CS-E4100 Fall 2019
Mobile Cloud Computing (5 cr)

Group Project – Cloud-based task management

Deadline Sunday, 8 December 2019 at 23:59 (local time in Finland)

Submission Through the Aalto Version Control System (`https://version.aalto.fi`)

Revision `v1.1 (06.11.2019)`

Projects must be submitted according to the instructions in this document (see the "Submission" section for the details). No other form of submission will be accepted. Deadlines are hard. No extensions will be granted. Should you have any issues in accessing MyCourses, A+ or Aalto Version Control System, contact the course staff at `cs-e4100@aalto.fi` as soon as possible. Also refer to the MyCourses workspace for additional information (`https://mycourses.aalto.fi/course/view.php?id=24344`).

Overview

Summary The purpose of the group project is to implement a task management application that enables planning both personal and collaborative work. The application must support text input and file attachments (e.g., documents, pictures and audio). In addition, tasks must be organized into projects and displayed based on different criteria.

Components The application consists of two components: a frontend and a backend. The *frontend* is the user interface running on a mobile device and must be implemented as an Android application. The *backend* consists of the database and the logic for processing and storage of tasks. The backend relies on products in the Google Cloud Platform (`https://cloud.google.com/products/`), particularly, Firebase (`https://firebase.google.com/`) and the Google App Engine (`https://cloud.google.com/appengine/`).

Frontend

Functionality The frontend *must* be implemented as a *native Android application* and *must* support the following features.

- *User authentication.* The frontend must provide a login screen prompting for user's email and password. In addition, a sign-up form must be available for new users. The form consists of three fields: *display name* (unique), *email* and *password*. If a display name is already taken, the app must inform the user and suggest three alternatives. Users could also provide an optional profile image during sign-up. Use Firebase Authentication (`https://firebase.google.com/docs/auth/`) for this feature.
- *Profile settings.* Users must be able to edit their profile information through the application. In particular, they are only allowed to change their profile image (if applicable) or edit their password.
- *Create a project.* Users can create personal and group projects. In a personal project, a user creates individual tasks. Actions for sharing or collaborating with other users are not available for personal projects. Furthermore, a group project is meant for collaborative work between several users that agree to complete a set of tasks. In both cases,

it is required to store the following attributes of a project: creation date, name and description. The user who creates a project becomes the *project administrator* and this role cannot be transferred to other users. Moreover, users can associate optional attributes to a project: a deadline which is set in a datetime format (chosen from a picker), a badge which identifies the project (an image chosen from the phone's gallery) and keywords (up to 3 terms) which are employed for searching projects.

- *Add members to a project.* After project creation, users can add members through a search based on the display name. A limited set of results must be shown as the display name is typed. For instance, an implementation could require that at least the first five characters of the display name are entered before displaying the list of names. Different criteria can be used as long as the whole list of users is not displayed by default.
- *Add tasks to a project.* Users can add several tasks in a project. Task creation requires three attributes: a description, a status, and a deadline. A task can have three possible statuses: *pending*, *on-going* and *completed*. By default, a task is created with *pending* status. Moreover, a task can be assigned to one or more users. When a task is assigned to a user, the status changes to *on-going*. The task assignment feature is only available for group projects and it can be only performed by the project administrator.
- *Convert image to task.* Users must be provided with an option to extract the text from an image (uploaded from the phone's gallery) and create a task. Text recognition can be supported by using the OCR detection feature in Cloud Vision API (`https://cloud.google.com/vision/docs/ocr`). Other attributes must be also provided when creating a task (i.e., status, deadline).
- *Add attachments to a project.* The application must support uploading files and associating them to a project. Supported file types are text, documents (.pdf), images (.jpg) and audio (.mp4). Images can be uploaded in two ways: by taking a picture from the camera of a smartphone or by choosing an image from the phone's gallery. Users belonging to a project must be able to download the files.
- *Delete a project.* An option for deleting a project must be supported by the application. Project deletion involves removal of all content including tasks and attachments. This feature is only available to the project administrator.
- *Show project content.* For each project, there must be a view to display all its content. The view consists of three tabs: *Tasks*, *Pictures* and *Files*. For all the tabs, the content is populated in lists and sorted by creation date. Moreover, tasks are displayed in a list with checkboxes to update status. Users can change the task status to *completed* by tapping on the corresponding checkbox. The task description will appear crossed out after changing the status.
- *Generate project report.* Users involved in a group project must be able to generate a report of the project. The outcome is a PDF file which contains: the project name, the list of members, and all related task events (e.g., creation, assignment, status change) sorted by date.
- *Show list of created projects.* The frontend must provide a view to display a list of created projects. Each listed project must show basic information about its content. First, it shows the last modification date. Second, it displays a media icon when the project contains media. Finally, it shows the profile image of the project members (only for the first three members). Similarly, the listed projects must include a button with a star icon and a menu of options. The button allows to select a project as favorite, whereas the menu includes option(s) to delete a project (if the user is a project administrator), show project content and generate project report (only available for group projects). In

addition, the view must display the projects list according to three filters: date, favorite projects and those with upcoming deadline. Specifically, projects filtered by date are sorted by modification date; favorite projects are sorted alphabetically by project name; finally, projects with upcoming deadline are those to be completed in a week sorted by earliest deadline.

