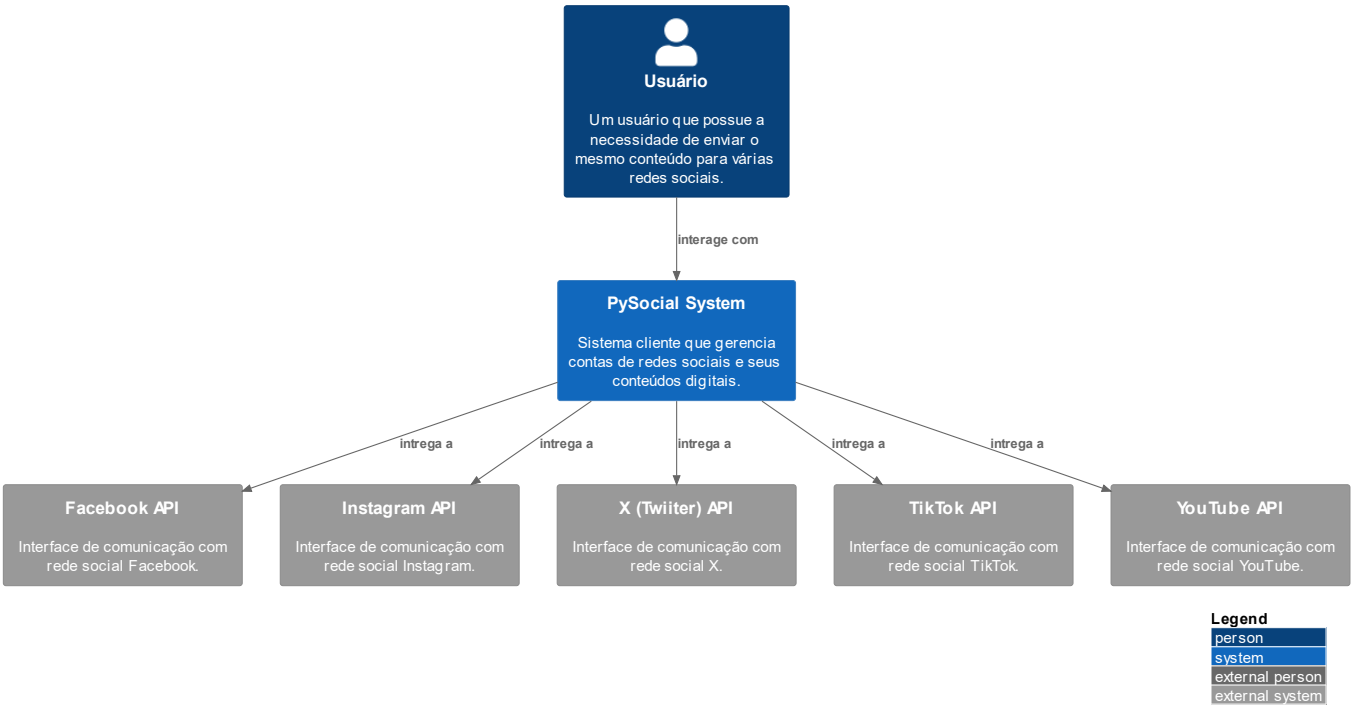
PySocial

PySocial Project



Level 1: Diagrama de contexto do PySocial

Scope: PySocial é um sistema que permite autenticar nas plataformas de redes sociais e gerenciar seus conteúdos digitais de forma cruzada.

Elementos primários:

- Usuário
- PySocial System

Elementos de suporte:

