TODO List Manager for Android
Initial Use Case Model
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

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

The mobile application can store multiple accounts in its local database. Different mobile devices may store different user information in their individual local databases. The central database on the other hand keeps track of all existing user accounts. This is depicted in the figure below. For example, user A can only access his/her account through mobile device 1 and not through device 2. However, all users can access their accounts from the web application since the web application uses the central database.

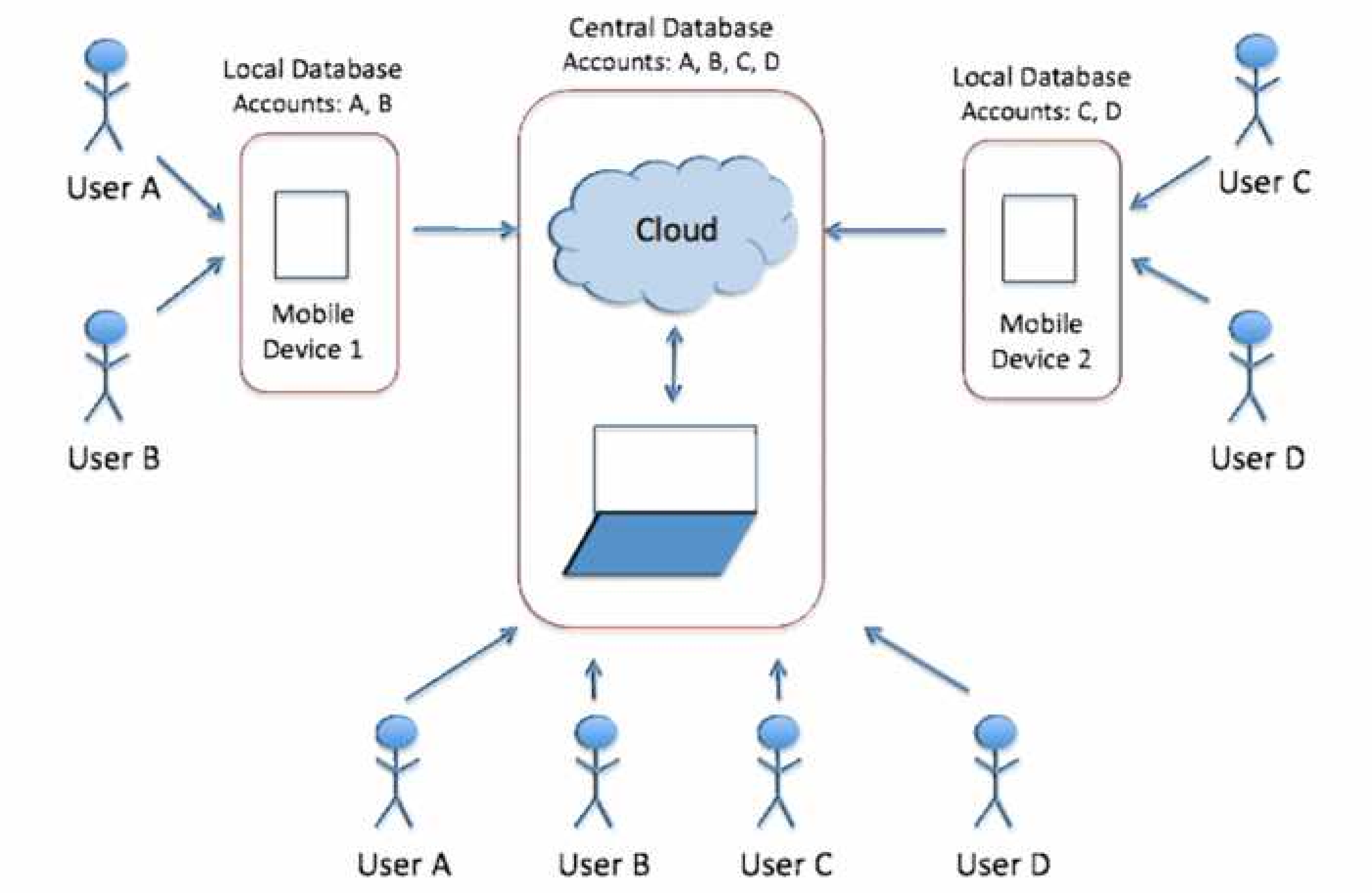

Figure 1. Architecture diagram distinguishing between central and local databases.

3 Use Cases

3.1 Accounts

The mobile application user and the web application users have different use cases due to the distinct architecture with local and central databases.

