Software Detailed Design Description

6.1 Introduction

This document presents the architecture and detailed design for the software for the Archived project. The project serves as an auction house which allows producers, artists, and users to sell and purchase beats/tracks within the music community.

6.1.1 System Objective Section

The objective of this application is to provide an user interface that stores all beats and tracks posted onto the application into the Google Firebase database. In addition to that, the application contains a payment system allowing everyone to bid and buy beats and tracks as products. Users can also direct messages with each other to deliver counter proposals until the final agreement is met.

6.1.2 Hardware, Software, and Human Interfaces Section

6.2 Architectural Design Section

The Archived application will consist of many architectural pieces, which are organized by their features and the functions they perform within the application. These architectural partitions are ultimately divided by Core, Components, and Services, with Components containing reusable software pieces that permeate throughout the application, the Core containing feature specific software pieces, and the Services as the integration of Components and Core into the UI of the application.

6.2.1 Major Software Component Section

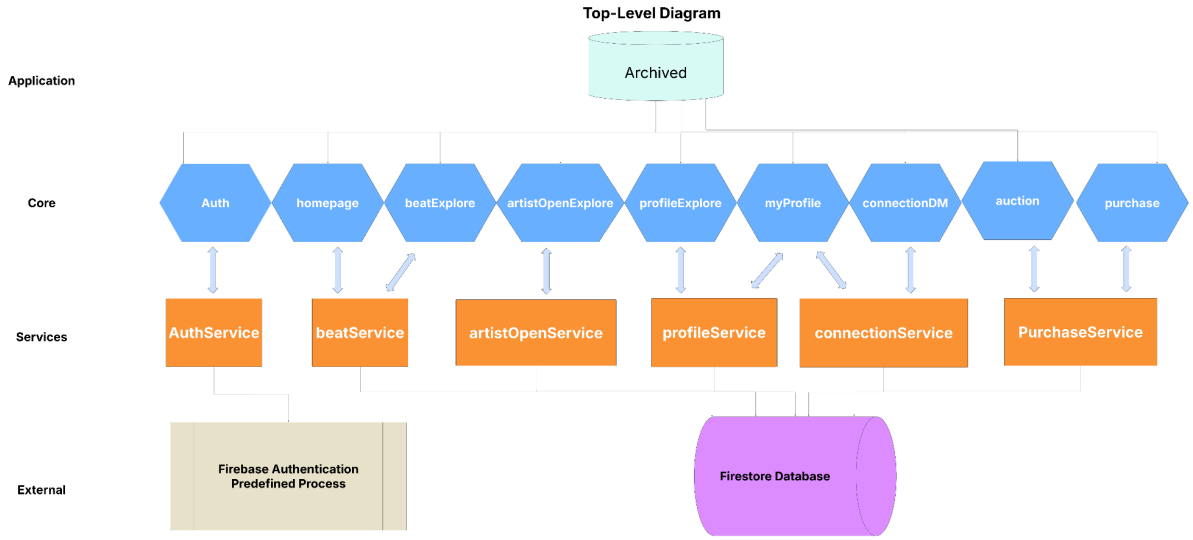
The major software components are contained within the Core partition of the software architecture. These components include the functional pieces of software that are responsible for the functionality of homepage, beatFeed, artistOpenFeed, profileExploreFeed, myProfile, and connectionDM. While the views responsible for each functional aspect of the application will be contained within their respective

partitions, many of the application's features will contain shared components of software. These components, including View components such as uploadView, beatRowView, beatProductView, marqueeTextView, waveformView, profileView, offerView, PurchaseView, and more will be used throughout the application to ensure consistency with look, feel, and usage of the application for the user.