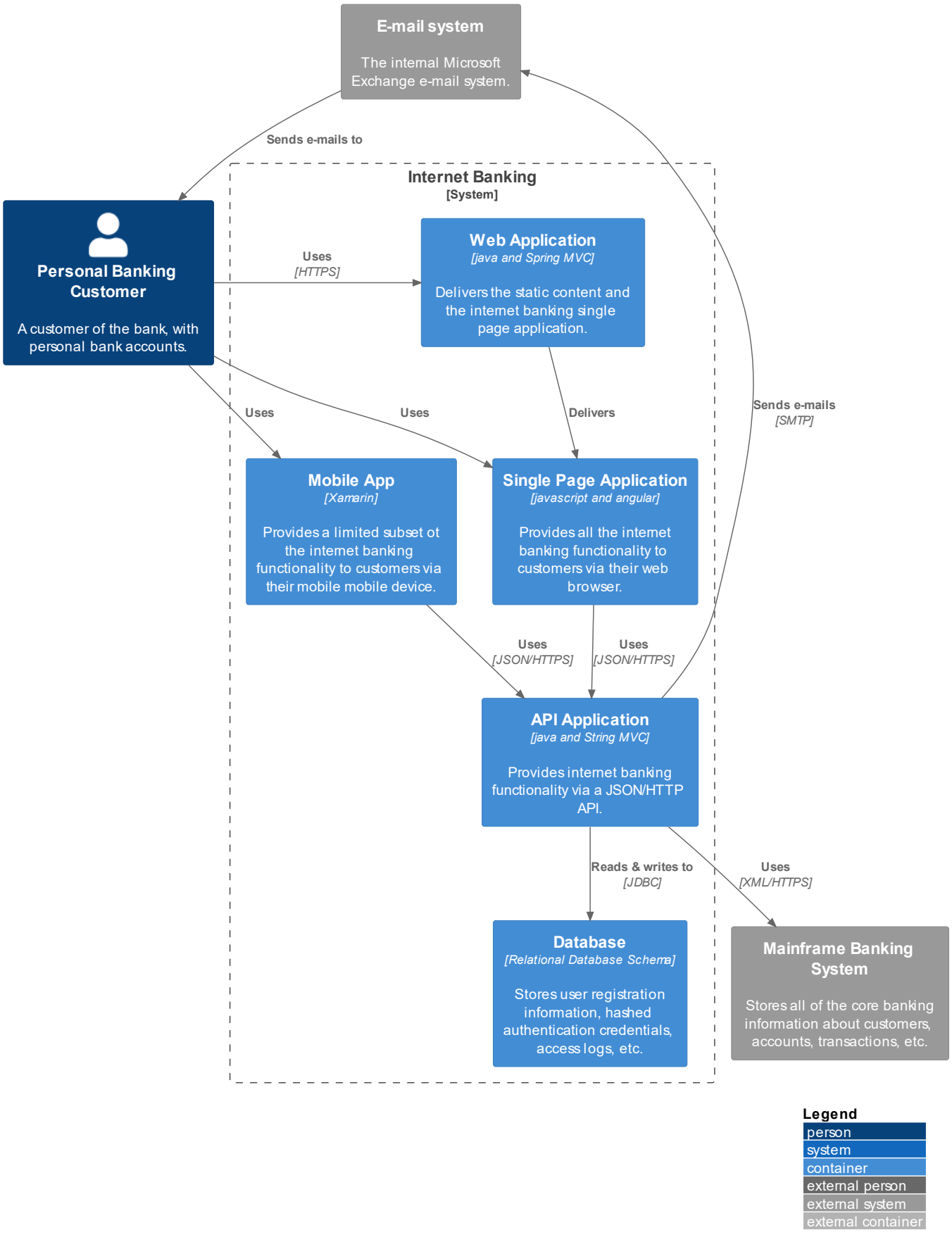
- Facebook API
- Instagram API
- X (Twiiter) API

- TikTok API
- YouTube API

Público-alvo: Todos, técnicos e não técnicos, dentro e fora da equipe de desenvolvimento de software.

1 Internet Banking System

```
\1 Internet Banking System
```



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.

