**Project 6**
**Semester Project – Final submission**

- Name of project: *Happy Learning*
- Team members: *Ji Zhao, Yiou Gao, Ling Liu*

## 1. Final State of System Statement

1.1 Language creation system:
The final language creation system is the same as we proposed: Users can specify the sounds for which they want to generate combined audio and specify rules about the order in which these sounds should be combined. The sounds should be specified with IPA symbols, and the rules can be specified either as sequences of IPA symbols or sound types: V for vowels, C for pulmonic consonants, CLICK for click consonants, EJECTIVE for ejective consonants, and IMPLOSIVE for implosive consonants. The system can throw out reminders if the sound provided is not part of the phonemic system of the language.

1.2 Game system:
We have already done with sign-up, log-in, learning, and quiz system. Our application supports two languages, which are English and Chinese.
In our application, user's information is saved in the cloud database - firebase.

1.2.1 In log-in system, we matched the user's username and password. If username not existed or username and password do not match, the user need to correct their input.

1.2.2 In sign-up system, user can register with username, email, and password. If username already existed in our record, the system will notify the user that the username should be changed.

1.2.3 In the learning system, the main page for user to has several choices for users. User can choose to go to learning page or quiz page. In learning page, there are two lessons are provided: Lesson One and Lesson Two. User can choose one of them to learn. In Learning Page, each character was displayed by image view. In addition, we used media player to implement playing sound for each character. By clicking the volume image, which is an image button, pronunciation can be played. The character and pronunciation were store in an array adapter and displayed through list view.

1.2.4 In the quiz system, user will listen to an audio clip then choose the correct answer. There are two options for users. If they choose the correct one, the button will turn green, else the button will turn red.

We didn't implement storing user's game level in our record. Originally, we tried to use aws database to implement log-in system, however, there are a lot of materials need to read and learn, and the example codes are very limited, finally, we decided to use firebase instead. Firebase could be implemented directly by Android Studio. It is a good choice for developers for want to user cloud database and android studio to develop apps.

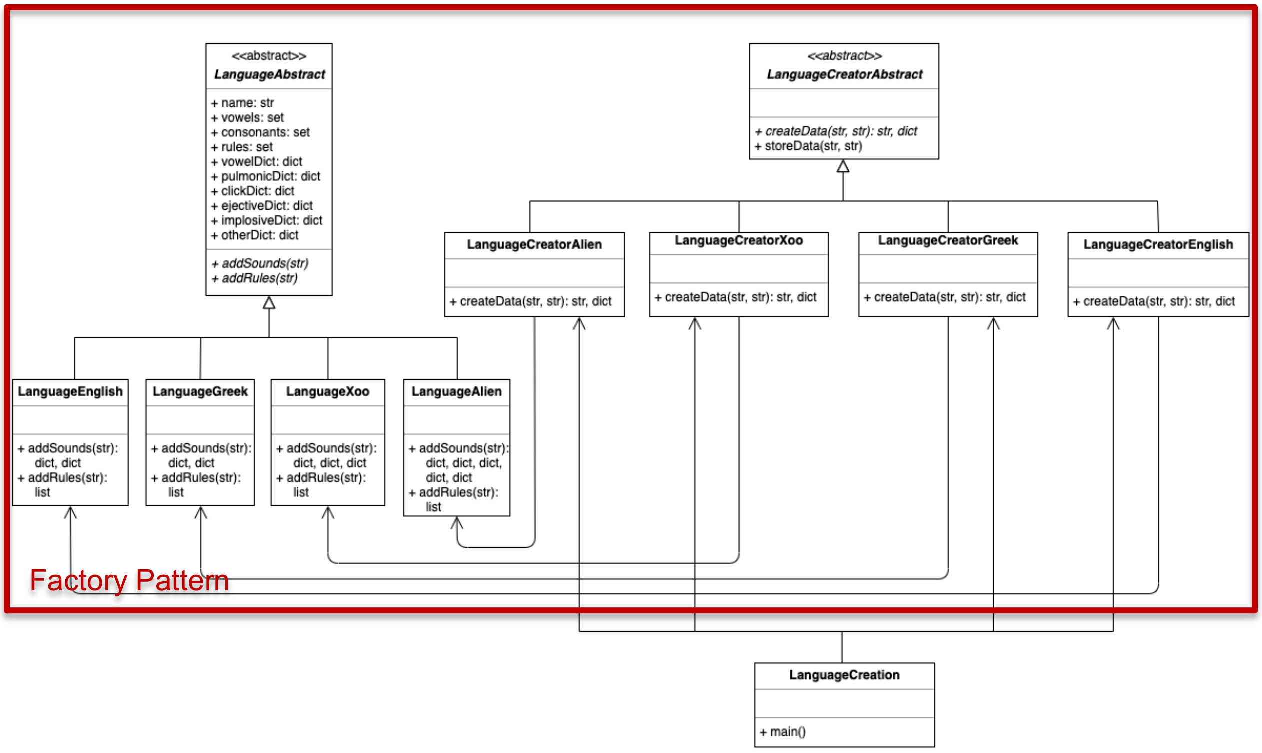
## 2. Final Class Diagram and Comparison Statement

