Cyndi Ai (aixin@cs.washington.edu)
Project Proposal
Jan 6, 2016

My Doggy Woof!

Vision

My Doggy Woof! is a light weighted Android app that seeks to be a community-like application that provides checklist for for any new dog owners or adopters, guides for caring for or training dogs, reviews or best place getting of all kinds of pet's products around you. It is designed for dog owners, or even potential owners by fetching them information needed in providing them the most accurate, helpful, and official suggestions or guides throughout the time.

The partial principle behind *My Doggy Woof!* has been used in some other apps now available in the market, such as the *PetSmart* App, or the *Good Dog Guide* App. However, those already developed apps are either not comprehensive or skewed towards the selling of the products, therefore it makes pretty hard for the users to achieve all kinds of useful and accurate information from one place. Additionally, none of the existing pet's app are providing a community-like social network, which encourages the users to be engaged and to post valuable/precise information. Having *My Doggy Woof!* will save dog owners a lot of time by not searching on the web with their specific questions and looking through many websites for an accurate solution, instead just use *My Doggy Woof!*.

Thus, based of our research results, My Doggy Woof! is a very unique and helpful solutions to all dog owners and it provides a free and realistic place for the owners to exchange thoughts or share experiences.

Software Architecture

Sample technology usages,

a. Android SDK – to develop the application for Android phones
b. Android Framework – to develop the user interface
c. SQLite database – to save the data in the forum or guide provided by the senior users
d. Object Relational Mapper – to make interaction with the database easier
e. Yelp API – to look for possible pet stores around the user
f. JavaScript – to extract the store website information from the Yelp search results
g. DNSBLs – to detect known domains of spammers
h. Other possible spam detection technologies
i. www.petparents.com - to extract useful guides from this external database

The project can be divided up into three main features,

a. checklist for dog owners
b. guides for training dogs
c. product suggestions

All three features can follow a Model-View-Controller structure, whereas the first two features are pretty similar. We will start off with using the external data from other websites and we will then add information written by the senior users or most read threads.

- Model – it will interact with the Android file system to store and retrieve data from the database and it will also first retrieve data from the external databases.
- View – handles the display of the information
- Controller – It will fetch any data from the external databases or APIs according to the users' location.

The last feature is a little different,

- Model – fetches the reviews, available stores from the Android file system
- View – displays the reviews for each of the products
- Controller – perform the searches and keeps track of the time and any other needed information as users write the posts.

Challenges and Risks

The most daunting challenge of *My Doggy Woof!* is spam detection. As we promised earlier, the application will provide the most accurate information to any dog owners. Thus, one job we need to do is avoid spam posted by product owners, or pet store owners. We need to make the *My Doggy Woof!* to be a fair and precise source of information available to our users. Other than making use of the DNSBLs (DNS Blacklists, the most popular list of domain names of known spammers), we also need to figure out a way to check for spam specifically for our app, which will most likely be basing off the content of the posts or comments.

Another challenge we might be facing is making large use of external databases and the great use of the external APIs like Yelp. Since our users are most likely to be around one area or at least one city most of the time. We need to compare the trade offs of doing a great number of searches per user search time or we should make the usage of the cache. We need pay careful attention and to handle the risk when a lot of searches are done by using an external resource that is out of our control. We need to find out a way of handling the situation when external resource is down (i.e. too many searches per day by using the Yelp API that exceeds the daily limit, external database is down and etc).