Bir Başka Yol
Mobile Application Project

Emre Reyhanlıoğlu – 150140126

Graduation Project Presentation

Contents

What is Bir Başka Yol?

- Bir Başka Yol is a social responsibility sub-project of ITU Volunteering Club.
- In this project, volunteers give lectures for more than 120 high school students, who are not able to get after-school support, at weekends.
- Aims of the project are preparing students for their university exam and decreasing inequality of opportunity between students.

Goals of the Mobile Application Project

- This mobile application will connect all of the members of Bir Başka Yol project in one platform. There are two user type in the project: Volunteers and students.
- Volunteers will be able to;

  - Share student's exam results and announcements

  - Connect with a student as an advisor, set weekly goals and watch student’s progress

  - Fill their one-to-one lecture profile with their expertises and available days
- Students will be able to;

  - See their exam results, announcements, leaderboard and their gamification profiles

  - Update and observe their progress of weekly goals from their profile page

  - Buy real items from the shop
- Both students and volunteers will be able to;

  - Communicate with each other via class message boards.

  - See the remaining time to the university exams.

  - Use the handwriting calculator

Technologies

- Project is developed with Flutter which is a software development kit(SDK) for both Android and iOS devices.
- Programming language that used is mostly(more than 15.000 lines of code) Dart, but Python is used to develop CNN model and JavaScript is used for cloud functions.
- Firebase Services are used for backend of the app.

  - Cloud Firestore Database which is NoSQL database is used to store all the data.

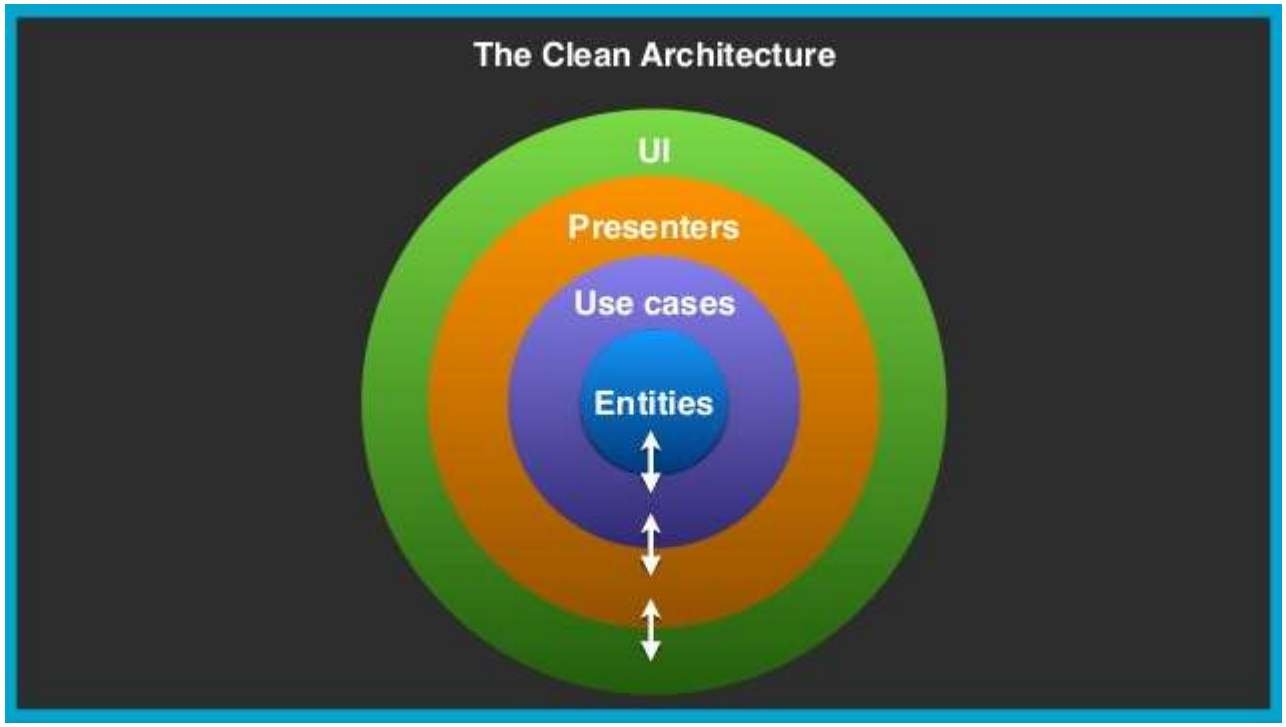
  - Storage is used to store images in the app and it generates access tokens which are stored in the database after generation.

