GitHub Username: marcotf-git

Learning App

Description

The app is for helping teachers and students in having a short class with video, text and useful links about the content. The teachers can use the Android device, intended to be a tablet, for making a mini video and a small text with some useful links for further reading. The students can load the content in their devices, phones or tablets, for viewing anytime.

The user can view and create the content. The app will have a local database storage (for viewing offline) and external database for uploading or downloading.

Intended User

This is an app for anyone interested in teaching short classes. The app can also be used in other areas, as its structure is of a general purpose knowledge sharing app.

Features

The app will have the main features:

- The app will help the user (teacher) in creating the class, by making the movie with the camera of the device, by making the text with the virtual keyboard, and storing the content in an accessible external database.
- The app will retrieve the data stored in the external database to the local database, when the user or student want to attend classes.

User Interface Mocks

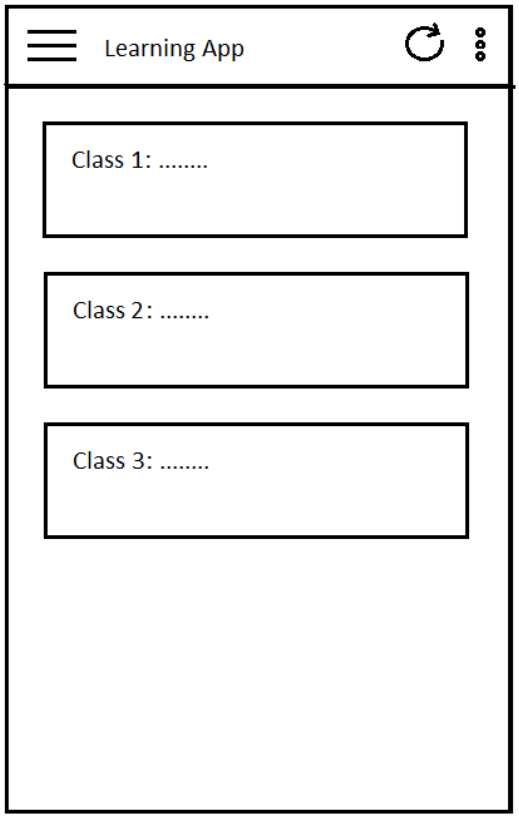

Fig. 1 - Screen 1

The main screen will show an app bar and some cards for the classes that are stored in the local database. The navigation menu icon will handle the login account. The settings icon will handle two main uses for the app: “create”, when the user can create the content, and “view”, when the user only views the classes. There will be a refresh icon, for downloading the latest content from the external database.

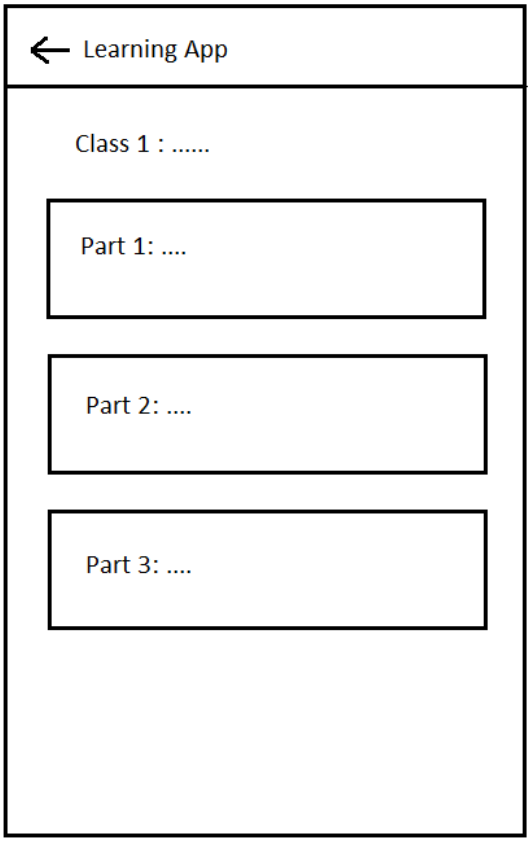

Fig. 2 - Screen 2

When the user clicks into one of the cards on the main screen (screen 1), another screen (screen 2) will open showing some cards with a short explanation of the content of each part of the class.