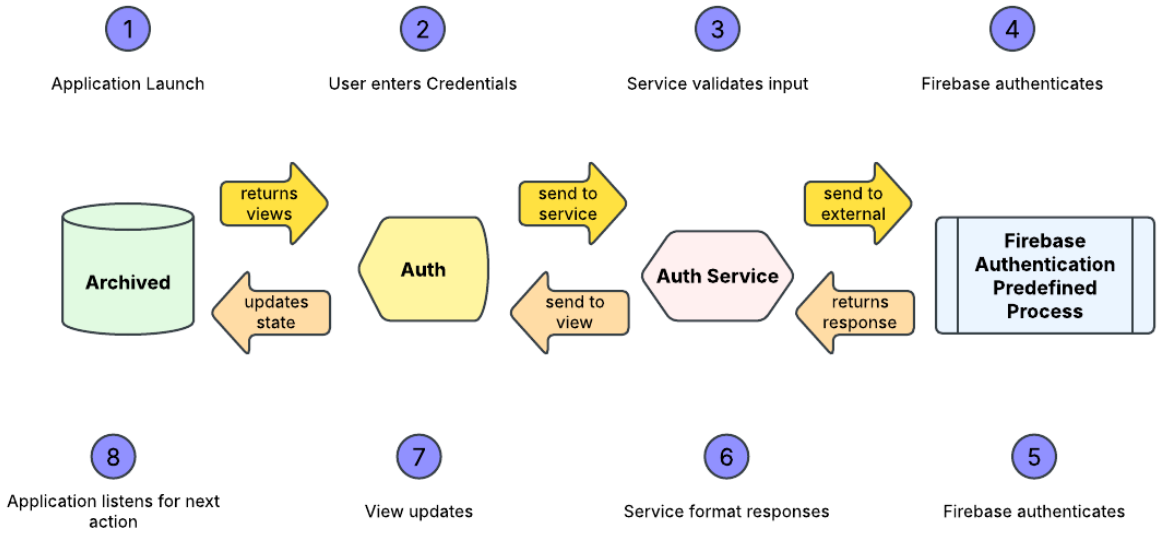
6.2.2 Major Software Interaction Section

The application will feature software components responsible for data services. These services will include the AuthService, for connecting to the Firebase Authentication product, as well various data services which will be responsible for enforcing and managing the Firestore database. These data services include the profileService, beatService, userService, openService, connectionService, purchaseService, and auctionService. Each service will contain software that ensures data is retrieved and committed to the Firestore Database in a manner that conforms to the application data schema and is also usable by the application views detailed above.

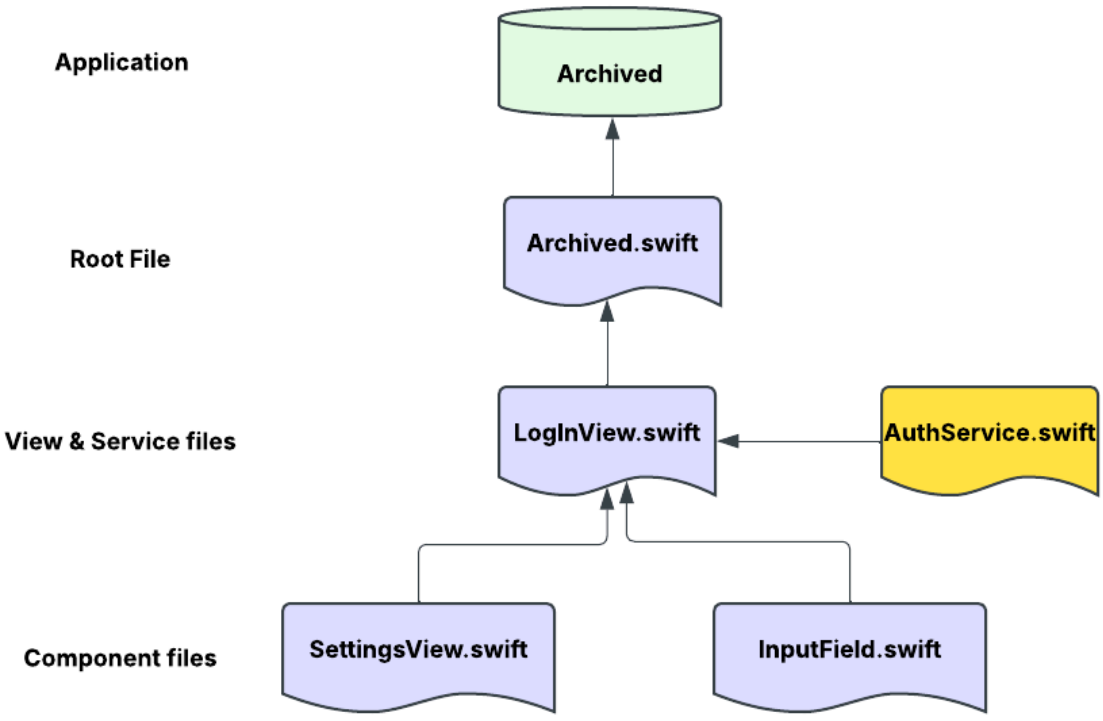
6.2.3 Architectural Design Diagrams Section