  - Authentication service is used for signin & signup operations

  - Cloud Function service is used for server side operations such as resetting the weekly leaderboard and one-to-one lecture matching algorithm

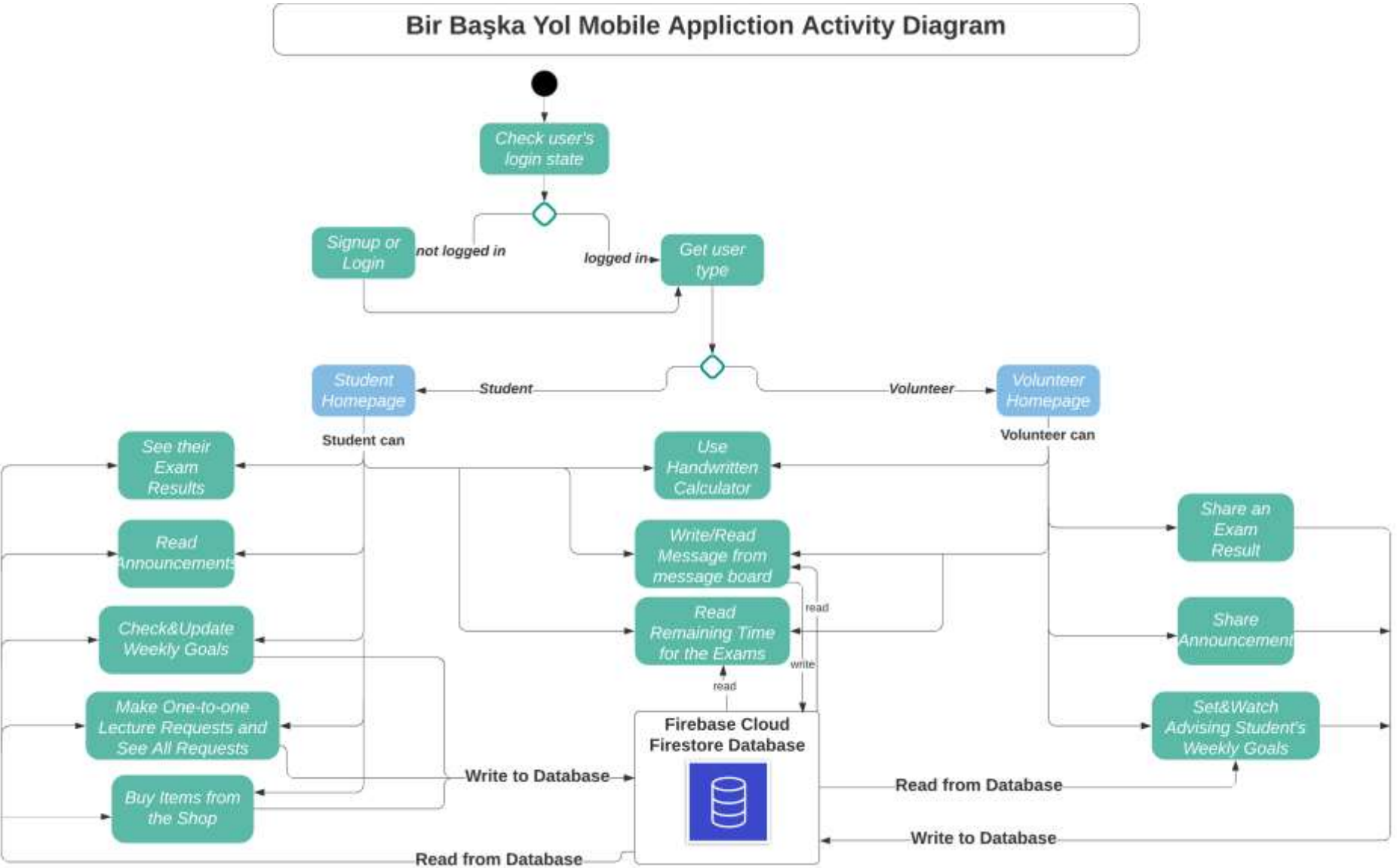
Architectural Design

- Most of the mobile applications have many different pages so that project architecture is very important for flexibility and maintainability of the project.
- Clean architecture is used in all of the modules in the project with SOLID principles



