# Objective

Produce an Android application **from scratch** that fulfils a set of minimum requirements and a pre-approved number of elective specifications.

# Instructions

1. Enrol in a project group on eConestoga (Course Tools > Groups). You may have up to three members in the group. If you are looking for group partners, you may wish to post on the discussion board (Course Tools > Discussions > Find Group Members).
2. Think of a mobile application you want to eventually publish to an app store and be available for public consumption. Don’t go overboard (you can always improve on version 2.0 of your app after the course is over), but ensure that your idea:
   1. Is not similar to any prior examples (e.g. Tax Calculator, News Reader), assignments (e.g. Tic Tac Toe, Rock Paper Scissors) or projects in this course or from prior offerings of this course. *There is a repository of submissions that will be cross-checked to enforce academic integrity*.
   2. Fulfils all the minimum requirements in the proposal and can accommodate at least four of the elective specifications. If your application has a higher degree of difficulty, you might be asked to complete fewer than four electives. The number of electives you will need to complete will be sent via feedback on eConestoga.
3. Complete the proposal below and submit it to the appropriate eConestoga assignment drop box.
4. Wait for approval of the proposal to be provided in the feedback on eConestoga. You can start working on the project while waiting.
5. Work on the project.
6. Have at least 30% progress in time for the project checkpoint meeting (see eConestoga for your assigned meeting time). The checkpoint meeting provides an opportunity for one-on-one guidance and coaching for the project. It is not an evaluation of a project’s accuracy — simply attending with at least 30% progress will receive full credit.
7. Complete the project, meeting the minimum requirements and specified number of electives in the proposal.
8. Submit the project’s source code **and** signed APK to the appropriate eConestoga assignment drop box.
9. Prepare either a live, synchronous demonstration of your project or an asynchronous, video recorded demonstration of your project that clearly shows how each of the specifications (required and elective) is met by your application.
   1. If you choose synchronous, sign-up for a live demo (find the Sharepoint document under Content > Evaluations > Group Project) and demonstrate on or prior to Friday, December 10, 2021. Make sure step #8 (APK and code submission) is done first.
   2. If you choose asynchronous, record a video of your application in action, demonstrating how each specification from your proposal is met. You can use [Zoom](https://conestogac.zoom.us/) to create this recording (host a personal meeting room, share screen, and [record](https://support.zoom.us/hc/en-us/sections/200208179-Recording)). [OBS Studio](https://obsproject.com/) is another viable option for desktop recording with microphone enabled. Physical device recording with voice commentary is complex and not recommended. Submit your video in **mp4** format to the eConestoga assignment drop-box for the Group Project.

# Rubric

You can find the entire rubric for the Group Project under Course Tools > Rubrics.

# Project Proposal

Read and complete all highlighted sections. Save and submit this document to the eConestoga drop-box for Group Project Proposals.

## Group/Company Name

Your group name on eConestoga will be changed to this. Use this name when signing up for the checkpoint demo (and synchronous final demo if you choose that option).

Our group name is: Khachig Kerbabian (Individual)

## What is the name of your app?

You can change this later when you publish to an app store or repository. It should give a good idea what your app is about. (Examples: Recipe Manager, E-Z Point-of-sale, Connect Four Game)

Our app name is: Save Me money

## In a few sentences, briefly describe what your app does.

App description:

Spending and earning tracking system. It will help you to log in every spending and earning transaction with categorized options. Also, it will display charts and a list of transactions for previous recodes.

## Minimum requirements:

Your app **must** incorporate **all** these specifications:

The project is created from scratch in Android Studio (1.3)

The project is coded in Java and XML (1.1)

The application has a custom icon. The application also has a custom splash screen that clearly displays your group/company name or logo **and/or** your app’s name or icon (1.4, 1.9). You may wish to use a free SVG editor such as [Inkscape](https://www.microsoft.com/en-us/p/inkscape/9pd9bhglfc7h) and [Vecteezy](https://www.vecteezy.com/editor/random).