Data Flow Diagram



Implementation Diagram



6.3 CSC and CSU Descriptions Section

The Computer Software Components (CSCs) of Archived include all of the primary feature modules of the application such as:

- Profile
- ProducerBeats
- ArtistOpens
- Inbox
- Connections
- Settings
- Authentication
- PaymentSystem (to be implemented)

Profile is comprised of the following Computer Software Units:

- ProfileView
- EditProfileView
- AppUserModel
- UserProfileViewModel

ProducerBeats is comprised of the following Computer Software Units:

- BeatHomeView
- BeatRowView
- BeatProductView
- BeatService
- CommentService
- BeatViewModel
- BeatModel

- CommentModel

ArtistOpens is comprised of the following Computer Software Units:

- OpensHomeView
- OpenModel
- OpensHomeViewModel
- OpensListingView
- OpensProductView
- OpensService

Inbox is comprised of the following Computer Software Units:

- ActiveUsersView
- InboxView
- InboxRowView
- NewMessageView
- DirectMessageView
- InboxService

Connections is comprised of the following Computer Software Units:

- ConnectionListView
- ConnectionViewModel
- ConnectionModel
- ConnectionService

Settings is comprised of the following Computer Software Units:

- SettingsView
- SettingsOptionViewModel

- SettingsModel

Authentication is comprised of the following Computer Software Units:

- LogInView
- RegistrationView
- ProfilePhotoSelectorView
- AuthViewModel
- AuthHeaderView
- UserService

PaymentSystem is comprised of the following Computer Software Units: (to be implemented, subjected to change)

- PaymentSystemViewModel
- OffterView
- PurchaseView
- PaymentSystemModel
- PaymentAuthenticationModel

6.3.1 Detailed Class Descriptions Section

Each CSU is composed of a struct or class that performs a specific function in Archived

ProfileView: responsible for providing an UI view of each user's profile section

EditProfileView: allows user to edit profile, including display name, profile photo,

AppUserModel: contains an instance of a Firebase User data in the form of a member variable for the ID, display name, username, profile picture url, and email of the user

BeatHomeView: responsible for providing an UI view of the beat homepage; lists beats posted by producers in chronological order

BeatRowView: a component within BeatHomeView that's served as a template for beat listing in row view

BeatProductView: a component within BeatHomeView that's served as a template for beat listing in full screen navigation view

BeatModel: contains an instance of a Firebase beat data in the form of a member variable for the ID, beat name, file url, cover art url, beat description, tag, price, purchased (bool), bid (bool), like counter, comment counter, and archive counter

CommentModel: contains an instance of a Firebase comment data in the form of a member variable for the ID, username, comment, like counter, and reply counter

OpensHomeView: responsible for providing an UI view of the opens homepage; lists opens posted by artists in chronological order

OpenModel: contains an instance of a Firebase open data in the form of a member variable for the ID, open name, file url, cover art url, open description, tag, price, purchased (bool), bid (bool), like counter, comment counter, and archive counter

OpensListingView: a component within OpensHomeView that's served as a template for open listing in a rounded rectangle view

OpensProductView: a component within OpensHomeView that's served as a template for beat listing in full screen navigation view

InboxView: responsible for providing a UI of the user's inbox, recent connections, and message request

ActiveUsersView: a component within InboxView; responsible for displaying the active users the user has added connections with

InboxRowView: a component within InboxView; responsible for display the most recent message with a user in row view; contains data such as username, timestamp, and most recent text

NewMessageView: responsible for providing an UI to allow user to send new message to other users the user has made connection with

DirectMessageView: responsible for providing an UI of full-screen direct messaging with another user; contains component to allow user to type and send out and receive messages

ConnectionListView: provides a list view UI displaying all the users the user has made connections with

ConnectionModel: contains an instance of a Firebase connection data in the form of a member variable for the ID, username, timestamp, mutual connections (string[]), and mutual connection counter (int).