Source: https://www.slideshare.net/ssuser8fd0ea/mobile-architectures

Activity Diagram of the Project



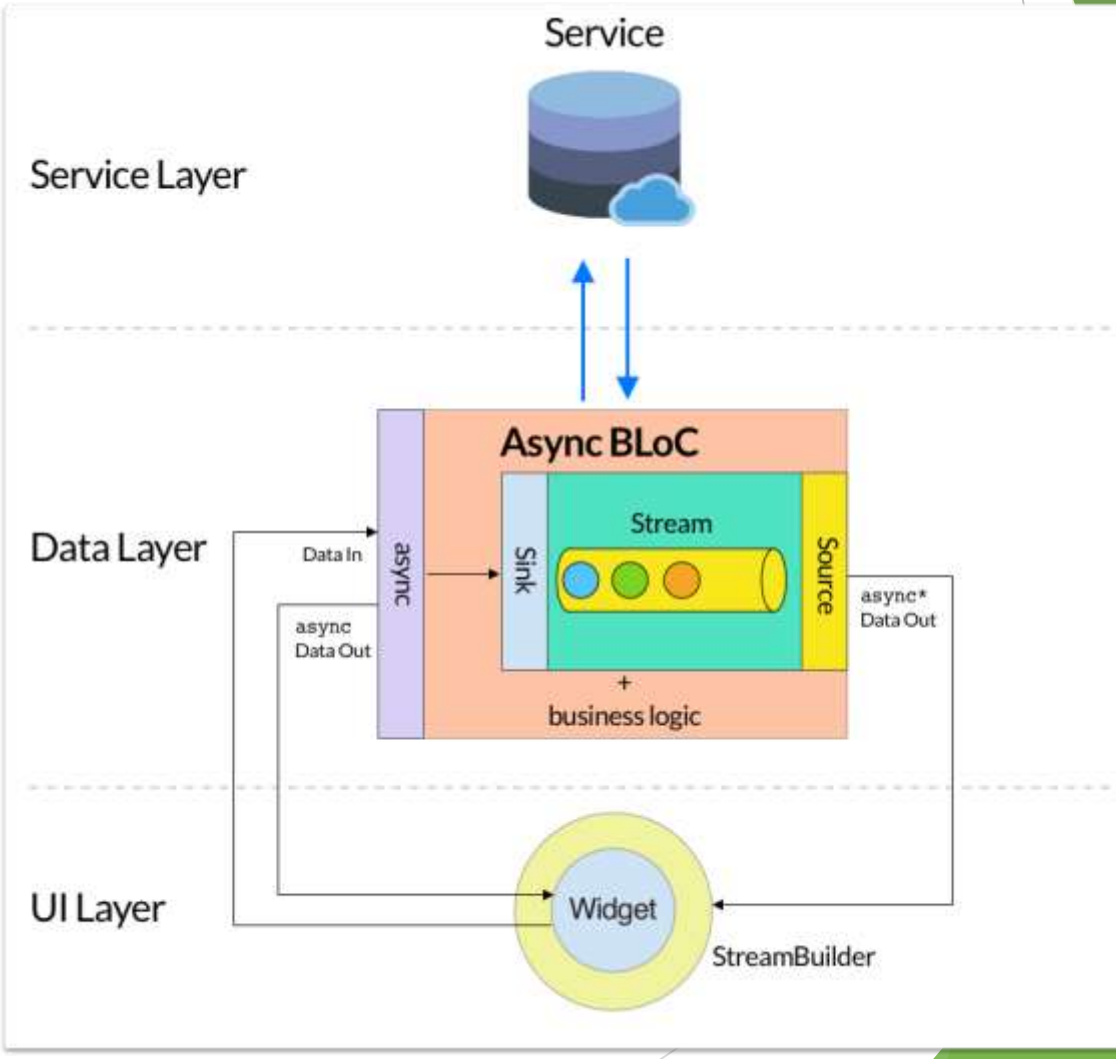
Design Patterns

- Design patterns provide general solutions for similar problems and they are reusuable.
- Some of the common design patterns which I used in the project are;
  - BLoC
  - Adapter with Observer
  - Singleton
  - Facade

Design Patterns

BLoC(Business Logic Component)

- State management pattern
- Created by Google and announced at Google I/O 18
- It handles state changes in the app by using reactive programming concepts such as sinks and steams
- Middleman between service and UI layers
- Example module in the project: Login & Signup



Source: https://codewithandrea.com/articles/2019-05-21-wabs-practical-architecture-flutter-apps/

Design Patterns

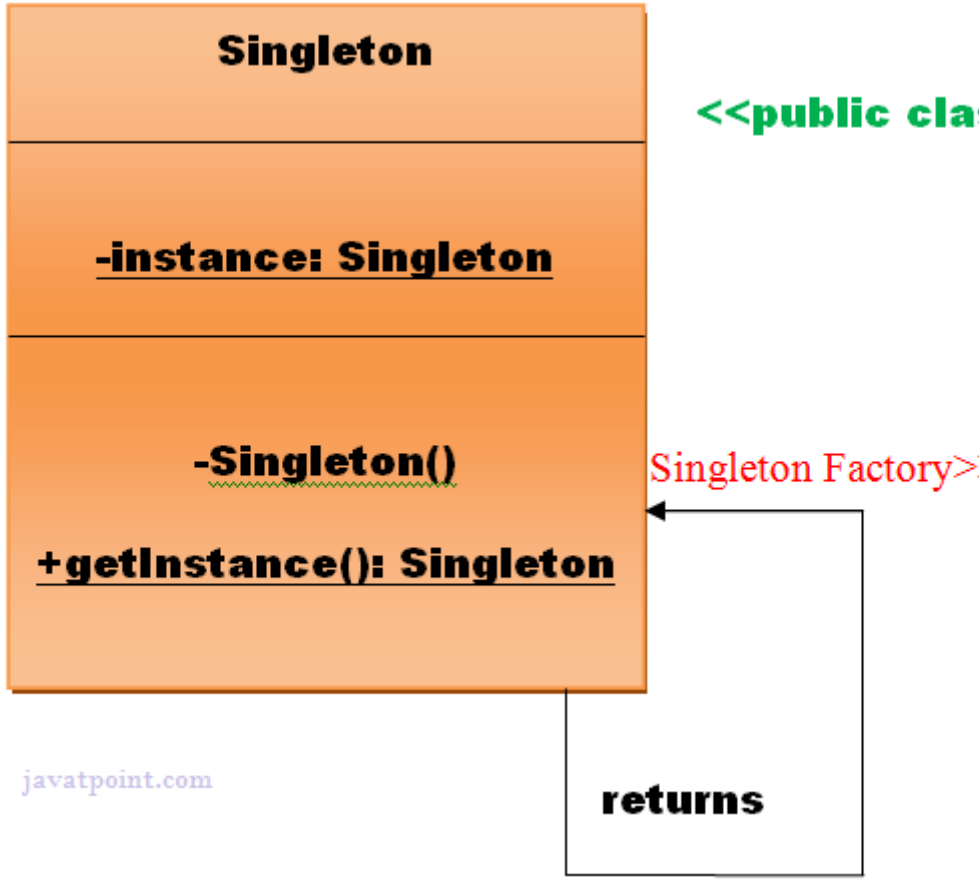
Adapter with Observer

- Adapters and observers are used together in mobile projects.
- Observer pattern retreives the list of data as a live data in the project.
- Adapter gets the list of data and populate the items by using view holder widget.
- List of data is presented by using proper widgets nicely.
- Some example modules in the project are message board, announcements, attendances and exam results modules.