2.1 Language creation system:

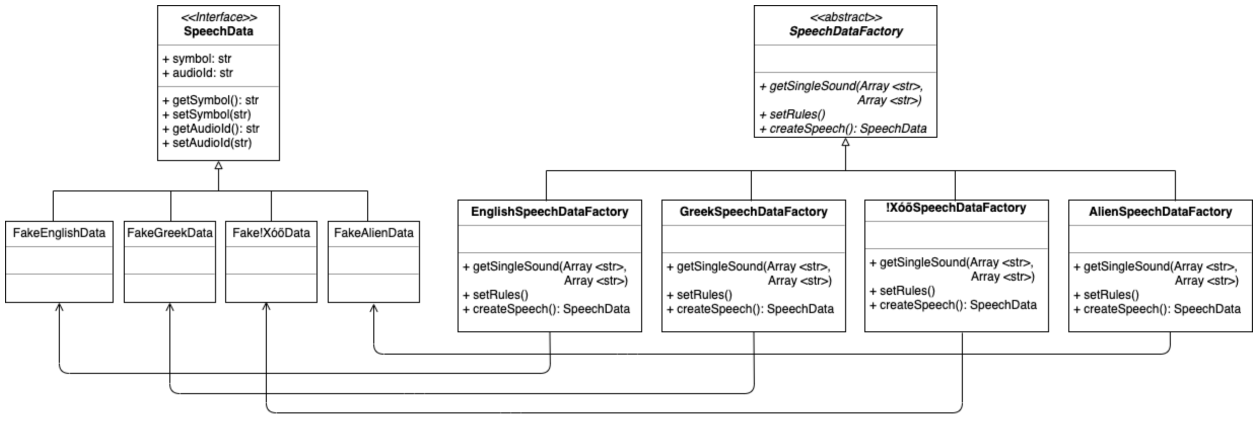
The Factory design pattern is used for this part of the system. The structure of the class diagram is generally the same as the one we submitted in project 4.

For the final class diagram, we added the class **LanguageCreation**, which is the main class to call the concrete language creation classes to create audio data. This is the main difference in the final class diagram and the class diagram we submitted in project 4. Other than that, we renamed the classes to increase readability. For the abstract product in the Factory pattern, i.e. the *LanguageAbstract* class, we treated it as an abstract class, rather than an interface in project 4. We also changed the attributes and methods in the classes in the final class diagram.

Final Class Diagram: The part in red rectangle used **Factory Pattern**.