SettingsView: provides a list view UI displaying all the settings the user can change and view

SettingsModel: contains an instance of a Firebase user's settings data in the form of a member variable for the ID, username, timestamp, mutual connections (string[]), and mutual connection counter (int).

LogInView: responsible for providing an UI of the log in view, allowing user to input their email address and password

RegistrationView: responsible for providing an UI of the registration view, allowing user to sign up by inputting their email address, username, display name, profile picture, and password

ProfilePhotoSelectorView: a component that allows user to upload a profile picture and resize/realign such picture when the user is signing up or logged in

AuthHeaderView: a component within LogInView or RegistrationView to display the header of the log in or registration view.

OfferView: provides an UI for user to bid on beat/open

PurchaseView: provides an UI for user to purchase a beat/open

PaymentSystemModel: contains an instance of a Firebase user's payment system data in the form of a member variable for the ID, username, contact information, and various card information

LogInView: responsible for providing an UI of the log in view, allowing user to input their email address and password

PaymentAuthenticationModel: contains an instance of a Firebase user's payment system authentication data in the form of a member variable for the ID, username, contact information, and various card information

6.3.2 Detailed Interface Descriptions Section

UserProfileViewModel: fetches information of an user from the firebase and stores user information into database

BeatService: responsible for interfacing between the BeatHomeView and the Cloud Firebase Database application. It contains a member function for retrieving a member document from the Firestore Database, as well as one for updating or creating a beat's document in the Cloud Firestore Database.

CommentService: responsible for interfacing between CommentModel and the Cloud Firebase Database application. It contains a member function for retrieving a member document from the Firestore Database, as well as one for updating or creating a user's profile document in the Cloud Firestore Database.

BeatViewModel: fetches information of a beat from the firebase and stores beat information into database

OpensHomeViewModel: fetches information of an open from the firebase and stores open information into database

OpensService: responsible for interfacing between the OpensHomeView and the Cloud Firebase Database application. It contains a member function for retrieving a member document from the Firestore Database, as well as one for updating or creating an open's document in the Cloud Firestore Database.

InboxService: responsible for interfacing between the InboxView and the Cloud Firebase Database application. It contains a member function for retrieving a member document from the Firestore Database, as well as one for updating or creating an inbox's document in the Cloud Firestore Database.

ConnectionViewModel: fetches information of a connectionListView from the firebase and stores connection information into database

ConnectionService: responsible for interfacing between the connectionListView and the Cloud Firebase Database application. It contains a member function for retrieving a member document from the Firestore Database, as well as one for updating or creating a connection's document in the Cloud Firestore Database.

SettingsOptionViewModel: fetches information of a SettingsView from the firebase and stores user's settings information into database

AuthViewModel: responsible for interfacing between the LogInView, or the RegistrationView, and the Cloud Firebase Authentication application. It contains a member function for retrieving a member document from the Firestore Database, as well as one for updating or creating a user profile's document in the Cloud Firestore Database.