Design Patterns

Singleton

- Singleton is one of the simplest and most common pattern in design patterns
- It makes sure that only one instance from a class exists in whole application
- Example usage of the project is database class, shared preferences class which is used to store data locally on the device



Source: https://www.javatpoint.com/singlet
design-pattern-in-java

Design Patterns

Facade

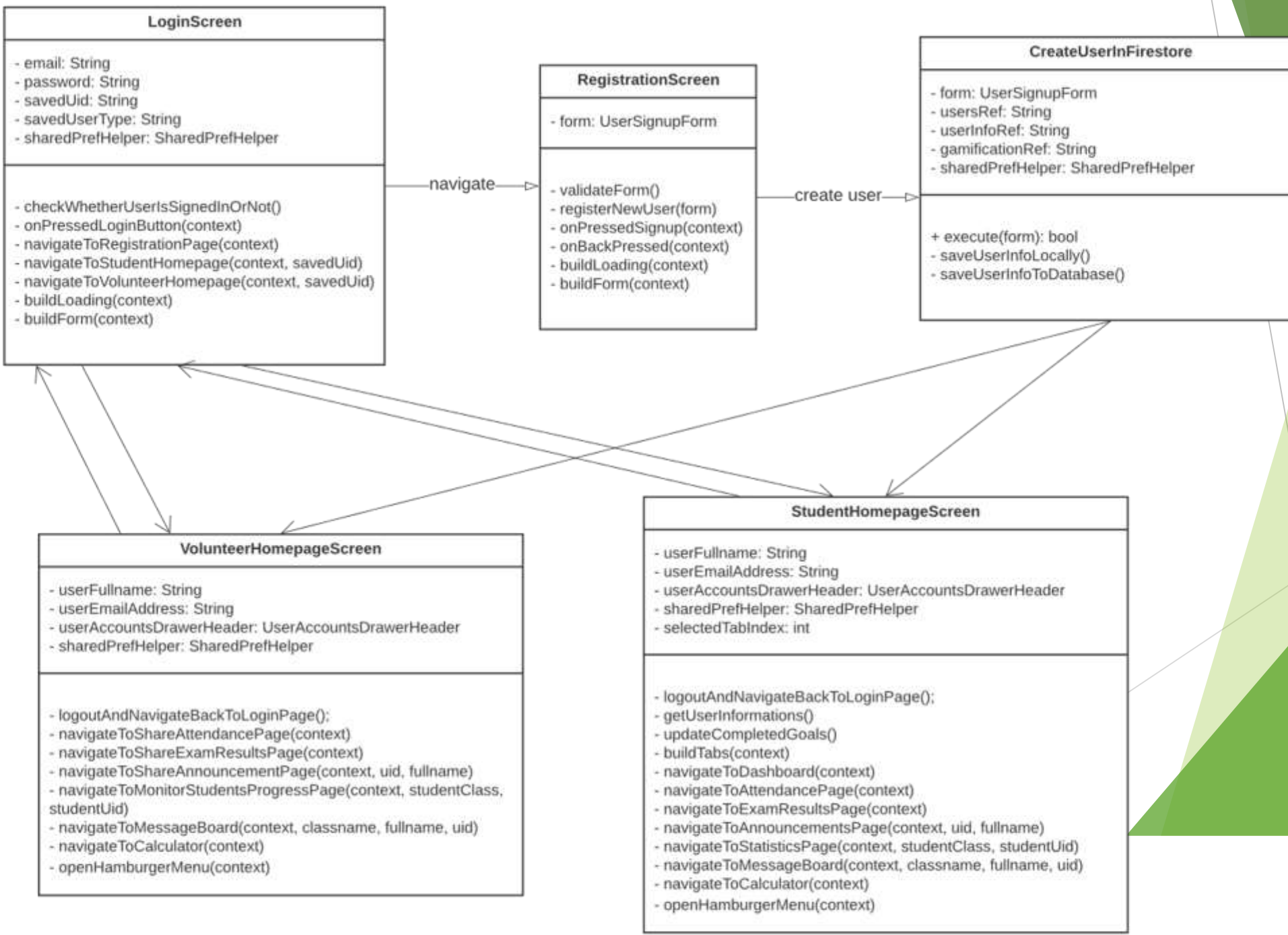
- Facade design pattern simplifies the complex systems by masking the complexity with a simple interface
- Mobile applications generally have simple user interfaces but there are many operations on the backend
- This pattern is used at almost all of the pages in the application.