3.1.1 Web Application

On the web platform the user has the following uses cases:

- Create a new account: this checks the availability of the specified username in the central database.
- Login: verifies information in central database.
- Logout
- Edit Account Information
- Delete Account

A use case diagram is provided below.

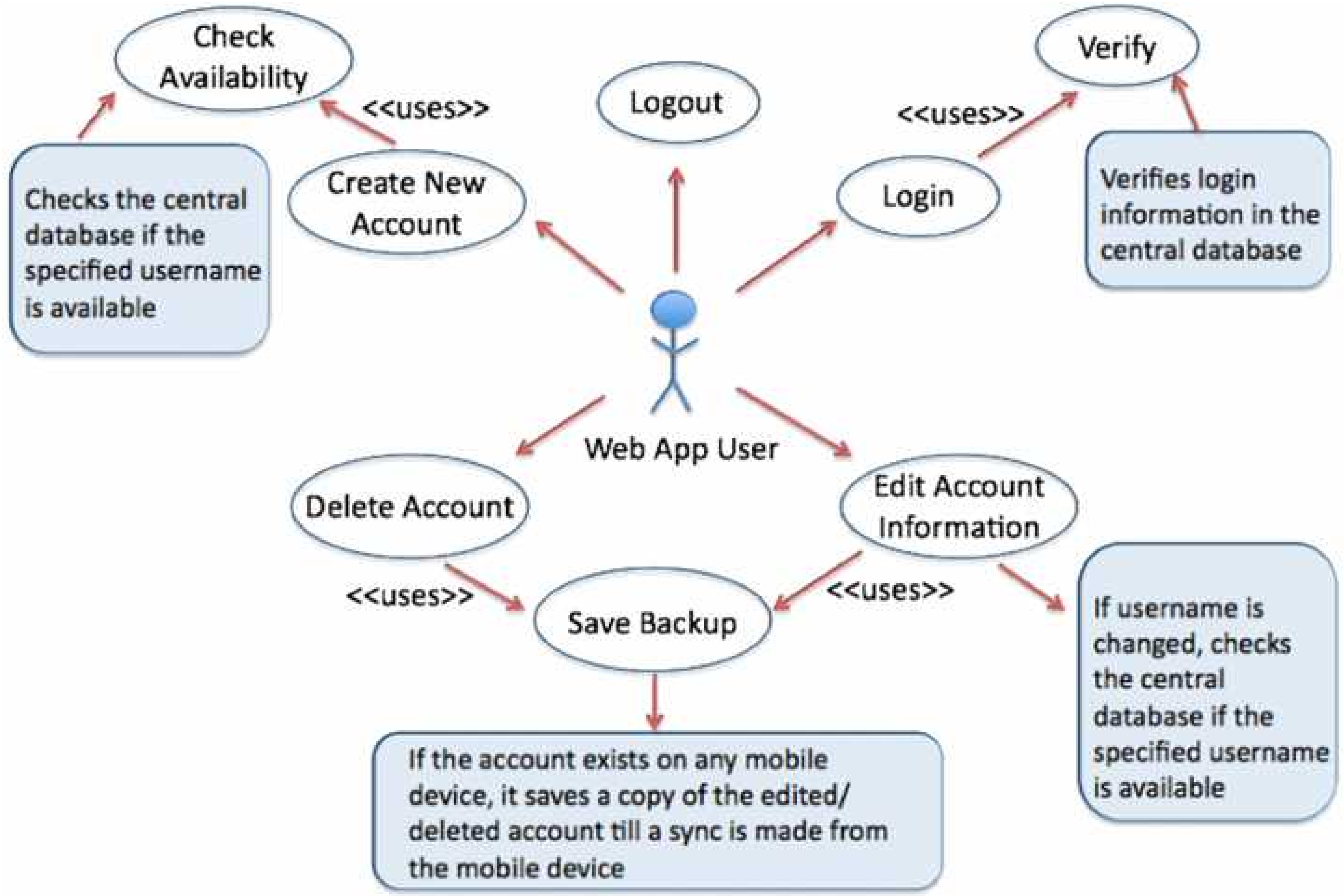
Check Availability
<<uses>>
Logout
<<uses>>
Verify
Checks the central database if the specified username is available
Create New Account
Login
Verifies login information in the central database
Web App User
Delete Account
Edit Account Information
<<uses>>
Save Backup
<<uses>>
If username is changed, checks the central database if the specified username is available
If the account exists on any mobile device, it saves a copy of the edited/ deleted account till a sync is made from the mobile device

Figure 2: Accounts related use case diagram for web app user

Editing account information or deleting the account will not reflect on the mobile app till am implicit sync is made or the app is re-launched. In this time the user may make local changes to the account on the mobile app that has deleted or edited in the central database. For this reason the changes are backed up till async is made.

