Software Architecture
Team 3.02

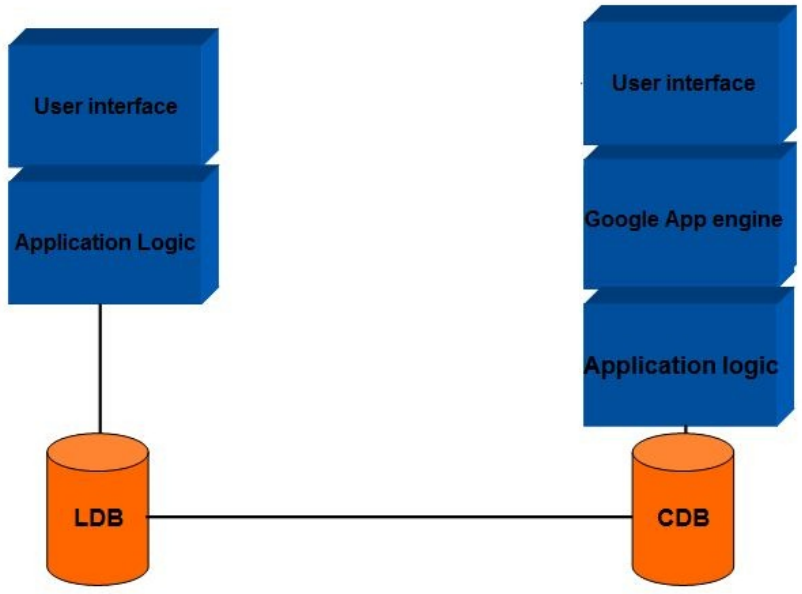
1 Introduction

Do-D-Due is an application that allows a user to manage their daily tasks. This application allows the user to create a task list, edit and delete it via the Android application or the web interface. The tasks will have a name, description, due date and priority. The user will be able to check off items in the list and will be able to hide them. The application will support multiple users, each one with their own lists.

2 Architecture Overview

The MTM, WTM applications will be available to multiple users and each of them may have large lists, thus it must be extensible, maintainable and robust. Two different platforms need to be supported - the web, the mobile. Since the mobile application has already been developed, this architecture represents the existing mobile application. The web application is currently being developed, and the architecture represented in this document will be the architecture used during its development.

2.1 Architecture graphical representation



2.2 Component Overview

Component	**Responsibility**
User Interface (Web)	Users interact with the application through the UI, they can login, add users, tasks and edit tasks through a web interface primarily designed with Javascript and HTML.
Google App Engine	Cloud computing platform to develop and host Web applications.
Application Logic (web)	The processing logic for adding users, user authentication, editing, adding, sorting tasks and written and hosted on the App engine as well. This also handles the database connections.
CDB (Cloud Database)	A cloud database which is used as the main data source for the web application, provides user and task data. It has to be synchronized with the LDB used by the MTM. The CDB must be consistent and also fault tolerant.
User Interface (mobile)	Users interact with the different activity screens to perform login, addition of tasks, editing tasks, etc.
Application Logic	The processing logic for adding users, user authentication, editing, adding, sorting tasks and written and hosted on the App engine as well. This also handles the database connections.
Local Database	A SQLite database which stores the user, task data needed by the mobile application. LDB also has to be synchronized with the CDB.

2.3 Rationale

Our architecture provides a high level overview of the system. This overview roughly divides the individual components into the presentation, application (processing) and the data layers. The architecture shows the mobile and the web todo list managers and the interaction between them. In this case, the interaction happens in the data layer. Both the CDB and LDB have to be in sync, this can be ensured by regular synchronization or explicit saves.