University of Westminster
Department of Computer Science

5COSC011C Coursework 2 (Semester 2)

Module leader	Torin Wirasingha
Unit	Coursework 2
Weighting:	50%
Qualifying mark	30%
Description	
Learning Outcomes Covered in this Assignment:	LO1, LO2, LO3, LO4, LO5
Handed Out:	13/3/2020
Due Date	08/4/2020 13:00
Expected deliverables	Source code/XML files/Resources (images, etc)
Method of Submission:	Online via Blackboard
Type of Feedback and Due Date:	Individual feedback verbally straight after the viva and written individual feedback via Blackboard within 2 weeks of submission **All marks will remain provisional until formally agreed by an Assessment Board.**

Assessment regulations

Refer to section 4 of the “How you study” guide for undergraduate students for a clarification of how you are assessed, penalties and late submissions, what constitutes plagiarism etc.

Penalty for Late Submission

If you submit your coursework late but within 24 hours or one working day of the specified deadline, 10 marks will be deducted from the final mark, as a penalty for late submission, except for work which obtains a mark in the range 40 – 49%, in which case the mark will be capped at the pass mark (40%). If you submit your coursework more than 24 hours or more than one working day after the specified deadline you will be given a mark of zero for the work in question unless a claim of Mitigating Circumstances has been submitted and accepted as valid.

It is recognised that on occasion, illness or a personal crisis can mean that you fail to submit a piece of work on time. In such cases you must inform the Campus Office in writing on a mitigating circumstances form, giving the reason for your late or non-submission. You must provide relevant documentary evidence with the form. This information will be reported to the relevant Assessment Board that will decide whether the mark of zero shall stand. For more detailed information regarding University Assessment Regulations, please refer to the following website:**http://www.westminster.ac.uk/study/current-students/resources/academic-regulations**

5COSC005W MOBILE APPLICATION DEVELOPMENT - Assignment 2

Deadline 08/4/2020, 13:00

Torin Wirasingha

Email:torin.w@iit.ac.lk

Description

You are required to implement an Android application described by the specifications below.

You are not allowed to use any third-party libraries with the only exception of the IBM Watson Cloud classes services mentioned in this document. The only other libraries that you can use are the standard Android API libraries found in the following URL:

`https://developer.android.com/reference/`

It is important to follow exactly the specifications and your implementation must conform to these:

The application developed will help the user learn and practice different foreign languages.

1. When the application starts, it presents the user with 5 buttons labelled *Add Phrases*, *Display Phrases*, *Edit Phrases*, *Language Subscription*, and *Translate.*

   (3 marks)

2. Clicking on the *Add Phrases* button, it should display to the user a page where the user can create and save English words and phrases. The page should contain 1 textbox which can be used to enter a single word (or a single phrase). For example, the user can enter the following: “Good morning” or “university”.

   The page should also contain a *Save* button which when pressed saves the word (phrase) entered in an SQLite database local to the mobile device.

   (12 marks)

3. Clicking on the *Display Phrases* button displays to the user a list of all the saved words and phrases sorted in alphabetical order (one per line) The displayed words and phrases should come from the database and they should correspond to the ones which the user created using the *Add Phrases* functionality in this instance of running the application or a previous instance that the application was run (i.e. the application should NOT delete any saved words/phrases when it exits and all data are persisted in the database).

If the list of words/phrases is more than one page long, it should provide scrollbars that the user can use to see all of them.

(10 marks)

4. Pressing the *Edit Phrases* button the application should display to the user a list of all previously saved words and phrases (one per line). Next to each of them there is a radio button which the user can use to select only one word (or phrase), i.e. only one selection can be active at any point in time.

   A button titled *Edit* and a button titled *Save* are also present in the page together with a textbox which is initially empty. The user can select a single word (phrase) by ticking one of the radio buttons and subsequently pressing the *Edit* button. Once the *Edit* button is pressed, the selected word appears in the textbox and the user can use the textbox to change (edit) the contents of the word/phrase. If a different radio button is selected before pressing the *Save* button, the contents of the textbox are updated with the newly selected word (phrase). Once the user finishes editing the contents of the textbox, he/she can press the *Save* button and the newly entered information will replace the selected word/phrase in the database. At the same time the current page will be updated (refresh) with the new information available in the database.

   (15 marks)

5. Clicking on the *Language Subscription* button the list of all available foreign languages found in the following web page will appear:

   https://cloud.ibm.com/docs/language-translator?topic=language-translator-identifiable-languages

   A tickbox next to each foreign language is also displayed (the tickboxes should initially be blank, i.e. not ticked the first time that this application is ever run). The displayed languages should come from the database and each tickbox should only be checked if the the user has previously subscribed to the corresponding foreign language using the *Language Subscription* functionality.

   The user can check or uncheck the tickboxes and then click on a button titled *Update* which is also present in the page in order to subscribe or unsubscribe to the corresponding foreign language respectively. Once the button is clicked, the application updates the database with the list of foreign languages subscribed by the user.

   (15 marks)

6. Pressing the *Translate* button the user is presented with the list of words/phrases saved before in the database together with a dropdown menu and a `Translate` button. The dropdown menu contains the list of foreign languages that the user has previously subscribed to. After the user selects a foreign language he/she can click on a word/phrase to select it (this could be possibly indicated with a different background/font) and subsequently click on the *Translate* button. The translated word/phrase should appear somewhere in the page.