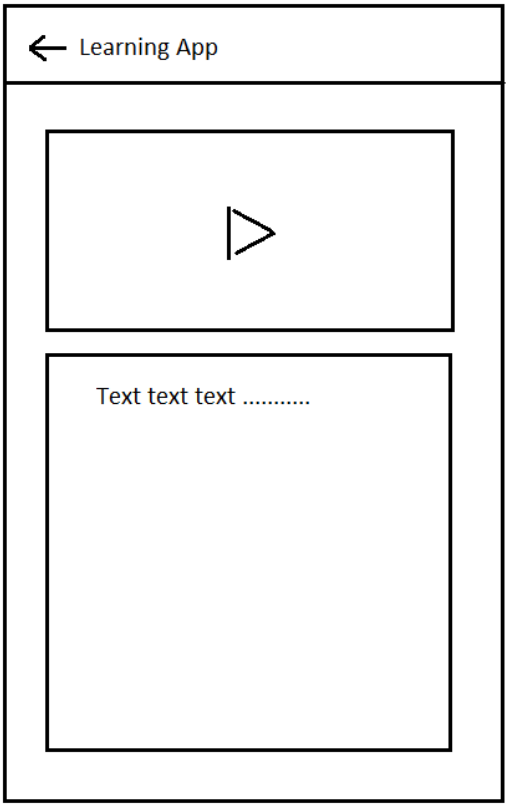

Fig. 3 - Screen 3

If the user clicks again into one of these cards, it will open another screen (screen 3: on the right) with the video or image, and the text and links.

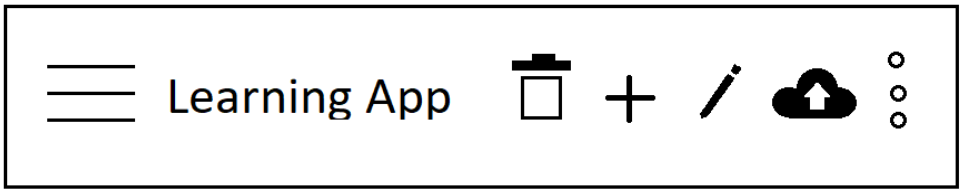

Fig. 4 - App bar

The app bar will be contextual. When the option to “create” is selected, the icons for adding, editing, deleting and uploading content will appear.

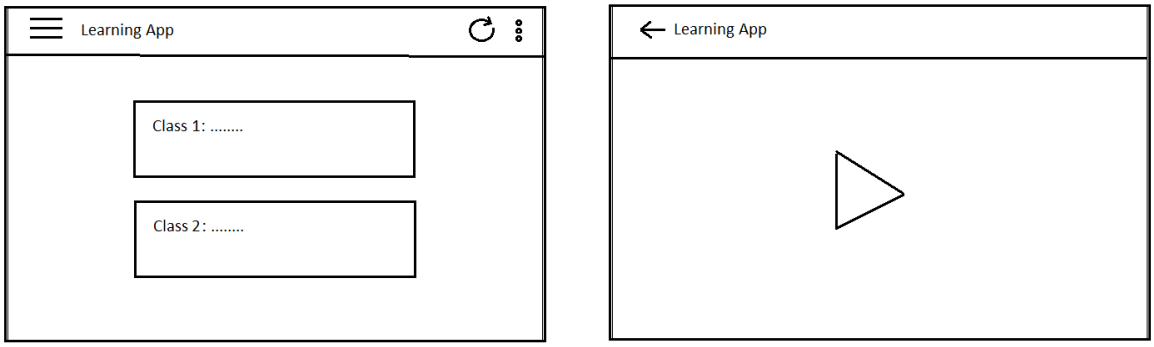

Fig. 5 - Phone in portrait mode

For a phone in portrait mode, it will use one column layout with a wide margin, in case of text (screens 1 and 2), and it will show only video on screen 3.