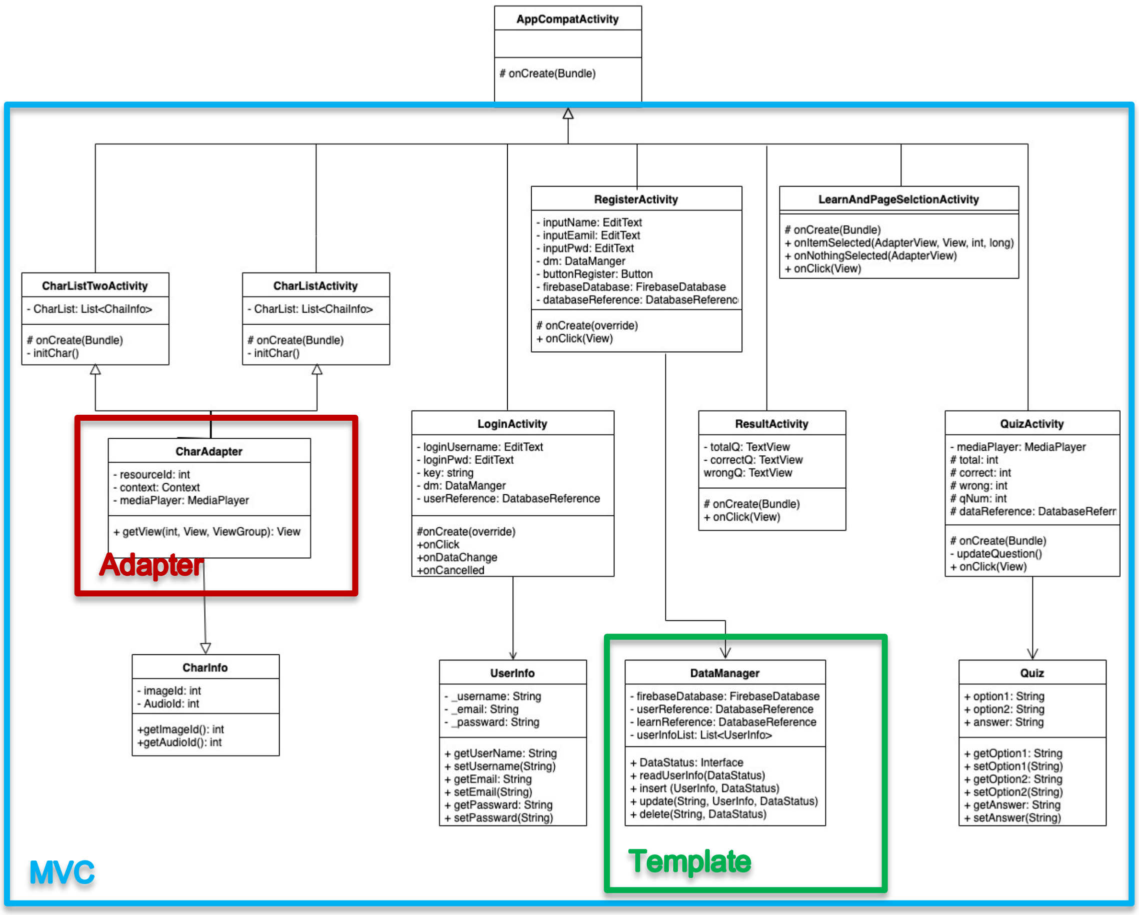


Class Diagram in Project 4: Factory pattern



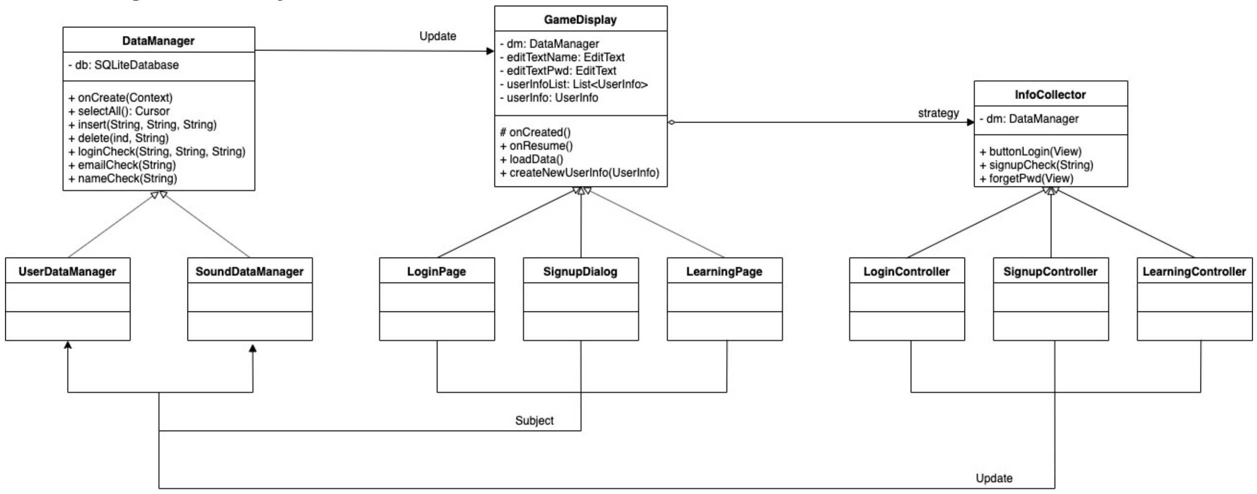
2.2 Game system:

The class diagram in our final project used **MVC**, **Adapter**, and **Template Pattern**. The **whole project** implemented MVC, **CharAdapter** implemented Adapter Pattern, and **DataManager** implemented Template Pattern, as highlighted in Figure below.



Compared with patterns used in Project 4, the MVC pattern is almost the same. The key difference is that we used Adapter and Template pattern in our final version. For Adapter pattern, it was used in list view our content in screen. The Template adapter was used in DataManager to track the data changes. We did not apply these two patterns before because we did not expected to track data changes or use list view to display our data.

Class Diagram in Project4:



## 3. Third-Party code vs. Original code Statement

3.1 Language creation system:
3.1.1 The code for this part is original.
Pydub (https://github.com/jiaaro/pydub) is used to read, concatenate and store audio files.

3.1.2 The data we started with are audio files of vowel and consonant combinations.
Praat (http://www.fon.hum.uva.nl/praat/), a free software package for speech analysis in phonetics, is used to manual cut out the audio file for single sounds.

3.2 Game system: We used the following websites to learn about how to develop the game system.
3.2.1 Using media player to implement playing sound.
https://developer.android.com/reference/android/media/MediaPlayer
3.2.2 list view
https://developer.android.com/reference/android/widget/ListView
3.2.3 spinner
https://www.youtube.com/watch?v=on_OrrX7Nw4
3.2.4 quiz system example
https://www.youtube.com/playlist?list=PLF0BIlN2vd8tJjVKdNegtQF2j9CjmGOc_

## 4. Statement on **the OOAD process** for your overall Semester Project

4.1. We found that the design process we went through for project 4, i.e. start with thinking about (which should be communicating with customers in the real world about) and writing down the requirements and use cases as well as alternative paths to fulfill them for the system, then draw class diagrams, activity diagrams, sequence diagrams etc. We also found that the architecture diagram, though not part of the UML, is very helpful to picture the high-level big picture of the system.

4.2. We found that the design process is iterative. Though we have laid out our system in project 4, we have to come back to it and think about and redesign the system in the process of implementation to make our system more readable, testable, and adaptable to change. For example, we changed the details of the classes for the language creation part a lot, though the Factory design pattern is still used as we proposed before. In addition, we switched to Firebase from our proposed AWS, because we found that it's easier to pick up Firebase for Android game development purpose.

4.3. We found that using design patterns can provide a good guide to the design and implementation process. It's a good way to make the code response to change, safe from bugs, and easy to understand. Thinking with the design patterns in mind can help us structure the implementation better, which is very helpful when we get stuck in the implementation details.

4.4. Testing is an indispensable and critical part of the development process. We have been testing our system while implementing it. Every time we add a feature to our system or implement some testable part of it, we test it before moving on. This helps us to keep the system safe from bugs and ensure that we have implemented the features we need.

4.5. There are a lot of details involved in the object-oriented analysis and design process. We started with highlighting coarse-grained elements and relationships and tried to consider it as detailed and thoroughly as possible, but there are still more details emerge as the project goes on. We also found problems which we did not expect.

4.6. Though the analysis and design process are critical for software engineering and development, a good software system requires more than that. One aspect of the more part is the quality of the data. For example, our language generation system can create audio data as proposed without problem, but the audio created are not natural because the data we have is limited. Generating natural speech audio by concatenating shorter audio clips requires more fine-annotated data, which is beyond the scope of this project.

4.7. Designing user interface requires a lot of time and work. This process took us a lot more time than we expected.