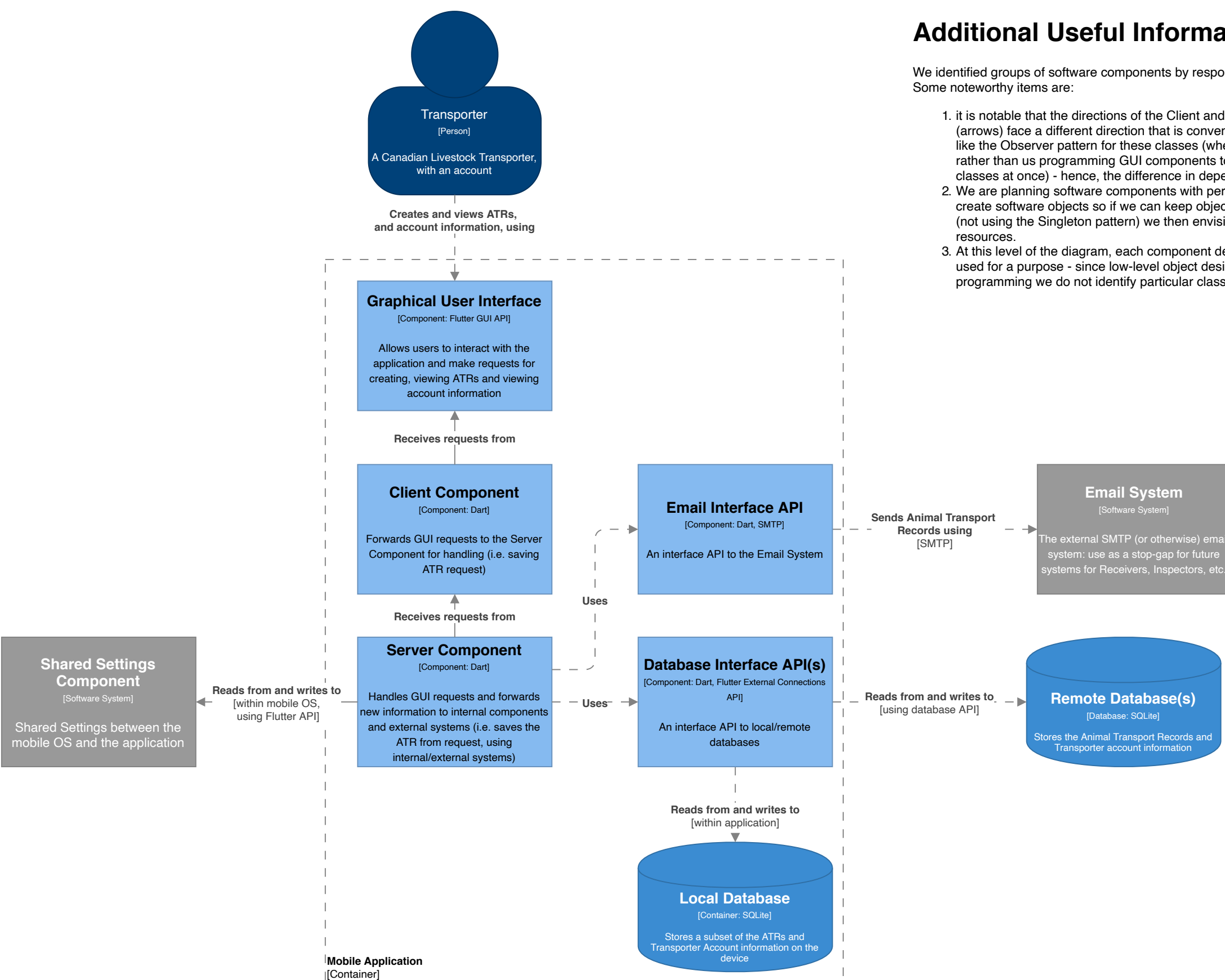# Additional Useful Information

We identified groups of software components by responsibility for the mobile application. Some noteworthy items are:

1. it is notable that the directions of the Client and Server Component data flow items (arrows) face a different direction that is conventional: we intend to use something like the Observer pattern for these classes (where GUI updates are "listened" for rather than us programming GUI components to create/use/destroy several client classes at once) - hence, the difference in dependency
2. We are planning software components with persistence in mind: it takes time to create software objects so if we can keep objects alive as long as they are needed (not using the Singleton pattern) we then envision better use of mobile OS resources.
3. At this level of the diagram, each component describes groups of software objects used for a purpose - since low-level object design changes often when programming we do not identify particular classes.

Transporter
[Person]

A Canadian Livestock Transporter, with an account

Creates and views ATRs, and account information, using

Graphical User Interface
[Component: Flutter GUI API]

Allows users to interact with the application and make requests for creating, viewing ATRs and viewing account information

Receives requests from

Client Component
[Component: Dart]

Forwards GUI requests to the Server Component for handling (i.e. saving ATR request)

Receives requests from

Server Component
[Component: Dart]

Handles GUI requests and forwards new information to internal components and external systems (i.e. saves the ATR from request, using internal/external systems)

Email Interface API
[Component: Dart, SMTP]

An interface API to the Email System

Sends Animal Transport Records using [SMTP]

Email System
[Software System]

The external SMTP (or otherwise) email system: use as a stop-gap for future systems for Receivers, Inspectors, etc.

Uses

Database Interface API(s)
[Component: Dart, Flutter External Connections API]

An interface API to local/remote databases

Uses

Reads from and writes to [using database API]

Remote Database(s)
[Database: SQLite]

Stores the Animal Transport Records and Transporter account information

Shared Settings Component
[Software System]

Shared Settings between the mobile OS and the application

Reads from and writes to [within mobile OS, using Flutter API]

Reads from and writes to [within application]

Local Database
[Container: SQLite]

Stores a subset of the ATRs and Transporter Account information on the device

Mobile Application
[Container]

Component Diagram for the Mobile Application Container

Identifies the groups of software components in the Mobile Application, grouped by responsibility

Last Modified: December 12, 2020