- *Search for a project.* The application must provide an option to search for projects. Two search criteria are considered: keywords and project name. Search by keywords can be performed in two ways: by showing a list of the keywords stored in the backend (without repetition) or by entering a keyword in an input text. Instead, search by project name can only be performed by entering the (full or partial) name in an input text. Results will show those projects that match the input provided.
- *Image resolution settings.* A settings screen must allow users to specify different resolutions for images. In particular, it allows to specify the image quality when uploading or downloading pictures. The image quality can be *low* (640×480 px), *high* (1280×960 px) and *full* (original size) resolution. The same resolution setting applies for both uploads and downloads. For instance, consider a scenario where user A sets the resolution to full and user B sets the resolution to low. When user A sends a picture, the full resolution version is uploaded. However, user B downloads only the low resolution image. When user B sends a picture, only a low resolution image is uploaded and user A can only receive the low resolution image. Note that the different resolution images are generated in the backend.
- *Notifications.* The application must notify users when they are added to a project, assigned to a task or when a task (project) deadline is approaching (i.e., one day before). Notifications must be received even if the application is in the background.
- *Responsive user interface.* The application must handle all operations without slowing down or freezing the user interface. Processing and downloading should be done in the background. Additionally, the app should properly manage loading and displaying large images without running out of memory.

Backend

Requirements The backend must provide an API for project management as follows.

- *Create a project*, which returns a unique project ID after successful project creation.
- *Delete a project*, which deletes all the content of a project given the project ID. This action is only available to the project administrator.
- *Add members to a project*, given the project ID and a list of members (it can contain one or more users).
- *Create a task*, given the project ID and the task attributes. It returns the task ID after creation.
- *Update a task*, which updates the task status given a task ID.
- *Assign a task to a user(s)*, given the project ID and the task ID.

Architecture The backend must be implemented by using the Google App Engine Flexible Environment. The application should use the Firebase Realtime Database (*the recommended option that is supported by the course staff*, `https://firebase.google.com/docs/database/`) to sync information across different devices and Cloud Storage (`https://firebase.google.com/docs/storage/`) to store media. In addition, you must use Cloud Endpoints (`https://cloud.google.com/endpoints/`) to deploy the API described in the previous section. You could use Cloud Functions (`https://cloud.google.com/functions/`) for sending notifications to devices, generating project reports and handling image resolutions.

Access Only authorized users should be allowed to access the project related information on the database. Specifically, information about a project – including member(s), tasks and user-generated pictures – should be only accessible to members belonging to that particular project. Moreover, the pictures in Cloud Storage should not be publicly available.

Advanced features

Description Implementing the following features *is not required* to complete the project but gives extra points. You can pick any combination of the features listed below. Even though the project itself will be given the extra points for correctly realized features, **the total number of extra points considered for grading is limited to a maximum of 20**.

- *Email notification* (**5 points**). Implement an option to send an email when a deadline is approaching. The email should be sent to the user as a reminder for a task deadline. The subject of the email is the project name and the body should have a message to indicate the upcoming deadline. The time for sending email should be set as an option in the Profile Settings. The user can choose between days and hours before the deadline.
- *Speech to text conversion* (**5 points**). Provide support to extract text from a voice note and convert to a task. Use the Google Cloud Speech-to-Text API (`https://cloud.google.com/speech-to-text/`).
- *Translation API* (**5 points**). Provide an option to translate the description of a task. The target language should be set as an option in the profile settings. Use the Google Cloud Translation API (`https://cloud.google.com/translate/`).
- *Image search* (**5 points**). Implement an option for searching images based on features. Use the MLKit with cloud-based image labeling to categorize images. The features can be predefined and populated in a list. For instance, a user selects the feature *landscape* from the list, and the app will show those images that fulfill the criteria. When clicking an image, the app will display another view with information about the project linked to such image. The view displays the project name, members and a button that redirects to the view for showing the project content. Refer to Section *"Functionality"* under *"Frontend"* for more details.
- *Web frontend* (**5 points**). Write a web frontend that offers the same basic features of the Android application.
- *Mobile app testing with Appium* (**7 points**). Use Appium (`http://appium.io/`) to run unit tests for some of the functions in the frontend.

