# Design and Implementation

Design details relate to Software Engineering perspective of the application. The user interface is always an essential perspective for any application. A happy client lies in the open truth nature of a GUI. The roots of the application lie in it being a mobile application, hence making it accessible almost everywhere. Mobile application or as it’s popularly known as ***Mobile ML*** is the forefront of modern-day machine learning. Making prediction and or recommendation available to handset is considered a powerful feat since handset or IOT devices tend not to have the level of processing power as an ordinary computer or laptop. To overcome this restriction processing power mobile application and browser-based interaction that require heavy load of processing are divided into frontend and backend part of the application. This also maintains a source of ***abstraction***for the user where what is visible is present locally on the smart phone or browser whilst the information is fetched from server via an API call that process the information on a server. Figure 4.1 shows the system architecture the project the detail of which have been prescribed in the sections that follow.

## 4.1 User Interface

Since this being an Android based application popular technology stack for this domain consist of Java for android, Kotlin, Flutter and React Native for planning and executing the frontend part of the project. The major difference between these approaches is the programming language that these technologies represent and the cross-platform compatibility that they provide. Java for android, as it clearly states works with Java that is compatible with Android and brings the whole functionality of the platform ‘Write once, run anywhere’ ***WORA*** to good use. It has been the founding stage of android programming for smartphones or any smart device that can even be incorporated into a vehicle. Still to this day it works on defining new paradigms and better interactivity to smartphones. After the base development for mobile applications was set conglomerates saw a huge potential in the industry for sciences where computation was available to anyone, anywhere on just their fingertips. Multiple mobile development platforms were present at a time these including Swift or Objective-C for iOS development, C and or C++ for windows phone platform and Java or Kotlin for android. However, these were used in native software development i.e. multiple programmers or developers were required which knew these technologies to work and deliver application on multiple platforms. To overcome this restriction cross-platform technologies had been introduced recently which provide a wrapper layer over there standard library to represent a single code base on multiple platforms these include Flutter by Google introduced in 2018 and React Native introduced by Facebook in 2015 with the motto of ‘Learn once, write anywhere’. While these did solve a bulk of problems, but what can’t be overlooked is the efficiency in these platforms. Applications that are developed in native framework outperform those with hybrid technology stack, this is mainly because the native code is executed as is in the application whereas, hybrid applications provide a wrapper around the code base which is in turn executed to be compiled as close as to native applications. Secondly, these hybrid or cross-platform technology stacks are still new to this paradigm and are in constant state of development where updates are still in progress and beta programs for new features are available for developers to enroll. Justifying these statements, the frontend for the application has been implemented in *Java for Android using Android Studio*.

## 4.2 Server Development

This being a Machine Learning centered project the ideal candidate that came to mind is ***Python***. Besides machine learning algorithms what the technology also provides is web development frameworks like Flask and Django which are an entire solution in themselves for providing standalone server deployments for making backend APIs calls that return the desired results providing the base input for the application. The server needs to be hosted to a secured platform as well which provides ***http*** web access to the exposed APIs for the backend. ***Heroku*** is viable solution in this case. Heroku is a cloud hosted platform that support frameworks like Python, Ruby on rails and Node.js servers natively utilizing the in-built AI to recognize technology stack and completely setup the requirements and multi-threaded servers by itself. Python has been a choice for server-side development for quite some time now, YouTube being one of the major platforms to switch its entire technology stack to Python from PHP due to its flexibility, performance, and ease of use. As well as being a proficient language in itself it comes with a handful of frameworks to adjust to the user needs, these consist of FastAPI, Flask, Django, and Dash Plotly which provide a high-level programming interface specially in the domain of Data Sciences and Machine Learning applications. In-terms of setting up a server, since Django is mainly used for multi paged heavy load bearing websites the ideal choice for this scenario was ***Flask*** which provides a bare bone structure of setting up your own environment and laying down web APIs in an instant with great flexibility and user provided authentication. The backend hosted via Heroku provides with 2 exposed function call or APIs that when provided with the current logged in user ID will return a list of their recommended items in the of product ID based on the user’s liked list. These product IDs are then in turn populated into the real-time Firebase database and the results fetched to the frontend of the application.