Some of the Modules in the App

Login & Signup Module

- Regular expression is used to validate e-mail address in the input field
- Initial, loading, signin success and signin failure states are handled in signin BLoC
- If signin is successful, then user is navigated to Student Homepage or Volunteer Homepage based on user type
- Some of the frequenly used user data is stored locally while user is signing in

Login and Register Module Simplified UML Diagram

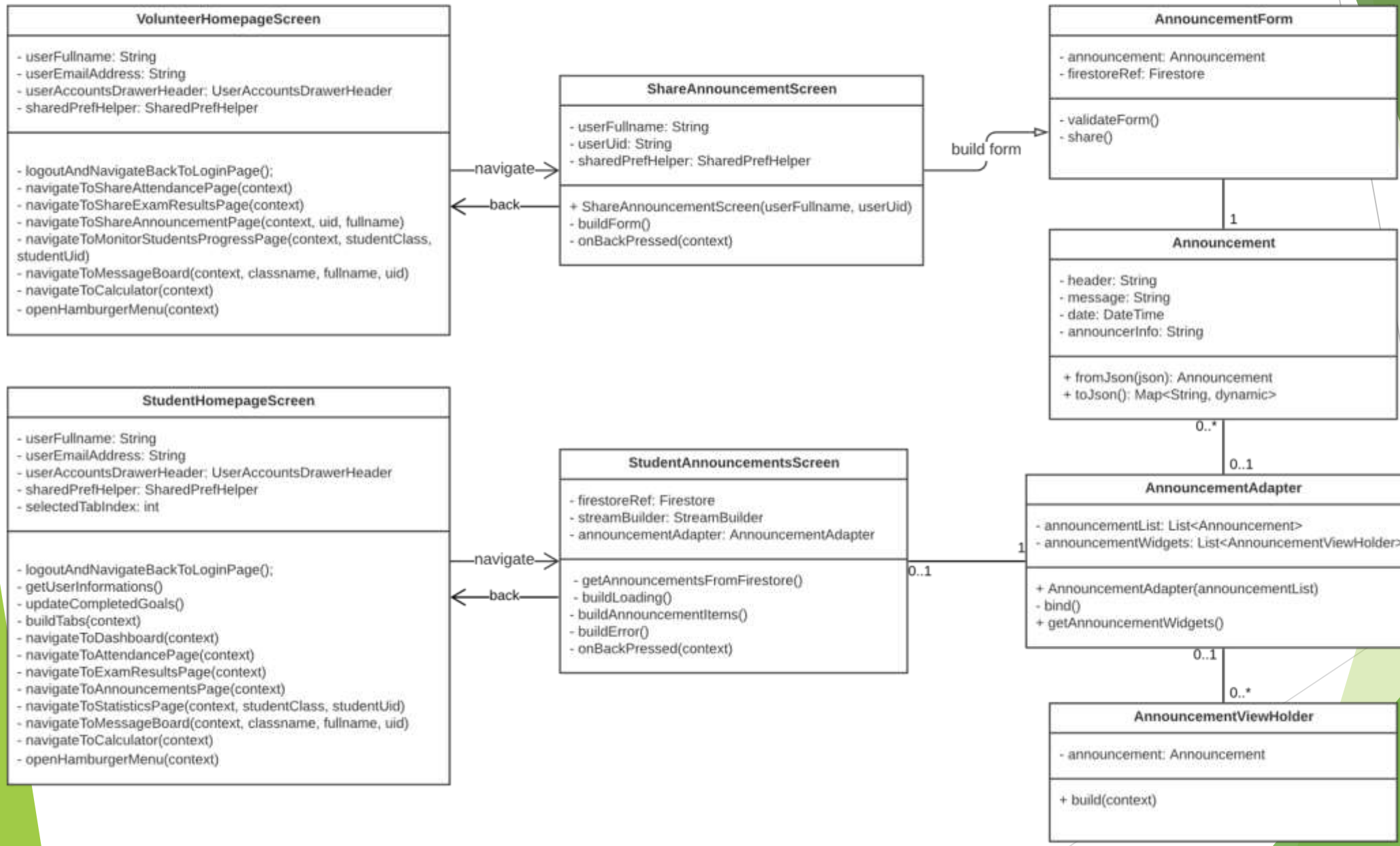


Some of the Modules in the App

Announcement & Message Board Modules

- Steam builders are used to present live data from database(any update from the database rebuilds the UI)
- User interfaces are designed based on loading, error, empty data and success states
- Announcements and messages are stored into database with their published date as a document name so that it is already sorted. (Different time zones have no effect)
- Speech-to-text functionality is added into message board to write messages with voice as an alternative way of writing with keyboard