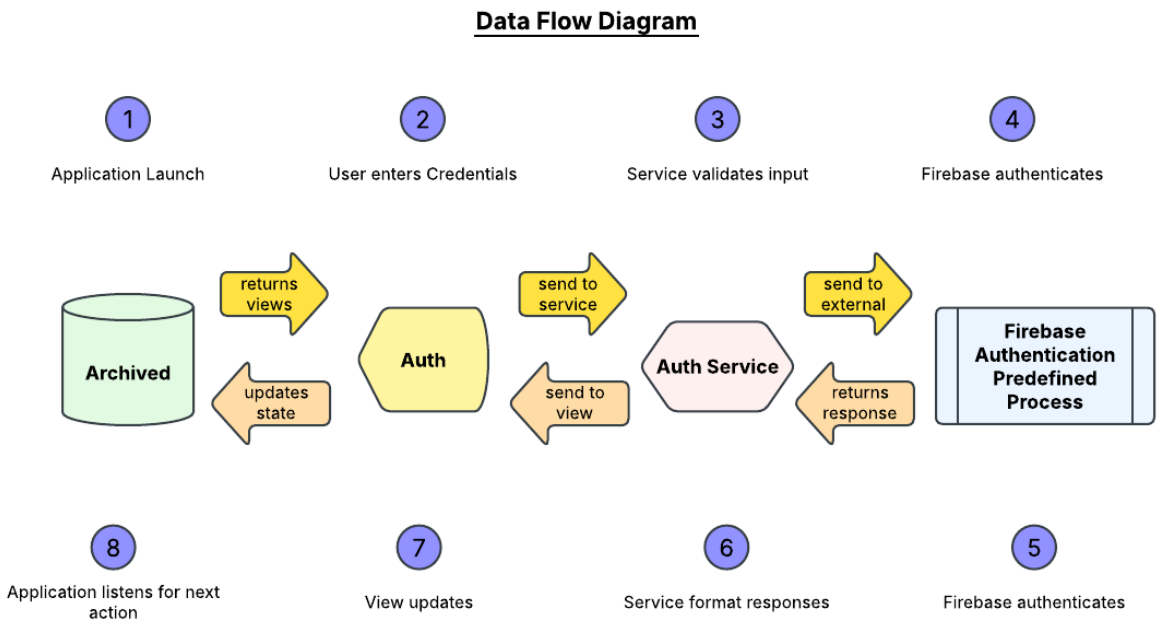
UserService: responsible for interfacing between AppUser and the Cloud Firebase Database application. It contains a member function for retrieving a member document from the Firestore Database, as well as one for updating or creating a user's document in the Cloud Firestore Database.

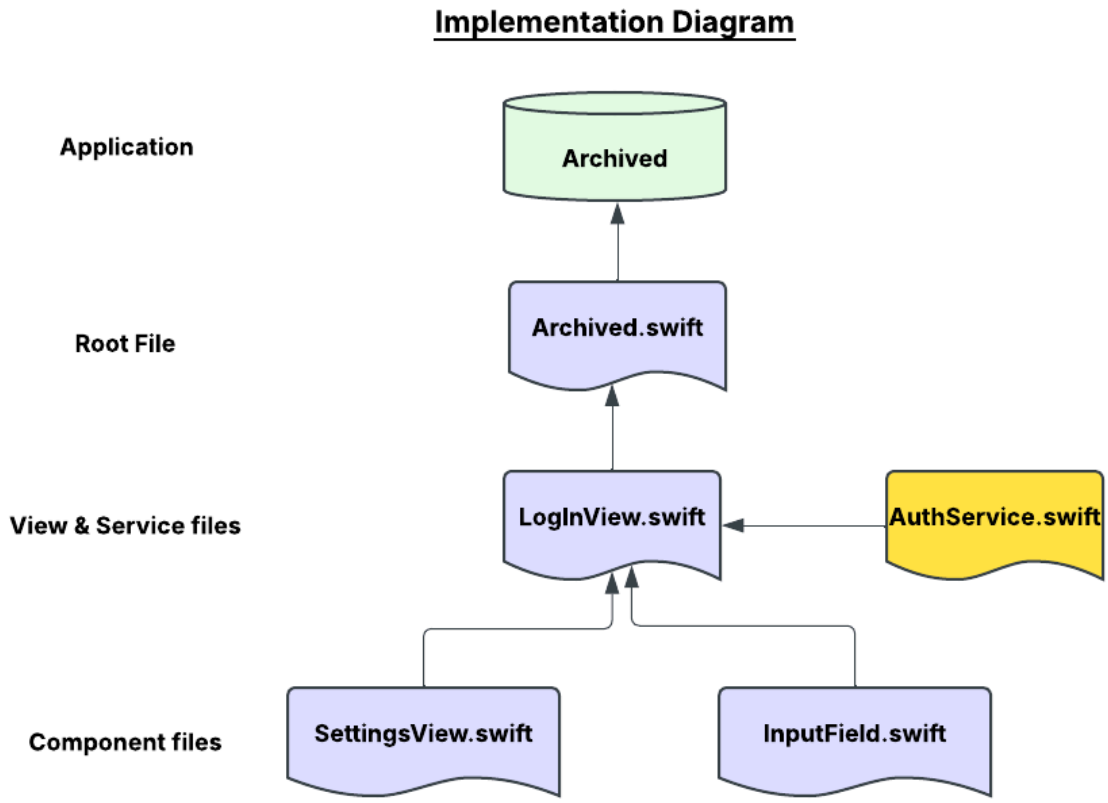
PaymentSystemViewModel: fetches information between OfferView and PurchaseView, and the Cloud Firebase Database application. It contains a member function for retrieving a member document from the Firestore Database, as well as one for updating or creating a user's payment system's document in the Cloud Firestore Database.

