Software Architecture Specification for No Kill Louisville Android App

Prepared by Nick Curry,

Sokheang Leang,

Christianne Maene,

Kei R

Indiana University Southeast

Individual Contributions

Nick Curry: team leader, developer, tester

Sokheang Leang: developer, tester

Christianne Maene: developer, tester

Kei R: developer, tester

Personal Information

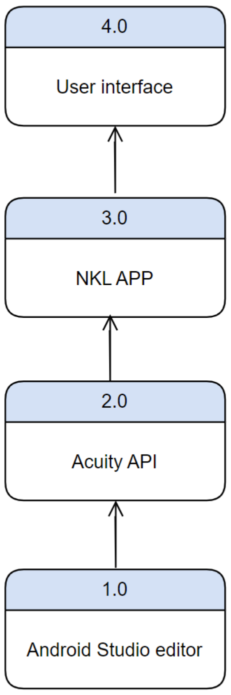
Nick Curry: nscurry@iu.edu

Sokheang Leang: sleang@iu.edu

Christianne Maene: imaene03@gmail.com

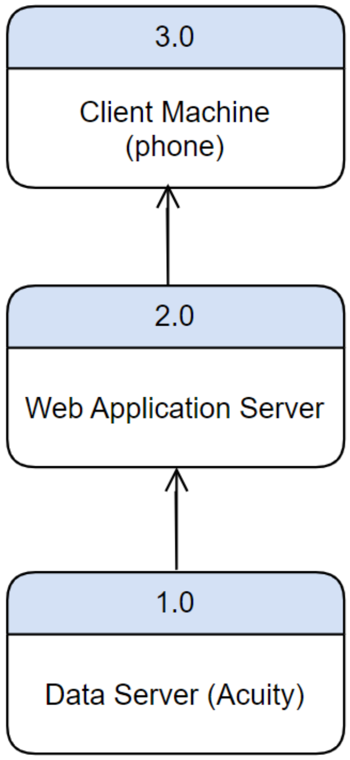
Kei R: irouf@iu.edu

Subsystem Decomposition



The user interface is what the user interacts with. The user interface is the face of the app which includes all the buttons, options and interactions. The Application itself contains the code behind the interactions that the user sees. It also works with the API to make writing code easier; it is a way of not reinventing the wheel. Below is the editor that we are using. In our case, we are using the Jetbrains tool Android Studio for an android phone app development tool.

Hardware and Software Mapping



The pet food bank has a data storage component with Acuity. We are obligated to use this database as it is the only place where users can get validated from. So the acuity API will connect us to the Data Server. From there, the android application will work with the web application server to display options. Some of these options are authentication; the app talks with the database to validate a true user of the app. Another option is signing in a pet owner and signing them out. All of these functionalities will depend on our ability to work with the Acuity data server.

Persistent Data Management

Most of the data that will be presented is sensitive. People's personal phone numbers and tax information are a few examples of data that is stored on the database. The pet food bank needs to verify the income of pet owners upon deciding how to distribute pet food. It is therefore imperative that information on income, and other personal identifiable information be committed to the database for verification.