3.1.2 Mobile Application

On the mobile platform the user has the following uses cases:

- Create a new account: this checks the availability of the specified username in the central database.
- Add an existing account (in the central database) to the mobile device: This does a login verification in the central database and triggers a sync for that particular account.
- Login: verifies information in the local database.
- Logout
- Sync
- Edit Account Information
- Remove Account from device: deletes the account from the local database only.
- Delete Account: deletes the account from both the local and central directory

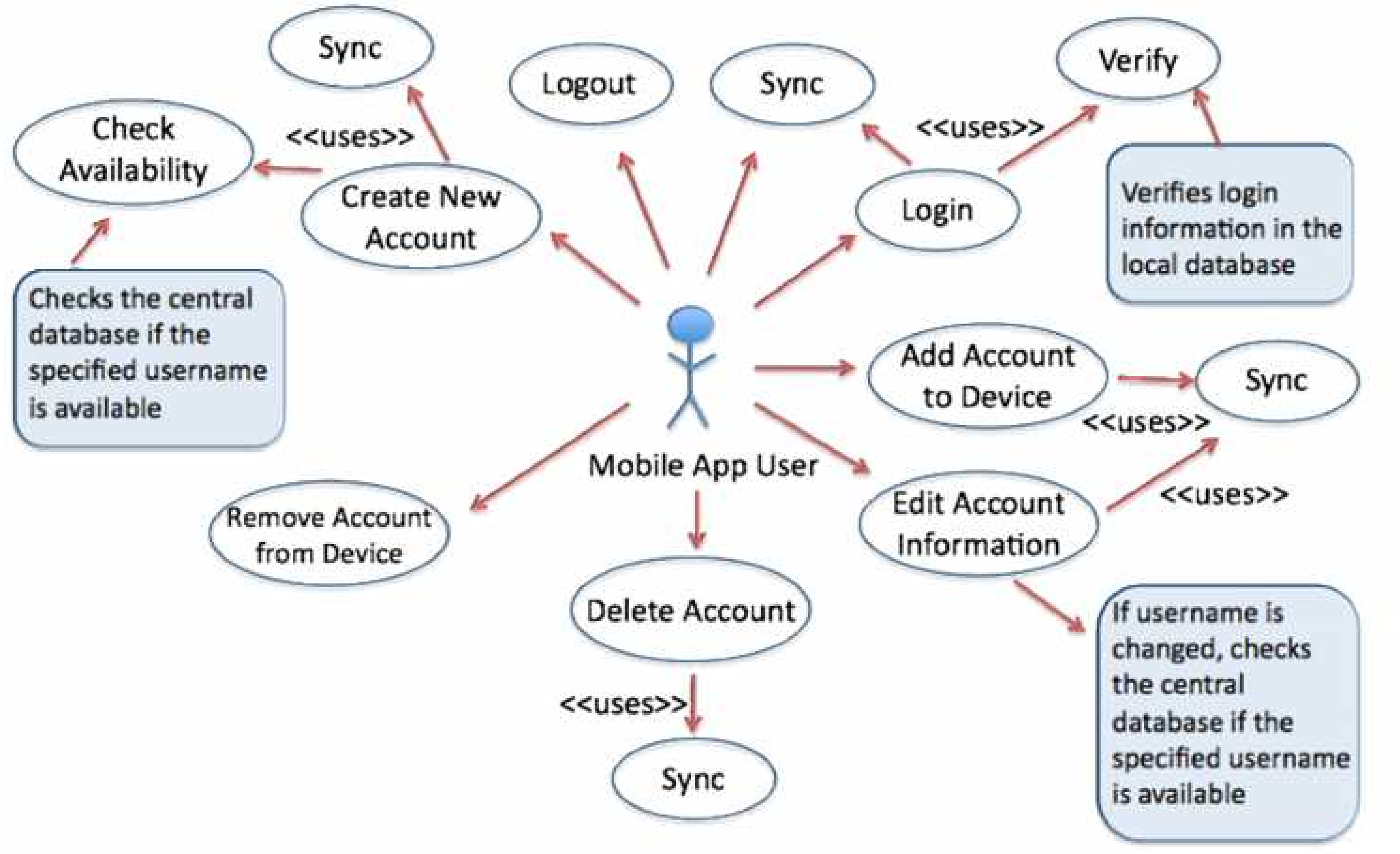
Sync
Logout
Sync
Verify
Check Availability
<<uses>>
Create New Account
<<uses>>
Login
Verifies login information in the local database
Checks the central database if the specified username is available
Add Account to Device
Sync
<<uses>>
Mobile App User
Remove Account from Device
Edit Account Information
<<uses>>
Delete Account
If username is changed, checks the central database if the specified username is available
<<uses>>
Sync

