TODO List Manager on Steroids
Architecture Document
Team 3.09

1. Introduction

The goal of the project is to extend the existent TODO List Application on Android to a Web application and to provide sync functionalities between the Android and the Web application, thereby enhancing the user-experience.

2. Architecture

The architecture of this ToDo List Manager application is displayed below. Google Cloud Endpoints is used to create an interface for various platforms. The Google App engine is used to create the web application. The Google App Engine is also used to create REST APIs that put, remove and edit tasks and user information from the Google DataStore. Once these APIs are created, Google Cloud Endpoints is used. This creates a set of libraries and Java classes that can be placed in any Java based application.

For example the libraries and classes generated from the Google App Engine application can be placed in the android application, thus enabling the android application to use these APIs. These REST APIs can also be called from Javascript files in the web application.

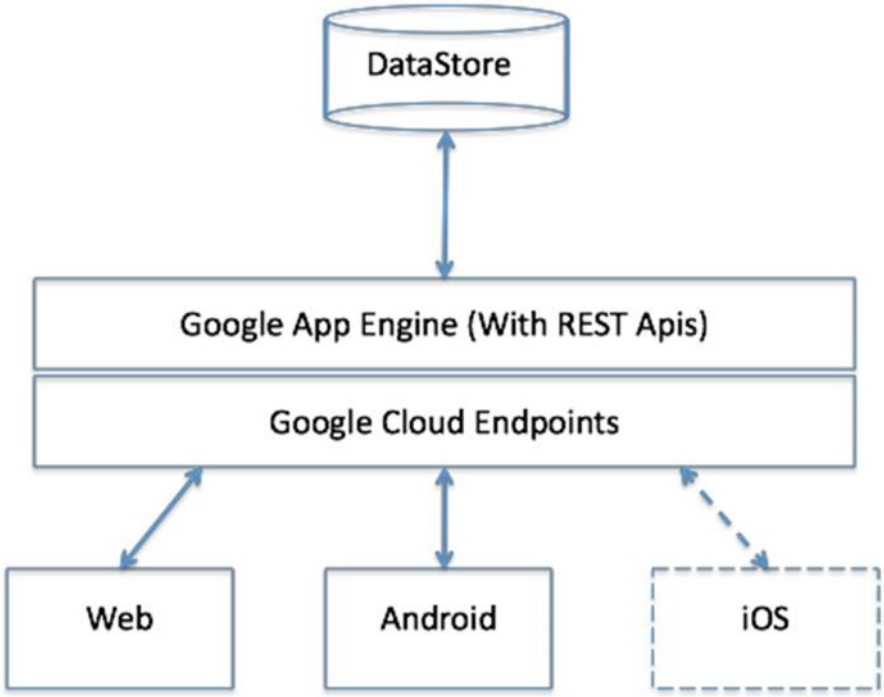

Figure 1. Architecture diagram.

Google Cloud Endpoints can also be used to integrate iOS applications. This can be a potential future plan.

3. Logical View

There are two components to this application; the mobile application with its local database and the web application with the central database.

The central database keeps track of all existing user accounts. The local database, on the other hand keeps track of the accounts used on that device. The purpose of this structure is to enable push notifications for the user accounts on the local device.

This is depicted in the figure below. For example, device 1 has user A and B account information stored in its local database