   The translation should be made by using the IBM Cloud Watson Translator Web service as covered in the tutorials (a free account should be created first to obtain a unique API key and a unique URL for the service):

   https://cloud.ibm.com/docs/services/language-translator?topic=language-translator-gettingstarted

   (13 marks)

7. Extend the previous (Translate) task by adding a *Pronounce* button in the page. Once a translation is made, the *Pronounce* button can be clicked and the application should speak and pronounce the translated foreign word (phrase) by using the IBM Cloud Watson TextToSpeech Web service covered in the tutorials:

   https://cloud.ibm.com/docs/services/text-to-speech

   (12 marks)

8. *Optional:* Extend the application so that the user can translate all the saved English phrases in a foreign language of his/her choice and save the translations in the database. Following that, the user should have an option where he/she can see all the phrases and the corresponding translations, e.g. selecting "French" displays all the saved phrases and the corresponding saved translations in French (the translations should come from the database not the Web Service so that the user can use this functionality while the device is offline). The user should be able to do the same for multiple foreign languages, i.e. the application should be able to save the words/phrases stored in the database in more than one foreign languages.

   This option will be useful for the user to revise all of the phrases learned in a specific foreign language. *(10 marks)*

Marking Scheme: The marks achieved for each part of the program are indicated in the description of the task above. In addition to these the following will be taken into account:

- *Code readability* (structure, comments, variable naming, etc,): 5%
- *Implementation* (e.g. quality, efficiency, look and feel of the application, based on fonts, colours, etc.): 5%

The maximum for work which does not compile (or XML files with syntax errors causing the Java code not to compile) is 30%.

Based on the functionality implemented, the marks awarded will consist of 2 parts:

- 30% of the marks achieved will be awarded based on the submission.
- The remaining 70% of the marks for the implementation will be awarded after a compulsory viva, that will test the understanding of the code by the student. The student will be asked to demonstrate the application and will be asked questions about the code to demonstrate his/her understanding. No understanding of any parts of the code will result in no marks given for this component (i.e. a maximum of 30% can only be achieved if a students cannot explain his/her code and he is unable to point to any references within the code of where this code was found in a textbook or internet).

A compulsory viva for each student based on his/her submission will take place during the next tutorial session after the submission (all of them taking place in the weeks starting the 30th of March and 6th of April). Each student will be notified a specific slot that he/she needs to attend. Failure to turn up in the viva slot designated (no changes will be allowed as this a normal examination) will result in awarding only 30% of the marks achieved for the submission (see marking scheme above)

It is the responsibility of each student to make sure that during the viva the code runs properly in the lab used during the viva, i.e. you should make sure in advance (allow enough time before the viva day) that everything is running properly in the machine you will be using. If you developed code at your home computer, it is your responsibility that you port it to the lab in advance, before the viva. Marks will be awarded based on the demo/viva and excuses of the type "it used to run - don't know what happened since last time" will not be accepted or awarded with extra marks.

You are allowed to use your own laptop during the viva if you wish to.

Submission of assignments using a different method other than Blackboard will not be accepted and zero (0) marks will be awarded in such cases.

Deadline: Wednesday 08th of April 2020, 13:00.

Submission Instructions

Files to submit: All of the files of the Android Studio project of your application in a zip file. *Referencing code:* Any code taken from other resources (i.e. a textbook or internet) should be referenced in comments within your code (full textbook details or full web URL), identifying the exact code that you used it as part of your application and the exact portions of the original source code that you reused.

You should submit via BlackBoard's Assignment functionality (do NOT use email, as email submissions will be ignored.), all the files described above. A single zip file with the name `wNNNNNNNN` (where `wNNNNNNN` is your university ID login name) containing all the above files could be submitted alternatively. You can create such a file by using the main menu in Android Studio and choose `File->Export to Zip File...`

Note that Blackboard will allow to make a submission multiple times. Make sure before submitting (i.e. before pressing the `Submit` button), that all the files you want to submit are contained there (or in the zip file you submit).

In the case of more than one submissions, only your last submission before the deadline given to you will be marked, so make sure that all the files are included in the last submission attempt and the last attempt is before the coursework deadline.

Request to mark submissions which are earlier than the last submission before the given deadline will be ignored as it is your responsibility to make sure everything is included in your last submission.

The following describes how to submit your work via BlackBoard:

1. Access `https://learning.westminster.ac.uk` and login using your username and password (if either of those is not known to you, ask the HelpDesk at the Library.).

2. Click on the module's name, `MODULE: 5COSC005W.2019 MOBILE APPLICATION DEVELOPMENT` found under `My Modules & Courses`.

3. Click on the `Assignments` button found on the left hand side menu.

4. Click on `View/Complete Assignment`.

5. Attach your zip file containing all your files of your Android project, by using the `Browse` button.

6. Fill in the requested information:

   - *Comments*: Type your full name and your registration number, followed by: ”I confirm that I understand what plagiarism is and have read and understood the section on Assessment Offences in the Essential Information for Students. The work that I have submitted is entirely my own. Any work from other authors is duly referenced and acknowledged.”

7. Click the `Submit` button.

If Blackboard is unavailable before the deadline you must email com_submission@iit.ac.lk with **cc:** to myself and your personal tutor before the deadline with a copy of the assignment, following the naming, title and comments conventions as given above and stating the time that you tried to access Blackboard. You are still expected to submit your assignment via Blackboard. Please keep checking Blackboard's availability at regular intervals up to and after the deadline for submission. You must submit your coursework through Blackboard as soon as you can after Blackboard becomes available again even if you have also emailed the coursework to the above recipients.

Coursework Marking scheme

The Coursework will be marked based on the following marking criteria:

Criteria	Mark per component	Mark provided	Comments
Implementation	**100**		
Functionality	90		For a split of the marks see the subquestions description in the main description of the coursework
Code Readability	5		structure, comments, variable naming, etc,
Software Quality	5		Quality, efficiency, etc.
Total	**100**		