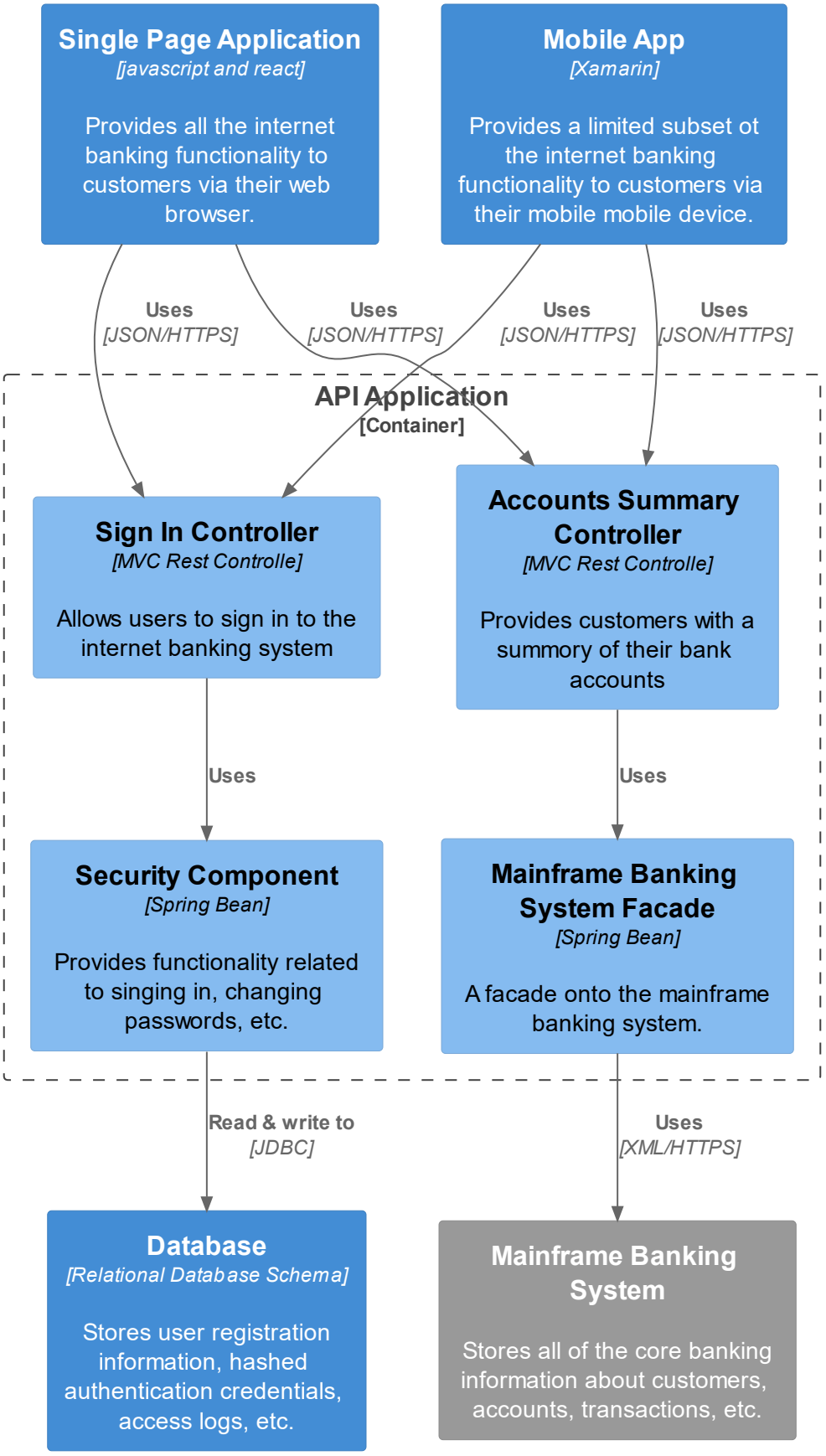
Primary elements: Containers within the software system in scope. Supporting elements: People and software systems directly connected to the containers.

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.

API Application

`\1 Internet Banking System\API Application`


Single Page Application
[javascript and react]
Provides all the internet banking functionality to customers via their web browser.
Mobile App
[Xamarin]
Provides a limited subset ot the internet banking functionality to customers via their mobile mobile device.
Uses
[JSON/HTTPS]
Uses
[JSON/HTTPS]
Uses
[JSON/HTTPS]
Uses
[JSON/HTTPS]
API Application
[Container]
Sign In Controller
[MVC Rest Controlle]
Allows users to sign in to the internet banking system
Accounts Summary Controller
[MVC Rest Controlle]
Provides customers with a summory of their bank accounts
Uses
Uses
Security Component
[Spring Bean]
Provides functionality related to singing in, changing passwords, etc.
Mainframe Banking System Facade
[Spring Bean]
A facade onto the mainframe banking system.
Read & write to
[JDBC]
Uses
[XML/HTTPS]
Database
[Relational Database Schema]
Stores user registration information, hashed authentication credentials, access logs, etc.
Mainframe Banking System
Stores all of the core banking information about customers, accounts, transactions, etc.
Legend
person
system
container
component
external person
external system
external container

external component

Level 3: Component diagram

Next you can zoom in and decompose each container further to identify the major structural building blocks and their interactions.

The Component diagram shows how a container is made up of a number of "components", what each of those components are, their responsibilities and the technology/implementation details.