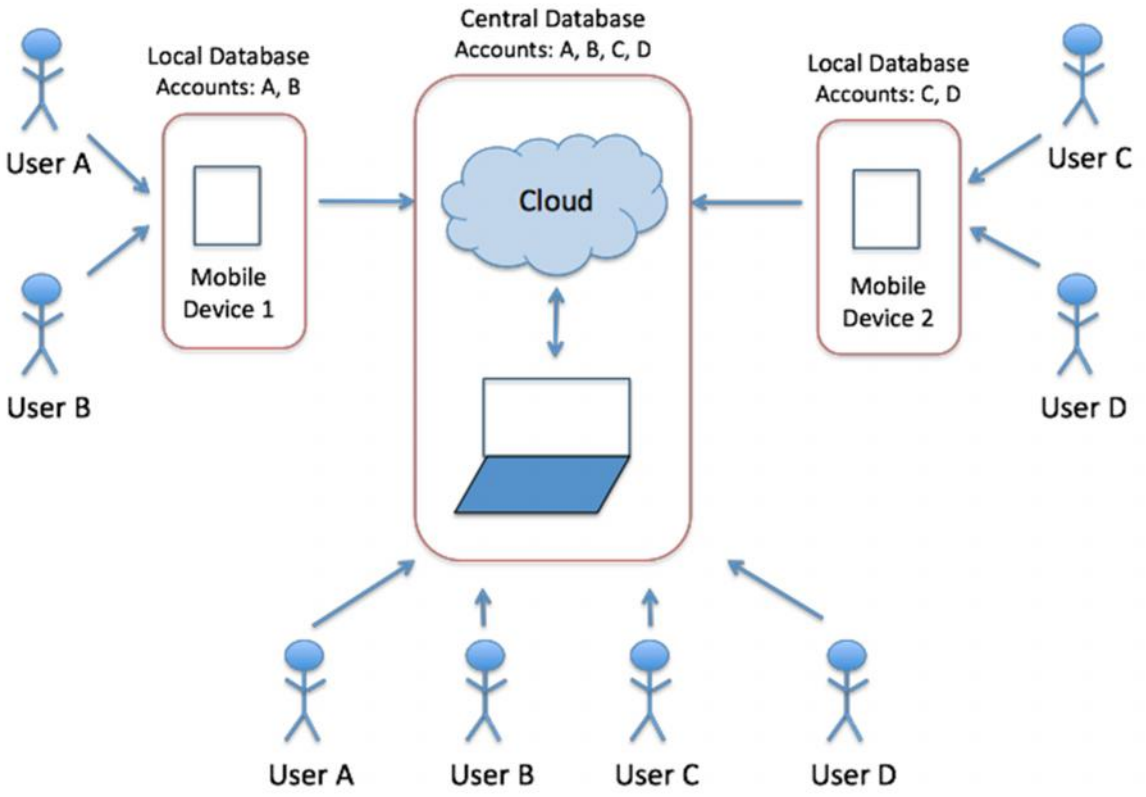

Figure 2. Architecture diagram distinguishing between central and local databases.

When a user tries to login to a new device, i.e, first time login, the device realizes that this user is new to the local database and therefore presents the user with a dialog box asking for permissions for push notifications. This is further discussed in the use case document.

4. Conflict Policies

Both account conflicts and tasks conflicts will be handled based on timestamps. For example, if the same account is edited in both the mobile and web platform, the latest one overrides the earlier changes in case of a sync. If the two changes on the two platforms happen to be at the same time (same timestamp), then the web app version will override the mobile app version.

4.1. Account Conflicts

The username of each account is unique to the account. Therefore every time a change is made to the username, the central database is checked to see the availability of the new username.

Any account change on the mobile side automatically enforces a sync. If an account is removed from the device, that account is simply removed from the local database but still remains in the central database. However, if the account is deleted, it is deleted from both mobile and web platforms. To avoid deleting the account by accident, the user will be shown a dialog box to confirm his/her action.

If a change is made on the web platform, there is no automatic sync. This raises some tricky situations. For example, if an account is deleted from the web platform, and the user adds tasks to the same account on the mobile platform. In such scenarios, every time the account information is changed or deleted, the older information is temporarily saved till there is a sync from the mobile device. In the above example, the sync will restore the deleted account and show an error message that notifies the user that the account had been deleted and has been restored.

4.2. Task Conflicts

Conflict in tasks are handled using timestamps, therefore at the time of sync, if the same task differs in the two platforms, the latest one will override the other. If the timestamp is the same, than the web app version takes preference.

Some syncs are not as straightforward as others, for example, if a task is deleted from the web platform and the same task is edited from the mobile platform without syncing it. When the sync is made it checks if the task has been deleted on either platforms. If so, it takes the latest one. If both the tasks

are deleted, then the task remains deleted. If one has been deleted and the other task has an earlier time stamp, the delete (which was done after) prevails. However, if the edit was made after the delete, than the task is restored in both platforms with a message alerting the user that the task had been recovered.