6.3.3 Detailed Data Structure Descriptions Section

As Archived is a SwiftUI project, all data structures are composed in the context of the computer software unit classes described in Section 6.3.1 above

6.3.4 Detailed Design Diagram Section

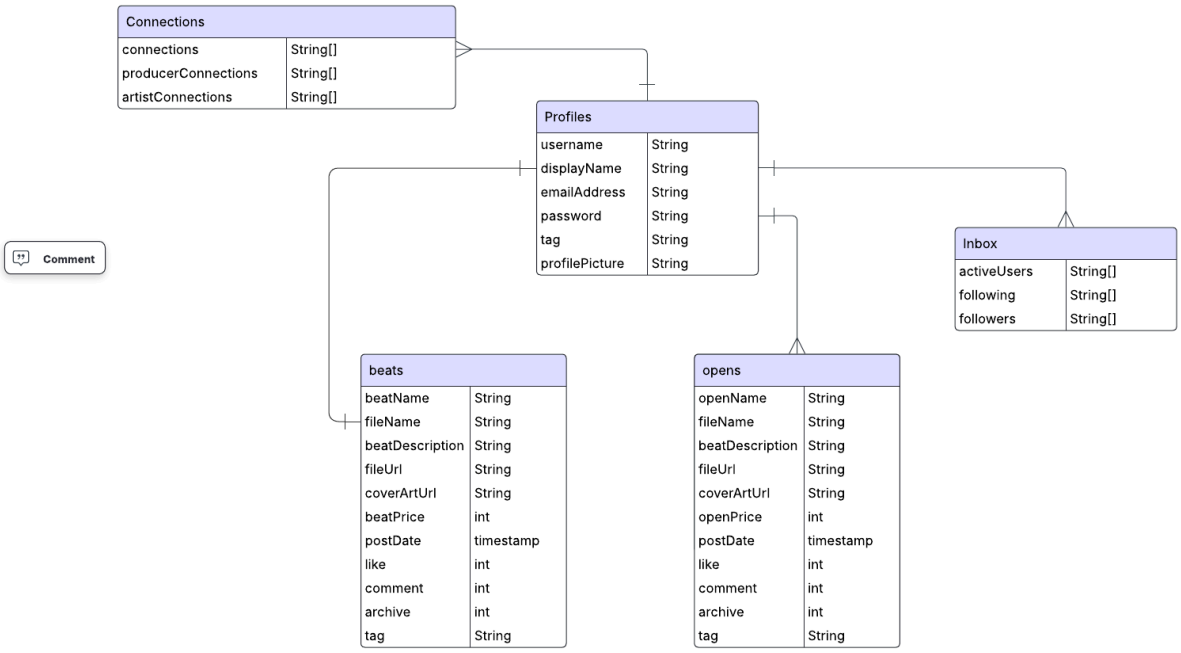




6.4 Database Design and Description Section

Firestore database collections are based on each feature of Archived. Each document in collections related to user data contains an Id field to reference the related user's profile. The profile collection contains additional User data not handled natively by the Firebase Authentication User.

6.4.1 Database Design ER Diagram Section



6.4.2 Database Access Section

Firestore database collections are access by dedicated Service classes, respectively created for each collection. See Section 6.3.2 for information regarding the interface of the Views with the Database

6.4.3 Database Security Section

Database security is handled by native Firestore Database Security protocols. Archived is configured to require an authenticated query or change prior to granting access to the database. In addition, Edit Permissions are restricted to users with a Profile that is an Admin and Read Permissions are granted to users who have a matched email address and username in their profile. This ensures that bad actors are unable to modify database data and restrict any data from other accounts to be visible.