Announcement Module Simplified UML Diagram



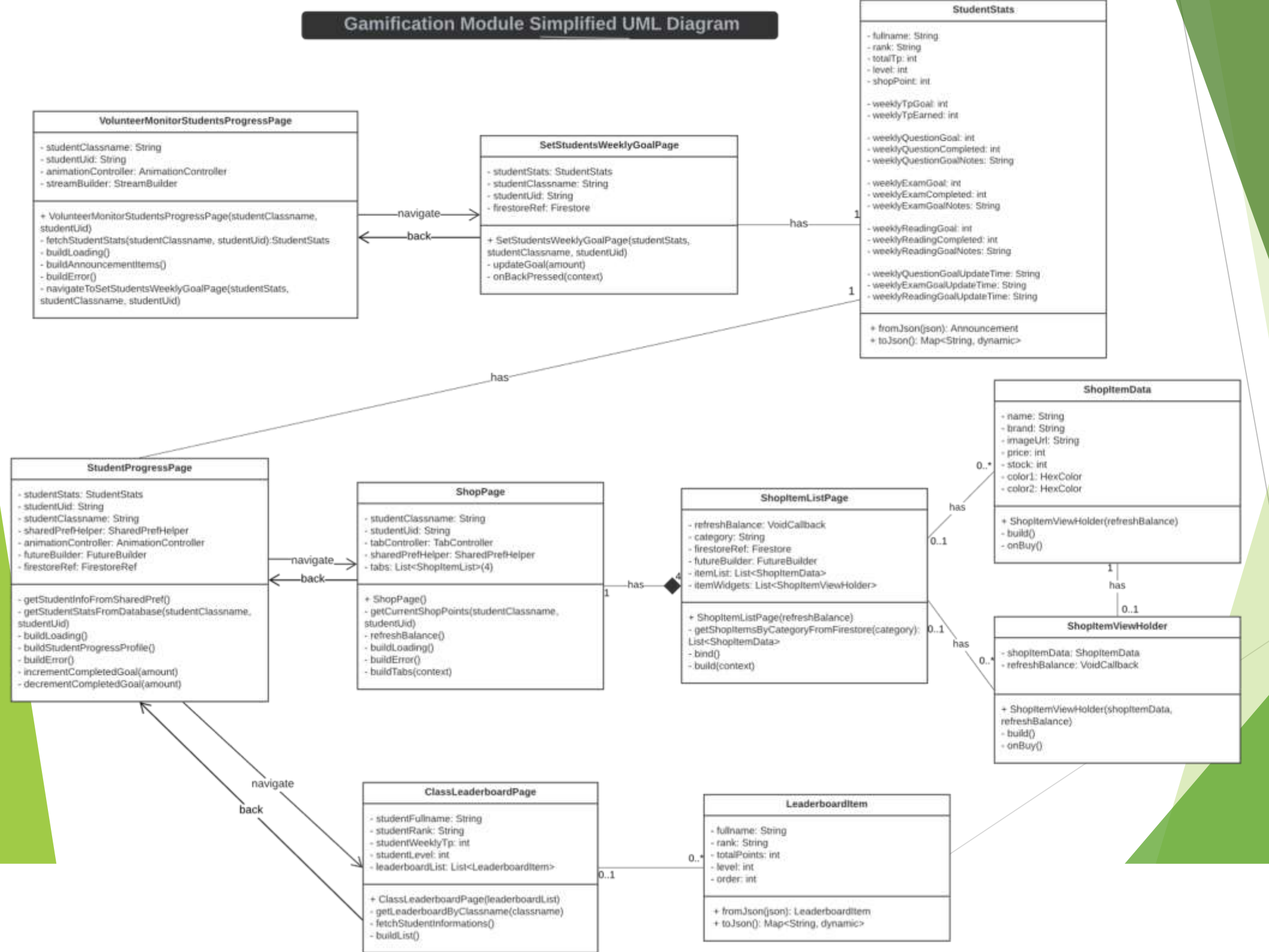
Some of the Modules in the App

Gamification Module

- Gamification concepts are added to motivate students to work harder
- In our real BBY project, all of the students have an advisor volunteer
- Gamified student profiles allow volunteers to set weekly goals and students can see their set goals and update their progress when they succeed
- Students collect **TP** points while they are completing their goals
- TP points are used for rank system and level system
- Collected TP points in a week are converted to shop points(**SP**) based on students success rate in every week

  For example student has 100 TP weekly goal but he/she completed %80 of it, in this case, gained SP would be 100 * 0.8 = 80; however if he/she would completed %130 of the goal, then gained SP would be 130 SP.
- Students can spend their shop points in the **Shop** to buy real items that are stored at ITU BBY club warehouse

Gamification Module Simplified UML Diagram



Some of the Modules in the App

One-to-one Lecture Module

- In the BBY project, students sometimes need help about a spesific topic and they tell this to their advisors, then advisors post these requests on the group to find a volunteer which can give one-to-one lecture about that topic.
- This mobile app provides a better solution for the matching problem described above
- Student-volunteer matching algorithm;