## 4.3 Database and Authentication

Database and authentication for the project is a major part for the application. It concerns with an important aspect of retrieving and writing data for the purpose of user login authentication and provided the models the data for making the right recommendation for the user. SQL databases have been a point of source for many production-based applications that can be deal with enormous chunk of data keeping the relation-based schema for tables in high regard. However, with the present paradigm of cloud-based software as a services ***SaaS*** many database applications have switched to pre-hosted ***NoSQL*** based database services such as the likes of Firebase and MongoDB. These databases provided quick ways of authentication utilizing the services of Googles authentication. As a comparison NoSQL database have proved to be more viable and sustainable for content-based applications where instant delivery overrules the need of relations or foreign keys in a database. Plus, these databases come pre-hosted that provide an authentication of their own and providing an additional layer of security with keys and access shared specifically with those people who require the access. In addition to an additional layer of security these services are highly scalable, with the database hosted on an entirely different server/remote server no matter the load on the backend or frontend of the application the database is in itself independent of the hits on the application. Even if the load increases on the database the remote machine can be configured to automatically adjust the tier/resources allocated to the database server making these services highly sustainable with zero to no down time anticipated. These databases consist of documents and collections instead of table and columns, and go hand in hand with Node based environments or dynamically typed language like Python due to their flexible nature as far as querying format is considered because ***JSON*** is a natively supported format for these applications/services. Whilst considering the option between MongoDB or Firebase by Google the choice to conclude was ***Firebase*** since Firebase provides a real-time database connectivity option and multiple authentication routes. Firebase is also considered the ideal choice for mobile ML solutions for its seamless connectivity and data engineering solutions that keep data-oriented delivery in high regard providing a ***serverless connection*** to any mobile application which makes the application independent of a compulsory backend server. All the data stored in the cloud is instantaneously available to the frontend regardless of the backend server status and all the query processing is done on the ***Google cloud platform*** which is what the Firebase database is hosted on. Since the database is hosted on Google cloud platform this provides us with additional strength for choosing the right platform of choice as far database connectivity is concerned. Scalability, zero down time and serverless hosting is what makes Firebase the choice of developers for native mobile applications.

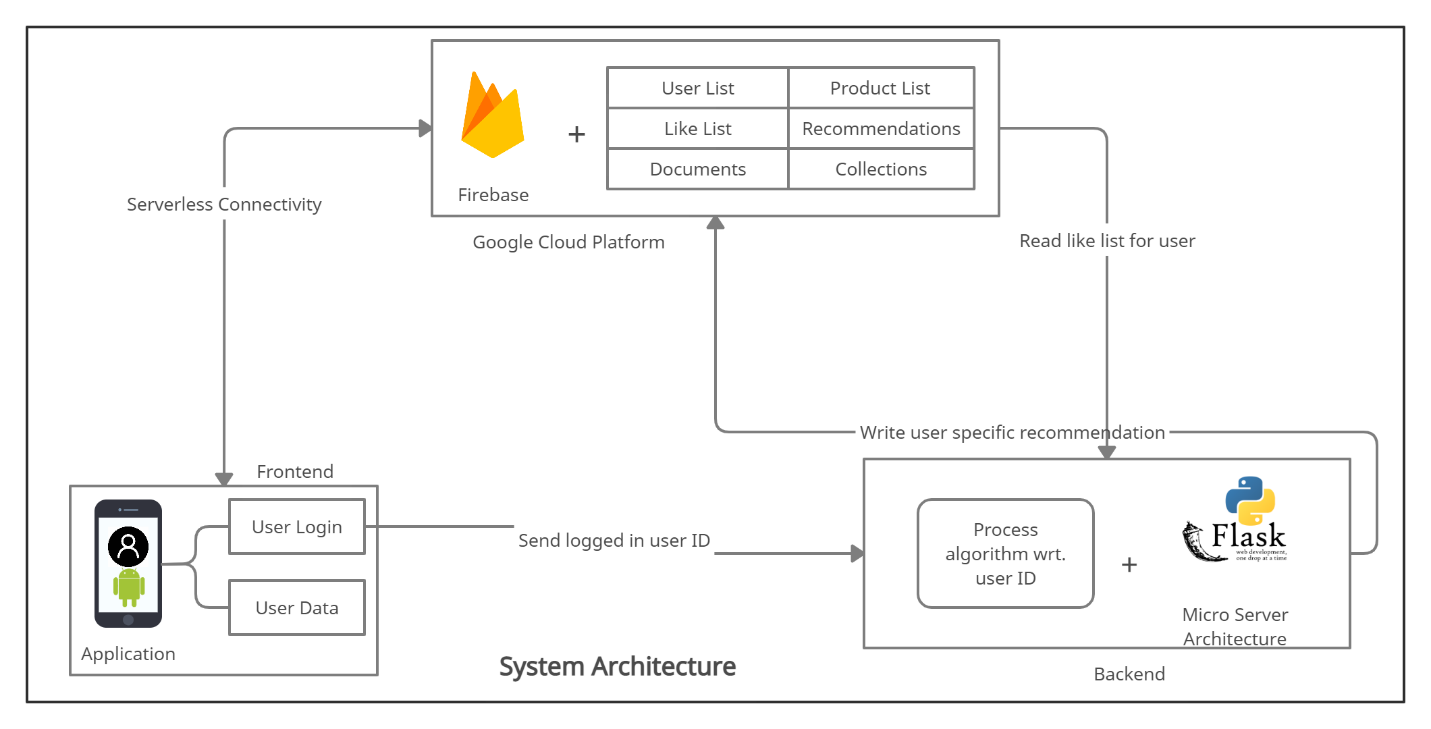


Figure . System Architecture