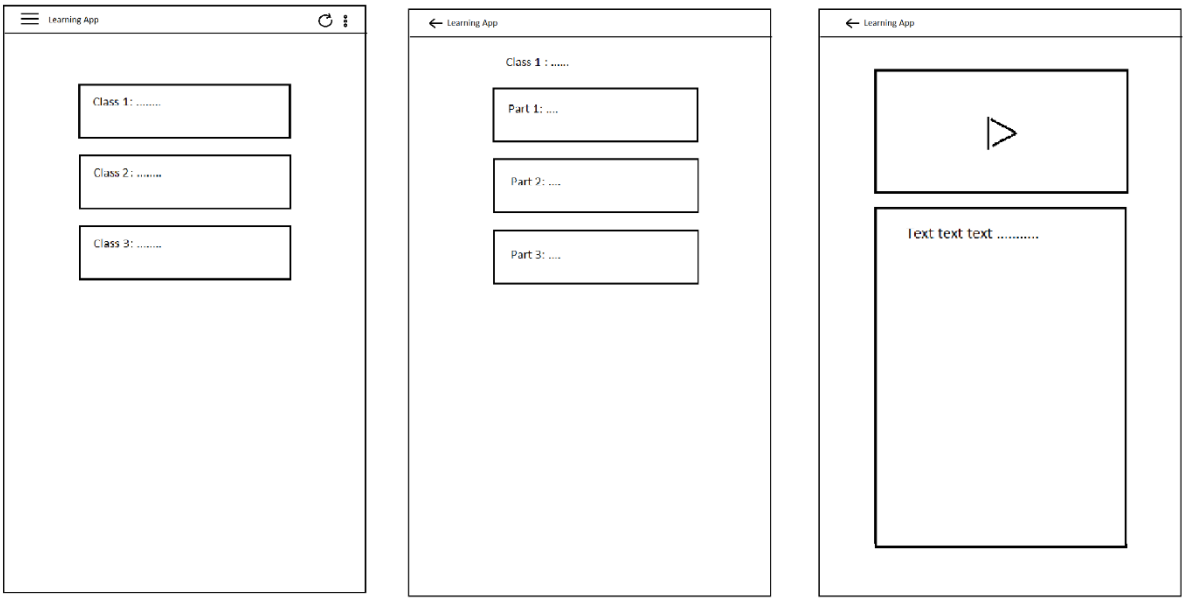

Fig. 6 - Tablet in portrait

For a tablet in portrait mode, it will have greater text sizes and wide margins and video view sizes.

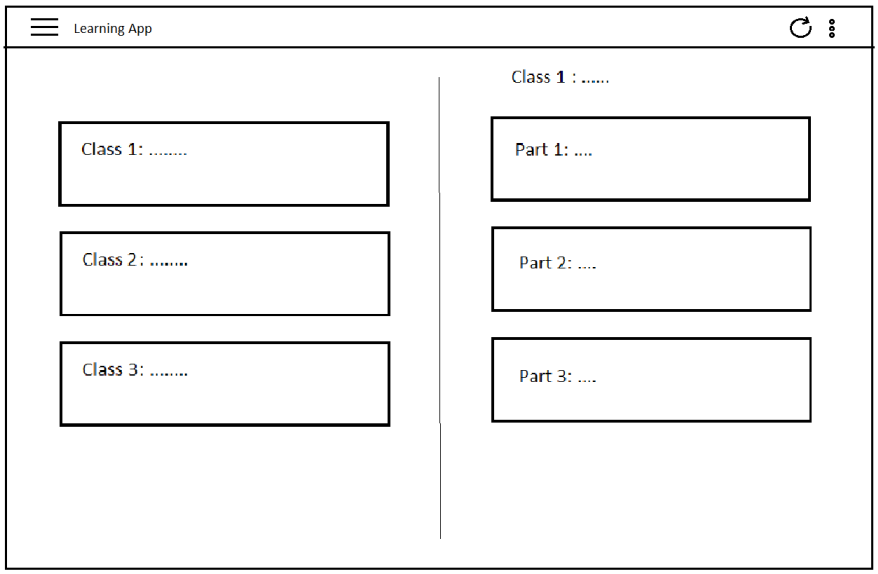

Fig. 7 - Tablet in landscape

For tablet on landscape mode, it will use two column layout, showing the class parts on the right, of the selected class on the left.