  - has 2 pools which are student request pool and volunteer profile pool

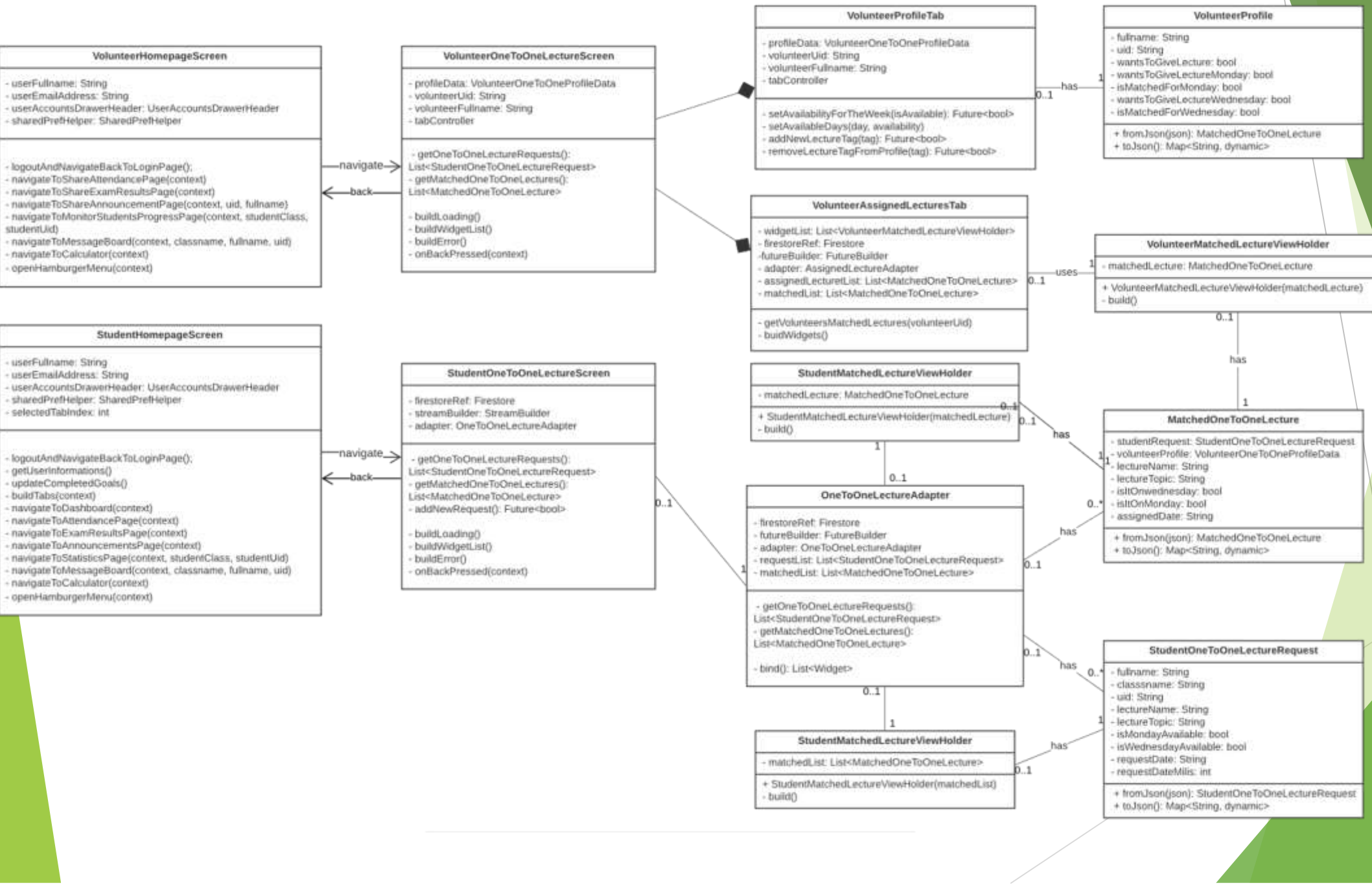
  - prioritizes students by request date

  - selects available volunteers for that week

  - tries to match students with volunteers by maximizing number of student requests that matched with a volunteer

- Students can have second request and matching system tries to match second requests after completing the first requests

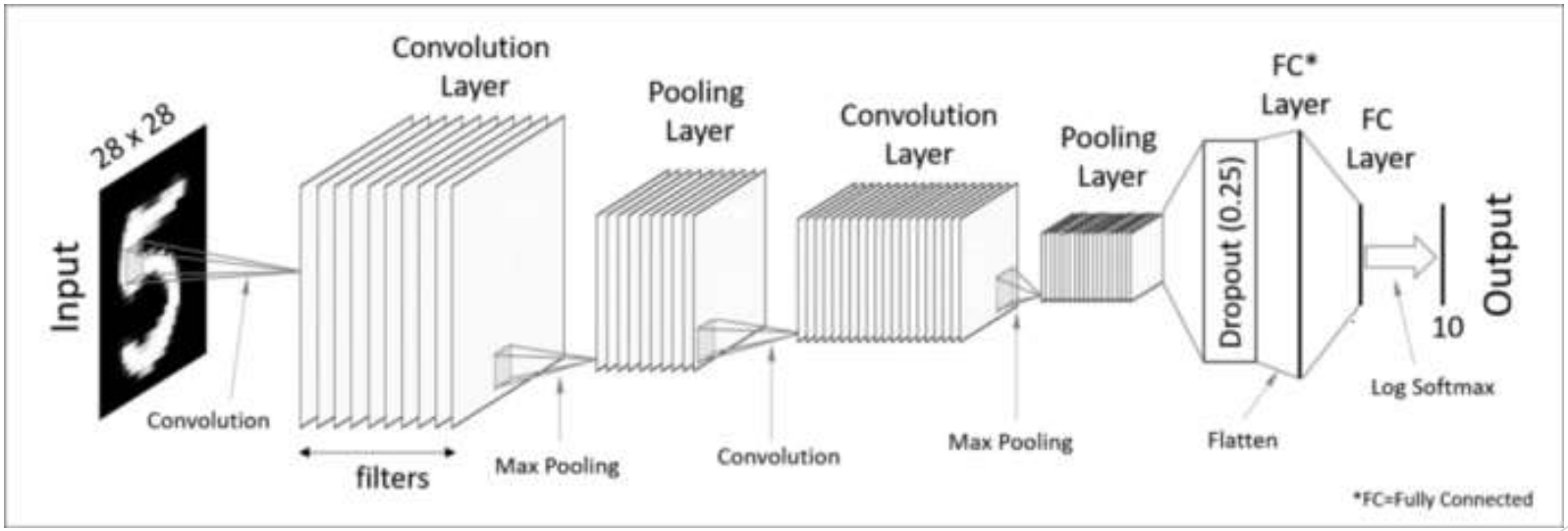
One-to-one Lecture Module Simplified UML Diagram