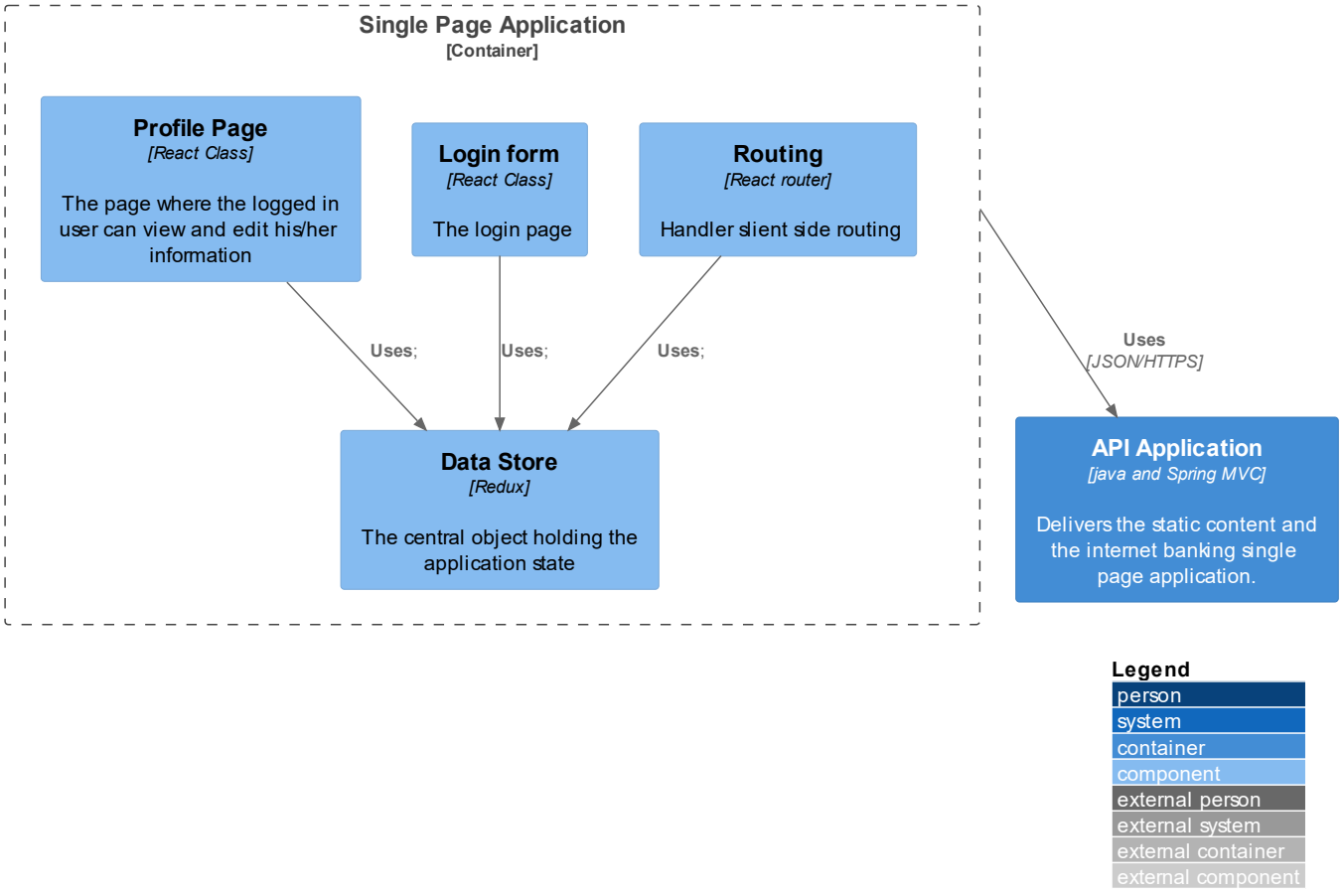
Scope: A single container.

Primary elements: Components within the container in scope. Supporting elements: Containers (within the software system in scope) plus people and software systems directly connected to the components.

Intended audience: Software architects and developers.

Single Page Application

`\1 Internet Banking System\Single Page Application`



Level 3: Component diagram

Next you can zoom in and decompose each container further to identify the major structural building blocks and their interactions.