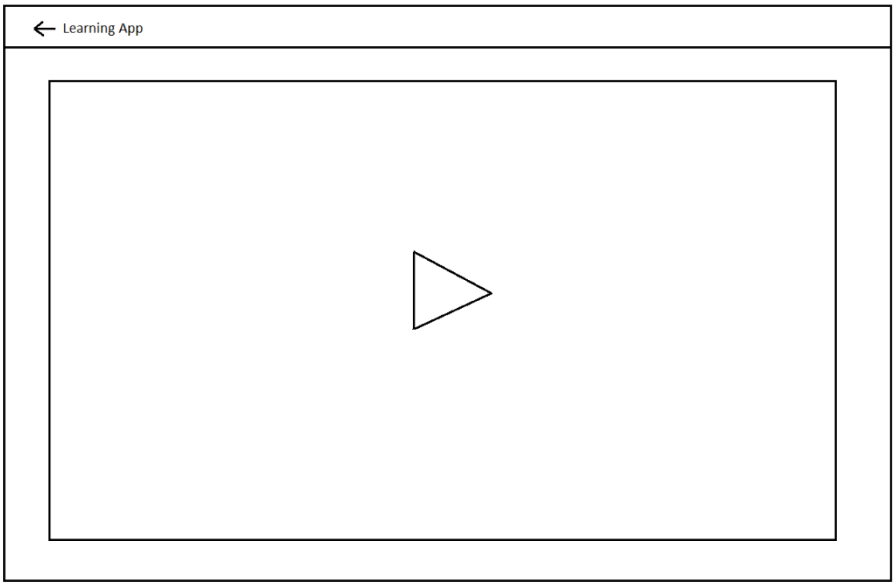

Fig. 8 - Tablet in landscape: video screen

For tablet on landscape mode, the video will fill the entire screen

Key Considerations

Data persistence

There will be a local database implemented with a Content Provider. There will be also an external database, on a remote server, for uploading and retrieving data.

When creating new data, the data will be inserted into the local database. After that, it will be possible to upload to the remote server.

App operation modes

The app will have two main settings: the “view” mode and the “create” mode, which will be selected on the overflow menu.

On “view” mode, the app will have three main screens, each one showing the data in a more specific way (fig. 1: Class Titles; fig. 2: Class Parts; fig. 3: Part Detail, with text and video or image).

On “create” mode, the app will have the same screens, but the app bar (fig. 4) will show the options in the form of action items to create new content, edit or delete, according to the view shown. Clicking on the “create” action item, it will open a text box in the same screen for creating that content.

Some useful libraries

The app will use **Picasso** (http://square.github.io/picasso/) to handle the loading and caching of images, and **ExoPlayer** (https://github.com/google/ExoPlayer) for playing the videos.

External services

The app will use the **Firebase** (https://firebase.google.com) for handling the external database and for the authentication and protecting the data.

Editing and deleting data

Only the user that has created the content will be able to edit or delete it.

Viewing data

All users will be able to view the data uploaded.

Required Tasks

Task 1: Project Setup

Add the dependencies (**AppCompat**, **Picasso**, **ExoPlayer**, etc.).

Setup the **Firebase** and add the libraries to the app:

- Create a **Firebase** project on the **Firebase** Console (https://console.firebase.google.com)
- In the options of the project, select “Add **Firebase** to your **Android** app” and follow the instructions (register the app; download the JSON configuration file in the app directory; add the SDK dependencies to the *build.gradle*)
- Configure **Firebase** Database and Auth dependencies (https://github.com/firebase/FirebaseUI-Android)

Task 2: Implement the simpler version

- Build UI for MainActivity and MainFragment, showing only one text box
- Build the account log in screen as part of the navigation drawer
- Build the setup with options “view” and “create”, as part of overflow menu

Task 3: Implement simple local database

- Implement a contract and a content provider, with only one field for testing

Task 4: Implement external database on Firebase

- Create Firebase account for the project
- Implement a simple one field database on Firebase
- Upload content to the database when user clicks the upload icon
- Download content when user clicks the refresh

Task 5: Test the simple version

- Simulate three users with their own data (create and view “classes” with the only one text phrase)
- See if the security rules are working (users can’t write on other user data)
- See what happens when the app is offline

Task 6: Extend the app: implement UI for each Activity and Fragment

Implement a complete version of the app, for mobile and tablet.

- Build UI for MainActivity and MainFragment (fig. 1, fig. 7)
- Build UI for DetailActivity and DetailFragment (fig. 2, fig. 7)
- Build UI for PartActivity (fig. 3, fig. 8)

Task 7: Extend the database

- Implement the contract and the content provider, with all the fields

Task 8: Extend the external database

- Implement all the fields in the external database

Task 9: Test the complete version

- Simulate three users with their own data (upload and download text and videos)
- Verify and rotate screens on mobile and tablet
- See if the security rules are working (users can't write on other users data)
- See what happens when the app is offline

Task 10: Verify Material Design

Verify if the app attend the Material Design specs.

- Try to make improvements to the layout, according to Material Design
- Verify the spaces between elements and the sizes of the views
- Check the colors
- Check the text font and visibility

Task 11: Test again

- Simulate three users with their own data (upload and download text and videos)
- Verify and rotate screens on mobile and tablet
- See if the security rules are working (users can't write on other user data)
- See what happens when the app is offline

Task 12: Make the instructions for installing

- Make the readme with instructions for installing and configuring the app
- Verify and correct any security issue with eventual app private keys
- Clean the project and upload