Some of the Modules in the App

Handwritten Calculator Module

- Purpose of the calculator is providing a calculator which has a different experience than normal calculators for the users
- Convolutional neural network(CNN) model is used to recognize digits
- MNIST data set which has 70.000 labeled sample data is used
- Architecture of the model is like in the figure below

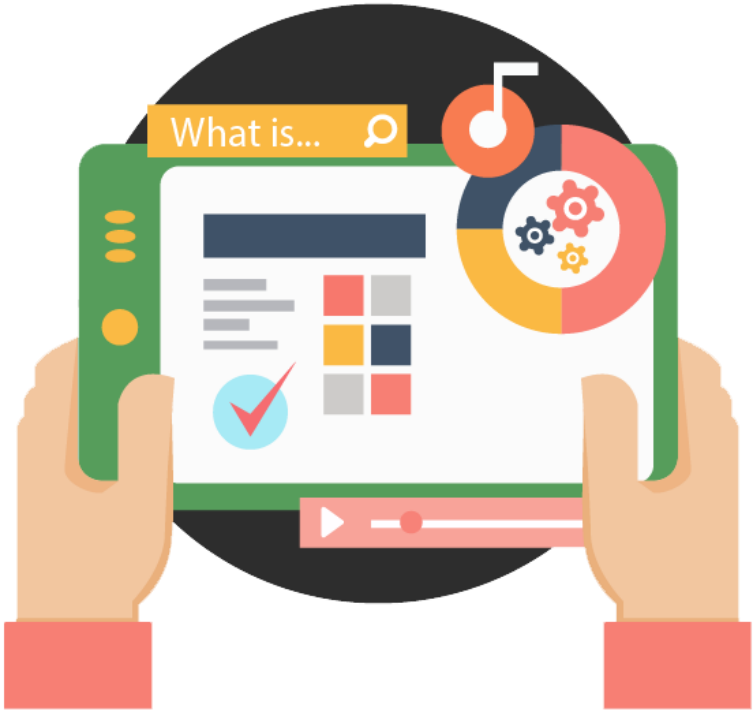


Source: https://medium.com/@himanshubeniwal/handwritten-digit-recognition-using-machine-learning-ad30562a9b64

Project Test Results

Entity	Evaluation Criteria	Result
CNN Model	Accuracy of the deep learning model should be at least %90	%98
Authentication Service	User should be able to signin in 5 seconds	0.2 second with a 8 mbps internet speed
Database Service	Any data in the app should be retreived in 3 seconds	List of 10 test documents are retreived in 0.6 seconds in average
Cloud Function	Resetting the weekly leaderboard for all classes should not be longer than 1 minutes	26 seconds in average
Storage Sevice	Images should be retrieved from the storage in 10 seconds	7 test images in a page are retreived in 2.2 seconds with a 8 mbps internet speed
Speech-to-text Module	Module should be able to understand both English and Turkish sentences correctly	Completed
Matching Algorithm	Complexity of the algorithm should not be more than quadratic	It is quadratic
Application Size	Size of the app should be less than 100 MB	29.6 MB for Android
Application Performance	App should run smoothly. FPS of the animations should be at least 30	40 FPS from Samsung S9 real device and Nexus 5X virtual device

Demo



Future Works

- Push notifications should be added
- Handwritten calculator should have more operators and UI & UX should change with a better ones
- Tasks & Achievements system should be added in the gamification module
- All exam results or attendance records should be published by uploading a single excell file that is in a spesific format
- Volunteers should be able to donate items and add these items into shop by using the mobile app.

QUESTIONS

Thank you for listening

References

- Y. E. KAŞ “Clean Architecture Nedir?,” *Medium*, 30-Dec-2019. [Online]. Available: https://medium.com/kodcular/clean-architecture-nedir-d5da08bd2f68.
- A. Volubuev “Mobile Architectures”, *SlideShare*, 08-Dec-2017. [Online]. Available: https://medium.com/kodcular/clean-architecture-nedir-d5da08bd2f68.
- A. Bizzotto, “Widget-Async-Bloc-Service: A Practical Architecture for Flutter Apps,” *Code With Andrea*. [Online]. Available: https://codewithandrea.com/articles/2019-05-21-wabs-practical-architecture-flutter-apps/.
- “SOLID,” *Wikipedia*, 14-Jan-2020. [Online]. Available: https://en.wikipedia.org/wiki/SOLID.
- “Beautiful native apps in record time,” *Flutter*. [Online]. Available: https://flutter.dev/.
- “React Native · A framework for building native apps using React” *React Native*. [Online]. Available: https://reactnative.dev/.
- A. Ravichandran, “React Native or Flutter-What Should I Pick To Build My Mobile App?,”*Medium*, 21-Jun-2019. [Online]. Available: https://medium.com/@adhithiravi/react-native-vs-flutter-what-are-the-differences- b6dc892f0d34.
- “XD to Flutter”. [Online]. Available: https://xd.adobelanding.com/xd-to-flutter/
- “Cloud Speech-to-Text”. [Online]. Available: https://cloud.google.com/speech-to-text
- “draw.io”. [Online]. Available: https://www.draw.io