Applying concepts of User eXperience, (UX) from your prior courses, the shipped application appears polished and is user-friendly in both landscape and portrait mode. Usage of linear layouts and scroll views are up to you. Linear layouts provide an expedited way to get a neat layout in both orientations. However, you are free to explore other layouts for positional fine-tuning (2.1)

The user must be able to interact with the application’s main activity in a meaningful way, that is, multiple views must have event listeners (2.2)

If it is not clear from your app description and title how the user interacts with the main activity, please provide a one or two sentence explanation how you plan on including event listeners:

I used mostly Fragments.

Having bottom navigation to roll around all fragments.

Setting fragment.

Firebase Authentication (firebase has anonymous login which I used it for testing)

Firebase Database (real time)

Firebase messaging system for reminding the user.

User can’t log anything either to view before logging or create account.

Vibration settings

More than one element (view, activity background, etc.) adopts a different style than the default using styles **and/or** themes (2.3, 2.4)

The application retains its state when deactivating and reactivating the app or changing screen orientation (2.6.1)

The application has a menu, capable of accessing a Settings activity and at least one other activity (2.5)

A Settings activity and Preferences object make at least two configurations available to the user that have an impact on the app’s behaviour (2.6, 2.7)

The application has at least one other activity other than the Main and Settings activity that plays a meaningful role in the app’s behaviour (3.3, 3.4, 4.1, 4.2)

An SQLite database is meaningfully employed (4.1, 4.2)

If it is not clear from your app description and title how a database is used in your app, please provide a one or two sentence explanation how you plan on incorporating an SQLite database:

Using Firebase database. I also used interface for a call back method. For example when firebase server Is still sending data , I am calling interface to wait till all data has been received then I am creating an notification.

One or more custom animations is noticeable (2.9)

The application can be found on a public app store **or** a documented full attempt was made to submit a signed APK to an app store, i.e. it may still be under review, rejected just prior to project due date, etc. (5.1, 5.2)

The application achieves what is written for the app description

## Elective specifications:

Your app **must** at be able to accommodate at least four (4) of the following specifications. Check four or more boxes (and provide any supplementary information required) of the electives you wish to implement. Some electives may require additional research outside the scope of the course material if you wish to implement them.

App rearranges fragments for a multi-pane layout according to screen size and/or screen orientation (2.8)

App incorporates the snackbar **and** floating action button from the Material Design library (2.9 + additional research at <https://developer.android.com/topic/libraries/support-library/features?hl=en#material-design>)

App uses a drawer layout in lieu of the drop menu (2.9 + additional research at <https://developer.android.com/guide/navigation/navigation-ui>)

App has a custom control, that is, a reusable view/widget that appears in the Project section of the Palette in a layout editor, and is employed more than once in the application (2.10)

Describe in one or two sentences what this custom control is:

Array list Display using Adapter

App implements drag and drop (2.11)

App detects connectivity to the Internet and downloads relevant data from the Internet (3.2)

Describe in one or two sentences how you plan on involving remote data in the app’s functionality:

Using FireBase

App runs a meaningful service in the background and then sends a local device notification when the service has accomplished a task (3.5, 3.6)

Describe in one or two sentences what the service intends to do and when it notifies the user:

App listens for system broadcasts (e.g. low battery, power connected) or broadcasts from other apps (3.8, 3.9)

Describe in one or two sentences what broadcasts from the system or other apps your app attempts to receive and how it would react to those broadcasts:

App interfaces and receives data from a push notification web service (3.12, 4.3)

Describe in one or two sentences how the application interacts with a web service and the purpose the push notification serves:

FireBase messeging system

Something else outside of the scope of the course outline of equal complexity or greater than any of the previous electives. An example would be incorporating hardware like geolocation or camera (outside of course scope, additional research required)

Describe in one or two sentences what this feature is:

Using PhilJay/MPAndroidChart library to display Barchart and PieChart

# All done the proposal?

Save and submit this document to the eConestoga drop-box for the Group Proposal by **11:59 pm Friday, November 19, 2021.**

|  |  |
| --- | --- |
| Number of chosen electives required (Instructor use only) Based on the perceived difficulty of implementing this application and the size of this group, the number of electives chosen above that **must** be completed is: |  |