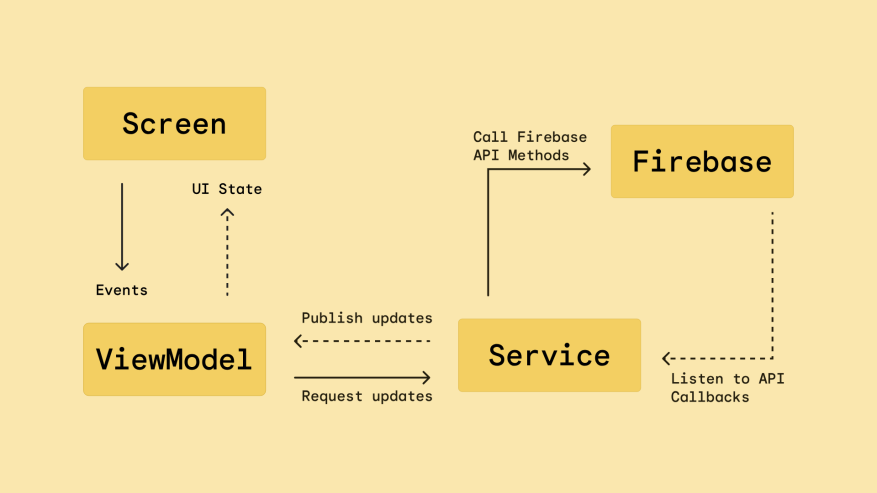
Coding Standards

MVVM Architecture

- Model-View-ViewModel Architecture
- Screen Data is accessed and modified in Respective ViewModel files, Screen UI is created and displayed in Screen files.
- In-depth description of MVVM architecture can be accessed through the link to this blog post.



https://firebase.blog/posts/2022/04/building-an-app-android-jetpack-compose-firebase

KDoc Function Headers

- Function headers to describe function's usage and parameters.
- In-depth description and example usage of KDoc documentation can be found here. Example of KDoc usage from StorageServiceImpl.kt file below.

```
/**
 * Retrieves course data associated with a specified courseId as a CourseData object.
 *
 * @param courseId The unique identifier that identifies a certain course in Firestore
 * @param onSuccess A callback function called when the course data is successfully retrieved.
 *                  Receives the retrieved CourseData object as a parameter.
 * @param onError A callback function called when an error occurs during data retrieval.
 *                  Receives an error message (String) as a parameter.
 */
override suspend fun getCourseData(
```

Retrieves course data associated with a specified courseId as a CourseData object.

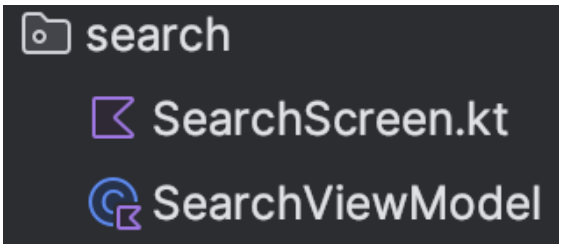
Params: `courseId` - The unique identifier that identifies a certain course in Firestore
`onSuccess` - A callback function called when the course data is successfully retrieved. Receives the retrieved CourseData object as a parameter.
`onError` - A callback function called when an error occurs during data retrieval. Receives an error message (String) as a parameter.

Service Oriented Architecture

- Utilizes defined interfaces and dependency injection.
    - Back-end service providers such as Firebase referenced only in these service files.
    - See StorageService and AccountService files.

File Locations and Modularization

- Files are placed in packages that correspond to their intended use.
    - SearchScreen.kt and SearchViewModel can be found in the search package.
- Repeated code should be modularized if possible



Variable Naming Standards

- All variable names clearly show intent and reasons for usage, no one-letter variables.
- Named constants and defines are used instead of any “Magic Numbers”
- Camelcase Variables and Function Names
    - Good:
        - coursesToDisplay
        - onDateSelected
        - onSettingsClick
    - Bad:
        - class_schedule
        - temp
        - Home_Screen_Content

Coding Standards

- MVVM architecture
  -https://firebase.blog/posts/2022/04/building-an-app-android-jetpack-compose-firebase
    - Screen data is accessed and modified in ViewModel files, Screen UI is created/displayed in Screen files
- KDoc function header comments for each function
- Service oriented architecture that utilizes defined interfaces and dependency injection
    - Back-end service providers like Firebase should only be referenced directly here
    - See StorageService and AccountService as examples
- Files are placed in packages that correspond to their intended use
    - Example: the signup screen view models and views are in the sign_up package
- Variable names should clearly show intent, no one letter variables
- Repeated code should be modularized if possible
- No “Magic Numbers” use named constants/defines

Write camelcase variables and functions

Good:
myClassSchedule
mapView
classLocation

Def myFunction

Bad:

Def my_function

my_class_scheudle
map_view
test
Temp

notes:

- makeitso github has online graphs to describe mvvm