Figure 3: Accounts related use case diagram for mobile app user

3.2 Tasks

Both mobile app users and web app users have similar use cases. Both the users can

- Create tasks
- View tasks
- Edit tasks
- Check/uncheck tasks
- Hide/show completed tasks
- Delete Completed tasks
- Delete tasks

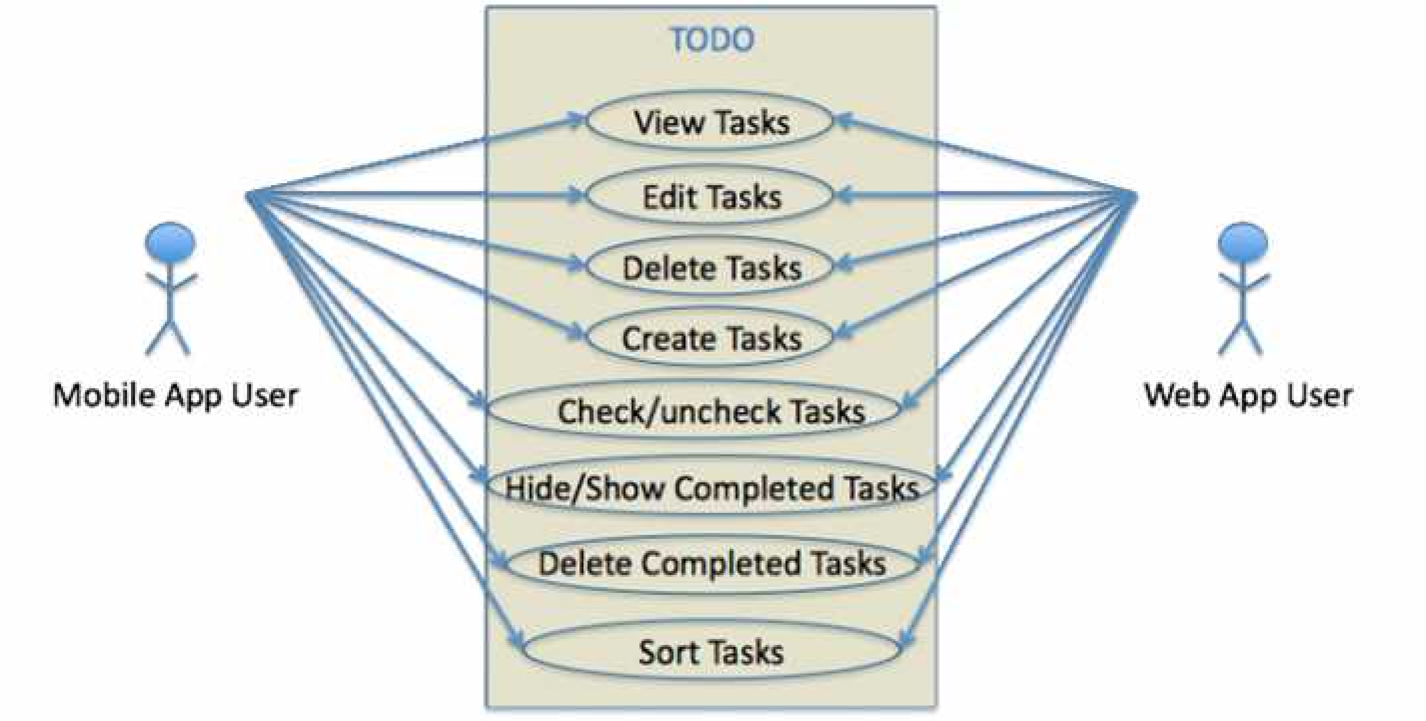

Figure 4: Task related use case diagram for mobile and web app user

3.3 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

3.3.1 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 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 eliminate deleting the accounts 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 snc 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.

3.3.1 Tasks 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.