The Component diagram shows how a container is made up of a number of "components", what each of those components are, their responsibilities and the technology/implementation details.

Scope: A single container.

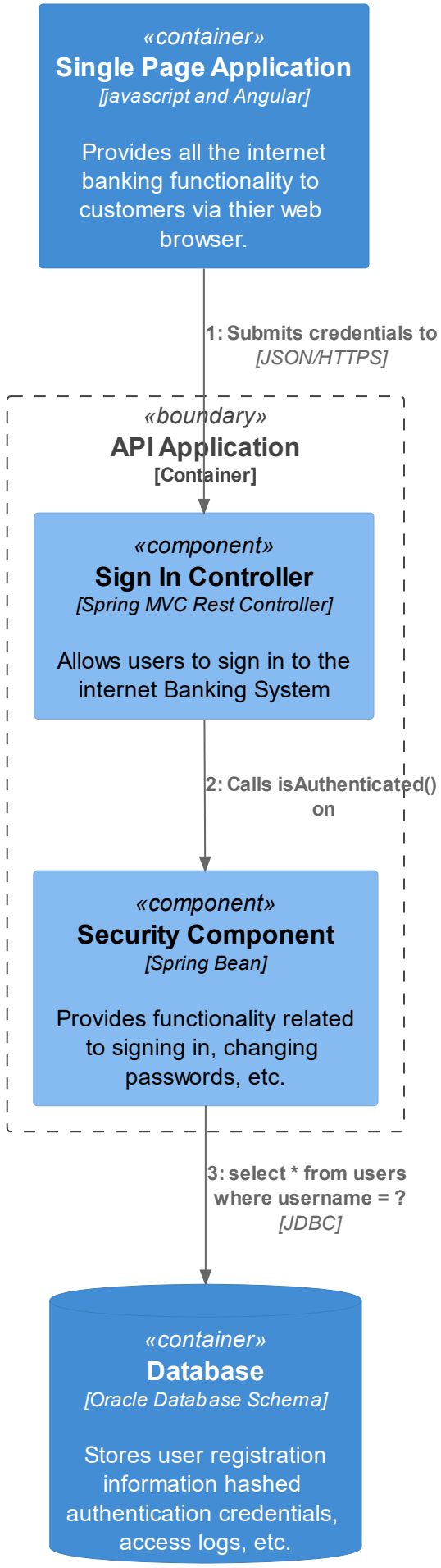
Primary elements: Components within the container in scope. Supporting elements: Containers (within the software system in scope) plus people and software systems directly connected to the components.

Intended audience: Software architects and developers.

> Example of included local image

Dynamic Diagram

```
\1 Internet Banking System\Single Page Application\Dynamic Diagram
```



Dynamic diagram

A simple dynamic diagram can be useful when you want to show how elements in a static model collaborate at runtime to implement a user story, use case, feature, etc. This dynamic diagram is based upon a UML communication diagram (previously known as a "UML collaboration diagram"). It is similar to a UML sequence diagram although it allows a free-form arrangement of diagram elements with numbered interactions to indicate ordering.

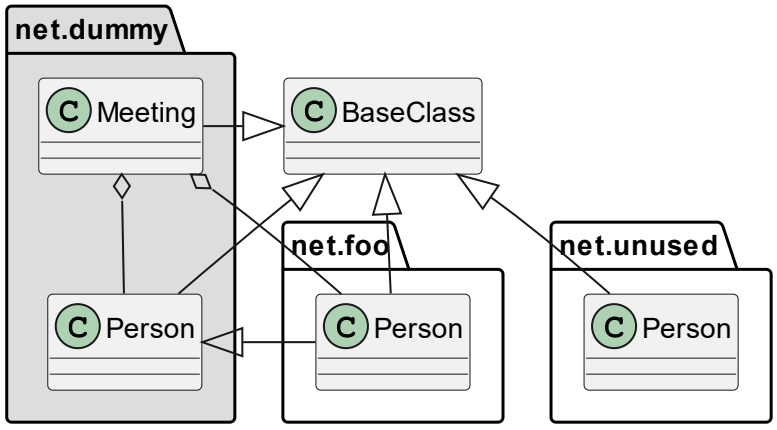
Scope: An enterprise, software system or container.