Submission

Instructions The project source code must be submitted through the Aalto Version Control System (`https://version.aalto.fi`) on the git repository associated with your group. Additional information about using the Aalto Version Control system and the group-specific repositories is available in MyCourses under the "Software project" section (`https://mycourses.aalto.fi/course/view.php?id=24344&section=4`). No explicit action is required to submit the source code: **the last commit before the submission deadline will be used for evaluation**.

Evaluation Each group must show a demo of the software project to the teaching assistants. A time for the demo has to be booked according to the instructions available in MyCourses under the "Software project" section (`https://mycourses.aalto.fi/course/view.php?id=24344&section=4`). The evaluation criteria for the software project are available in MyCourses under the “Software project” section (`https://mycourses.aalto.fi/course/view.php?id=24344&section=4`). Note that these criteria apply for the basic features of the software project (i.e., not those under “Advanced features”).

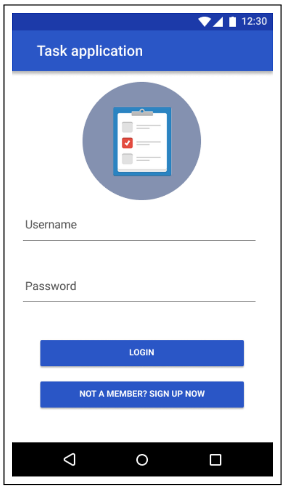
Revision history

1.1 Clarified use of Firebase Authentication.

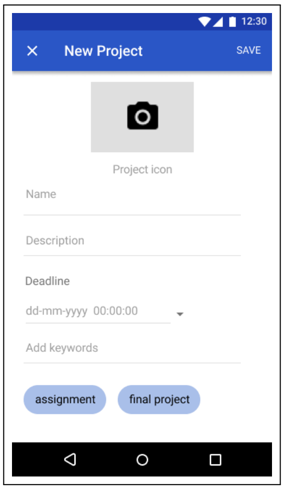
1.0 Clarified use of cloud resources. Updated advanced features.

0.9 Draft (preview) version of the project description.

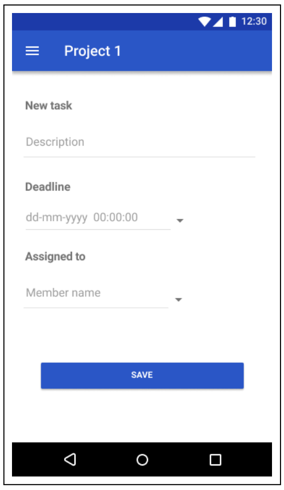
Sample user interface



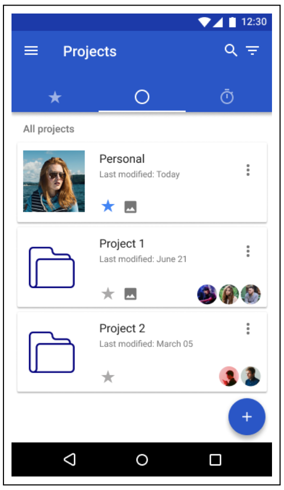
Login



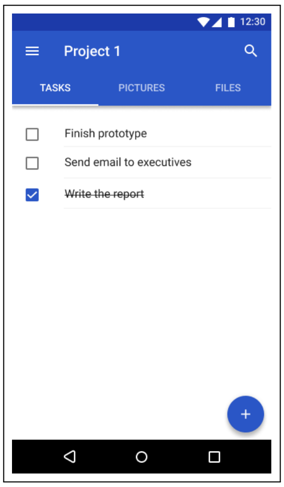
Create a project



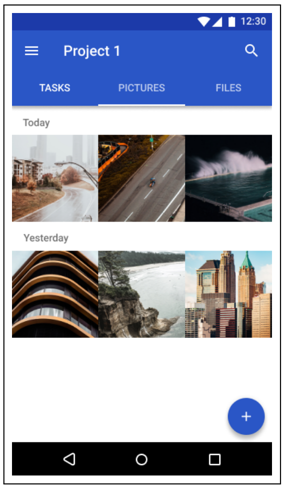
Create a task



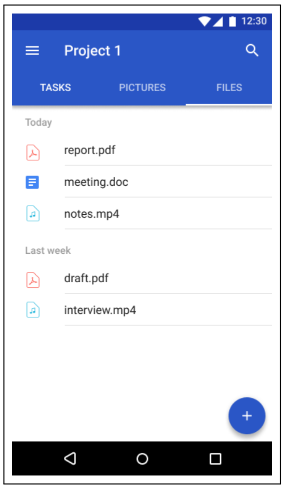
Project list



Project tasks



Project pictures



Project files