that is useful for software developers and support/operations staff alike.

Scope: A single software system.

Primary elements: Containers within the software system in scope. Supporting elements: People and software systems directly connected to the containers.

Intended audience: Technical people inside and outside of the software development team; including software architects, developers and operations/support staff.

Notes: This diagram says nothing about deployment scenarios, clustering, replication, failover, etc.