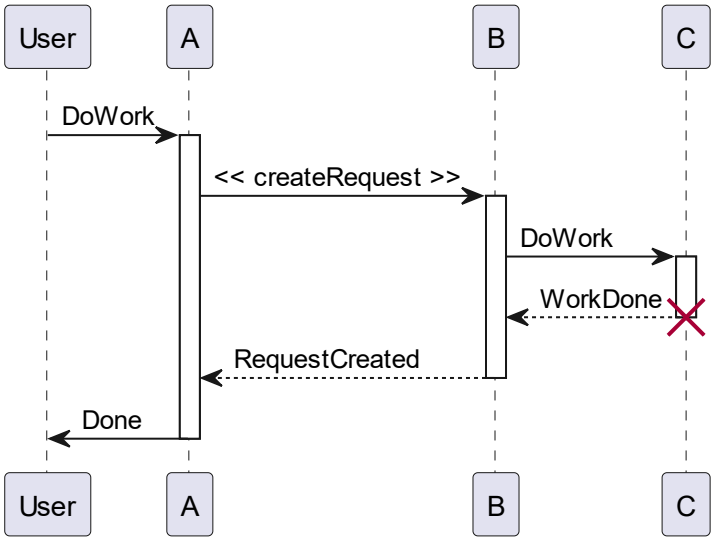
Primary and supporting elements: Depends on the diagram scope; enterprise (see System Landscape diagram), software system (see System Context or Container diagrams), container (see Component diagram).

Intended audience: Technical and non-technical people, inside and outside of the software development team.

Extended Docs

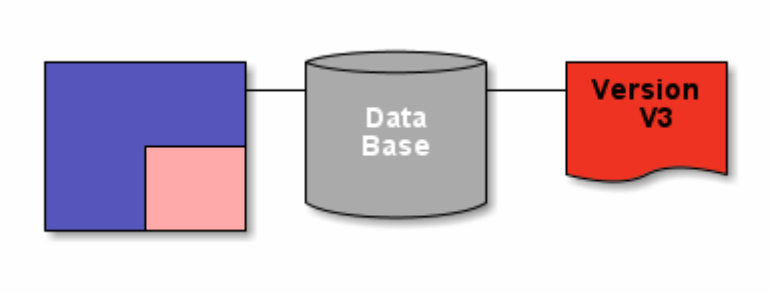
`\1 Internet Banking System\Single Page Application\Extended Docs`





Multiple markdowns can be ordered using `<name>.1.md, <name>.2.md .. <name>.<n>.md`

