Central Chat: A SaaS chat service powered by Android
Stan Gravchikov (Project Lead), Ivan Gutierrez, Joel Darling, Justin Tilley

1 PROBLEM AND MOTIVATION

Our group was tasked with developing a full featured mobile application that runs on Android OS which would serve as an amalgamation of all the skills, knowledge and techniques we gained during the Mobile Applications Programming course.

Since our group was relatively large, we decided to develop an instant messenger SaaS (Software as a Service), similar to the likes of WhatsApp or older, non mobile based services such as AIM (AOL Instant Message), which both allows for bi and multilateral communication between connected users.

2 BACKGROUND

Within the past 5 years, SaaS based applications have risen in popularity, mostly due to the proliferation of android enabled mobile devices. With Google Play and Google Cloud Messaging Services, android developers are more equipped than ever before to take full advantage of SaaS architectures, which is why programs such as Telegram and WhatsApp (both of which are SaaS instant messenger apps) are becoming so popular.

It is our hope to use these tools to our advantage to create an instant messaging program which takes use of not only the android SDK but also Google Play as well as Google Cloud Services.

3 SOLUTION

2-tiered approach

- Android Client App
- PHP Server (remote server)

Google Cloud Messaging (GCM),

- Store smart client data.
- Automatically queue messages
- Automatically pull messages

GCM Broadcast Receiver

- Redirect the message to the notification intent

GCM Notification Service

- Filter messages based on the "to" address of the recipient
- Makes use of the observer Design Pattern

Shared Preferences

- Store the messages
- Store user data

Dynamic Fragments

- Display chat in a portrait layout
- Display chat and a user list in a landscape layout

Threads

- Updating user list and message list dynamically

4 RESULT

- Together, all of these elements allowed for a full-fledged instant messaging application which allows for clients to have both bi and multilateral communication without them having to understand how the server handles the messaging, or that there is a server to begin with. As with most SaaS applications, a user only needs a local client which gives them access to the service. However, unlike most SaaS applications, we made full use of Android’s subsystems for saving, parsing and filtering of content locally, which significantly lowers potential server overhead and costs.

- Using a database allows us to retain a list of all connect users so that we can ensure user concurrency within the application. GCM allows us to effortlessly send offline messages to users, who would then receive said messages which they powered on their device.

- The use of fragments allows us to segment the application in such a way that we can maximize screen real estate for large devices by showing both the user list and chat window at all times, while giving users of smaller devices a smaller view so that their messages remain readable.