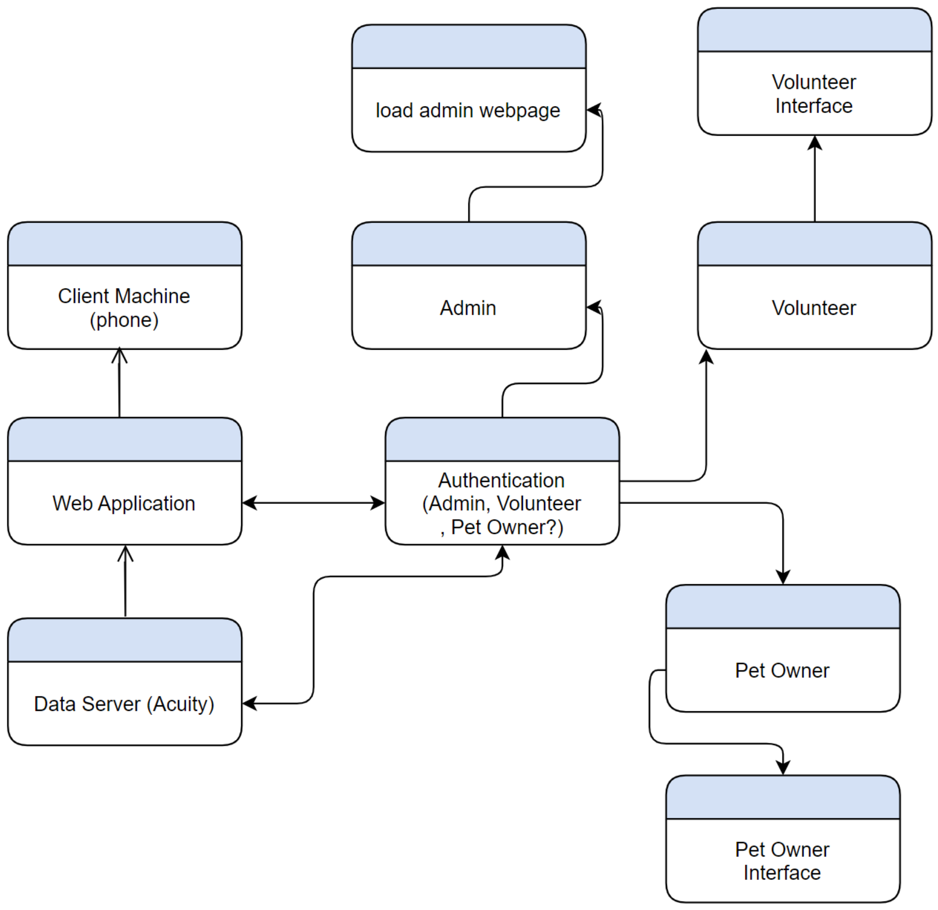
Access Control and Security

Any good application has levels of privileges. Not all privileges are granted to any one user. As a system administrator, the role is given to the president of the pet food bank. They are responsible for uploading sensitive documents to the database. They are also responsible for keeping track of valid pet owner recipients and those that should be opted out of the system. This person can see if a pet owner is missing documentation or if a pet owner is all set.

As a volunteer, they should not be able to see any sensitive information concerning a pet owner. They should only see a green checkmark signifying that a pet owner does not need to bring documentation. Otherwise, they should see a red 'x' signifying that the pet owner needs documentation. A list of required documentation may be specified. However, volunteers cannot upload this documentation to the database.

As a pet owner, they should be able to verify that they have arrived on time for their appointment. The database should validate them based on their phone numbers and names.

Global Software Control



We will be using event driven flows. As given in the diagram from the application, we will authenticate to the data server from acuity. Based on the authentication, users will be admin or volunteer or pet owner. Each one of those users have their interfaces.

Boundary Conditions

Boundary conditions are documented in the unit test cases. A lot of pain has been taken to test a huge percentage of the code. Exceptions and errors should be handled by these unit tests. Some common tests might include if the back button works and if our security and authentication is secure enough and that there are no foreseen loopholes for a hacking scheme. Tests should be daily. Tests should be written before the actual code part. Since this is an android application, users can only use android devices. However, Acuity regularly tests code and keeps things up to date. Loading the app in 4 years into a brand new android phone or tablet should not be a hardship.

Component Types

Regardless of the techniques being used, we can say that any system can be said to be composed of nine basic component types:

Use Cases

The Process starts when the customer gets on the app and checks in, then the app verifies to make sure that this person is in the database and is eligible for the services they provide. After that the app checks in the person. Also if they are a new user , their information will be saved to the database.

Functions

The functions of this application are to check in the user, verify that this person is in the database, and to confirm the time and date they are supposed to arrive.

Trigger

Our triggers for this application starts with the person putting in their information, then the application will connect to their database to save it. Finally our application will confirm and verify the time and date they should arrive.

Data Stores

For now what we know of the database is that it is run by Acuity. All data is stored by acuity. All data is subject to change. Pet food recipients can update how many pets they have, what their income is and so forth. To discuss data stores, names and SSN numbers are stored.

Data Flows

Data flows involve things that change such as the number of pets, income levels, rent and utilities, etc…

Data Elements

Because the database is runned by No Kill Louisville, the only known element are: First Name, Last Name, Phone Number, Email, Name of Spouse/Roommate/Adult Child, Address, the person picking up pet food, employment status, government aid, etc. based.

Data Processors

The primary data processors are the people (developers, admin, volunteers, and pet owners), the android phones or tablets, application, and data servers. All of these components are crucial to the use of the NKL android phone app.

Data Storage

The data is stored by the organization running No Kill Louisville in a database(Acuity).

Data Connections

Data is connected with the application via the API which allows the application to read and write data from the database.

Data Actos/External Entities

Some external entities are governmental offices, IRS, and different offices that are responsible for the official tax documents and income documents that pet owners might submit. This is why keeping the information of pet owners confidential is important. The developers of this app would like to mention that they do not see the database but only have a deep knowledge of how the database interacts with their application.

Useful Definitions

Use Cases are an ordered set of processes, initiated by a specific trigger (e.g., transaction, end of day), which accomplish a meaningful unit of work from the perspective of the user.

Functions are context independent processes that transform data and/or determine the state of entities.

Triggers are the events that intiate Use Cases. There are three types of triggers: time triggers, state triggers and transaction triggers.

Data stores are data at rest. Data flows are data in movement between two processes, a process and a data store, etc.

Data elements are the atomic units within data flows and data stores.

Processors are the components which execute the processes and events (i.e., computers and people).

Data storage is the repository in which the data stores reside (e.g., disks, tapes, filing cabinets).

Data connections are the pipelines through which the data flows flow (e.g., communications network, the mail).

Actors/External entities are people or systems outside the scope of the system under investigation but with which it must interface.

Each of these components has many properties or attributes which are needed to fully describe them. For example, in describing a process we can state its algorithm, who or what executes it, where it takes place, when it takes place, how much information it must process, etc. Figure 1 lists the properties which can be described for the various component types.

In a given project and for a given component, the properties which must be gathered/defined may vary. The SDLC must allow for this flexibility versus an all-ornothing approach.