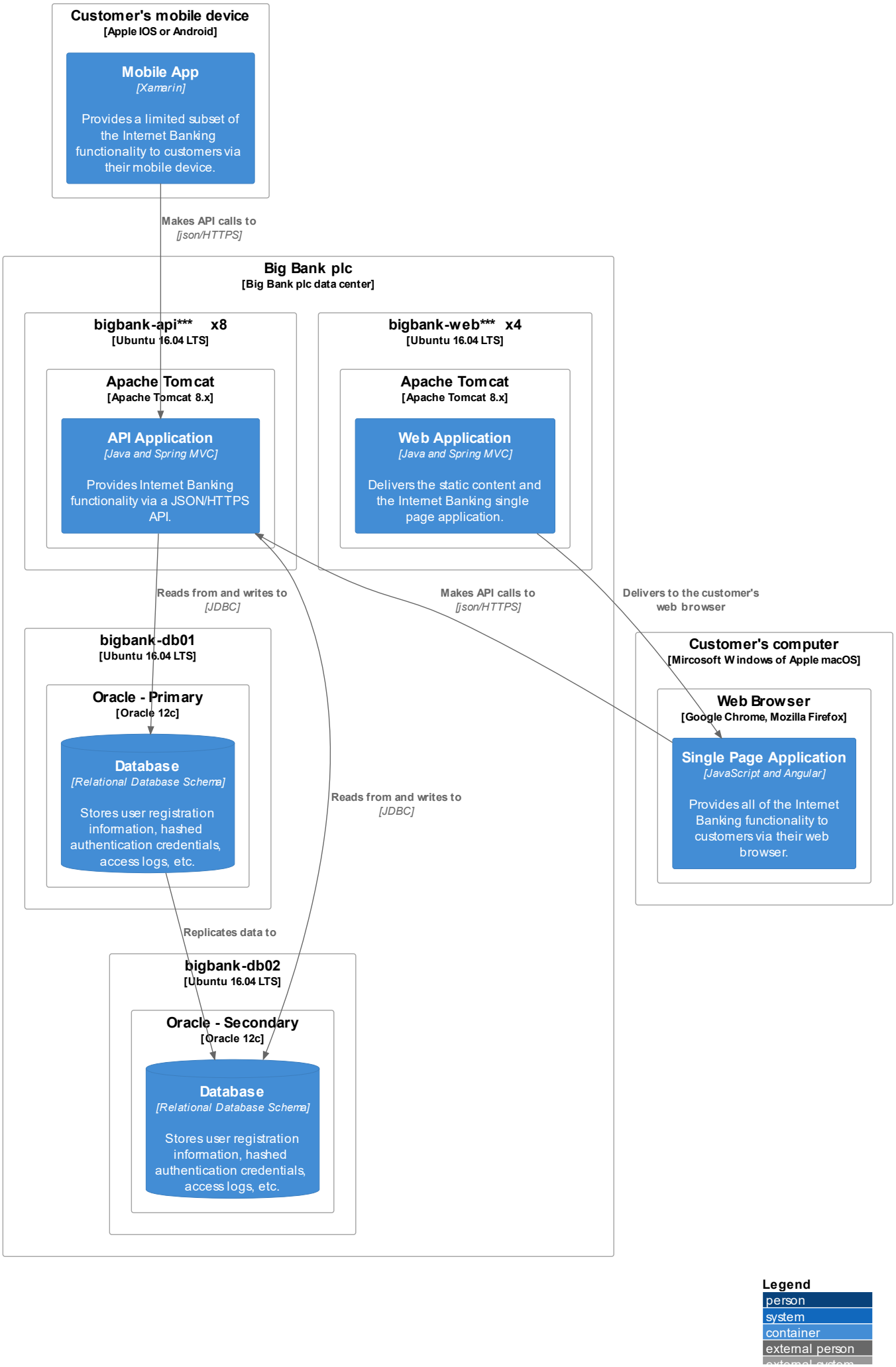
You can choose where to place a certain diagram by using `![name](<diagram name>.puml)`



Feel free to add any additional details necesary.

2 Deployment

```
\2 Deployment
```


Customer's mobile device
[Apple IOS or Android]
Mobile App
[Xamarin]
Provides a limited subset of the Internet Banking functionality to customers via their mobile device.
Makes API calls to
[json/HTTPS]
Big Bank plc
[Big Bank plc data center]
bigbank-api*** x8
[Ubuntu 16.04 LTS]
Apache Tomcat
[Apache Tomcat 8.x]
API Application
[Java and Spring MVC]
Provides Internet Banking functionality via a JSON/HTTPS API.
bigbank-web*** x4
[Ubuntu 16.04 LTS]
Apache Tomcat
[Apache Tomcat 8.x]
Web Application
[Java and Spring MVC]
Delivers the static content and the Internet Banking single page application.
Reads from and writes to
[JDBC]
Makes API calls to
[json/HTTPS]
Delivers to the customer's web browser
bigbank-db01
[Ubuntu 16.04 LTS]
Oracle - Primary
[Oracle 12c]
Database
[Relational Database Schema]
Stores user registration information, hashed authentication credentials, access logs, etc.
Customer's computer
[Mircosoft Windows of Apple macOS]
Web Browser
[Google Chrome, Mozilla Firefox]
Single Page Application
[JavaScript and Angular]
Provides all of the Internet Banking functionality to customers via their web browser.
Reads from and writes to
[JDBC]
Replicates data to
bigbank-db02
[Ubuntu 16.04 LTS]
Oracle - Secondary
[Oracle 12c]
Database
[Relational Database Schema]
Stores user registration information, hashed authentication credentials, access logs, etc.
Legend
person
system
container
external person

external system
external container

Deployment diagram

A deployment diagram allows you to illustrate how containers in the static model are mapped to infrastructure. This deployment diagram is based upon a UML deployment diagram, although simplified slightly to show the mapping between containers and deployment nodes. A deployment node is something like physical infrastructure (e.g. a physical server or device), virtualised infrastructure (e.g. IaaS, PaaS, a virtual machine), containerised infrastructure (e.g. a Docker container), an execution environment (e.g. a database server, Java EE web/application server, Microsoft IIS), etc. Deployment nodes can be nested.

Scope: A single software system.

Primary elements: Deployment nodes and containers within the software system in scope.

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