Software Design Document

CIPHERSERVE

Version 1.0

Prepared by
GROUP 5

Pranav Baburaj(51)
Agustin Kiran Leo(74)
Arshakh Muhammed(13)
Alok Aravind(9)

TKM College of Engineering

3rd April 2023

Table of Contents

1. Introduction

This Software Design Document (SDD) is intended to provide a comprehensive overview of the design and architecture of a software system. The document is aimed at developers, architects, and stakeholders involved in the development of the system. It outlines the key design decisions, architectural patterns, and technologies used to build the system.

The SDD covers various aspects of the software design, including system architecture design, application architecture design, GUI design, API design, and technology stack. The document provides detailed descriptions of the design decisions and provides an overview of the system architecture, the application architecture, and the user interface design. Additionally, it provides details about the API design and the technologies used to build the system.

This document aims to serve as a reference guide for the development team throughout the development process. It will help to ensure that the team remains aligned with the design goals and can make informed decisions regarding the implementation of the system.

1.1 Purpose

The purpose of the Software Design Document (SDD) is to provide a comprehensive overview of the design of the software system being developed. It serves as a guide for the developers, testers, and stakeholders involved in the project, and provides a common understanding of the system architecture, application architecture, GUI design, API design, database design, and technology stack. The document outlines the technical aspects of the software and the design decisions that were made during the development process, ensuring that the software meets the specified requirements and performs as expected. The SDD also helps to identify potential issues and risks that may arise during the development and implementation phases, enabling proactive measures to be taken to mitigate them. Overall, the SDD serves as a critical reference document for the development team throughout the software development lifecycle.

1.2 Scope

The scope of this Software Design Document (SDD) is to provide a comprehensive overview of the design and architecture of the proposed application. It covers the system architecture design, application architecture design, GUI design (mockups), API design, and the technology stack that will be used to develop the application. The SDD aims to provide a clear understanding of the design and architecture of the application to the development team and stakeholders involved in the project. It will also serve as a reference guide for the development team throughout the development process. The SDD is intended to be a living document and will be updated as needed throughout the project lifecycle.

1.3 Intended Audience

The intended audience for this SDD includes software developers, College Professors, and other stakeholders involved in the software development process. This document is intended to provide a detailed design of the system architecture, application architecture, GUI design, API design, and technology stack, as well as any other relevant technical details related to the development of the software application.

The audience can review the overall design and technical description, and focus on specific components such as the system architecture, application architecture, GUI design, API design, and technology stack. Any concerns or areas of interest that could impact the development or evaluation of the software application can be noted for further discussion and clarification

1.4 References

- IEEE Standards Association. IEEE Std 1016-2009, IEEE Standard for Information Technology--Systems Design--Software Design Descriptions. IEEE, 2009.
- Martin, Robert C. Clean Architecture: A Craftsman's Guide to Software Structure and Design. Prentice Hall, 2017.

2. System Architecture Design

2.1 Description

The system architecture design is based on a client-server model. The client-side of the application runs on the user's mobile device, while the server-side runs on the Firebase platform.

The application's client side is built with Java and Android Studio. It consists of the user interface (UI) and application logic for managing user input, data processing, and output. Users can search for and identify nearby service centres, as well as browse available hardware and software parts for repair or replacement, using the UI's simple navigation menus and intuitive panels. To fetch and update data, the application logic communicates with the server using APIs.

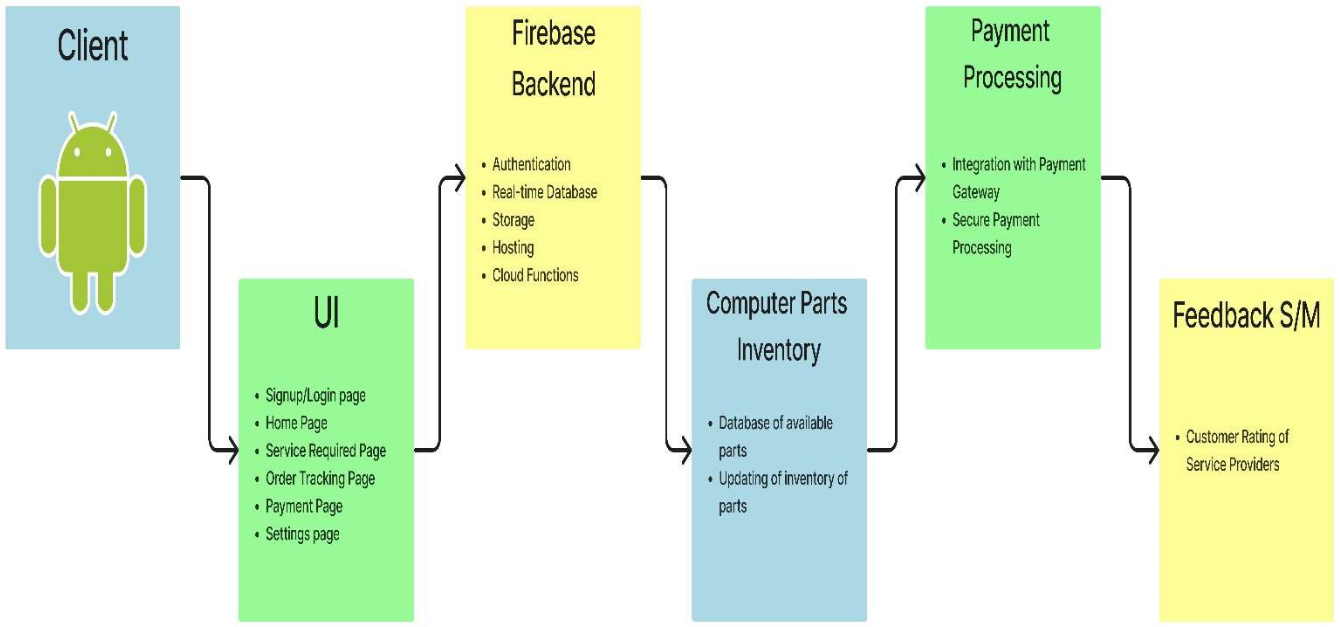
The server-side of the application is built with Firebase, a cloud-based platform for backend development. Firebase offers services such as real-time database, hosting, cloud storage, authentication, and security. The server-side stores and retrieves data using Firebase's real-time database, and push alerts are sent using Firebase cloud messaging. The app's client sideFirebase Authentication is used to safeguard user accounts and authorise data access.

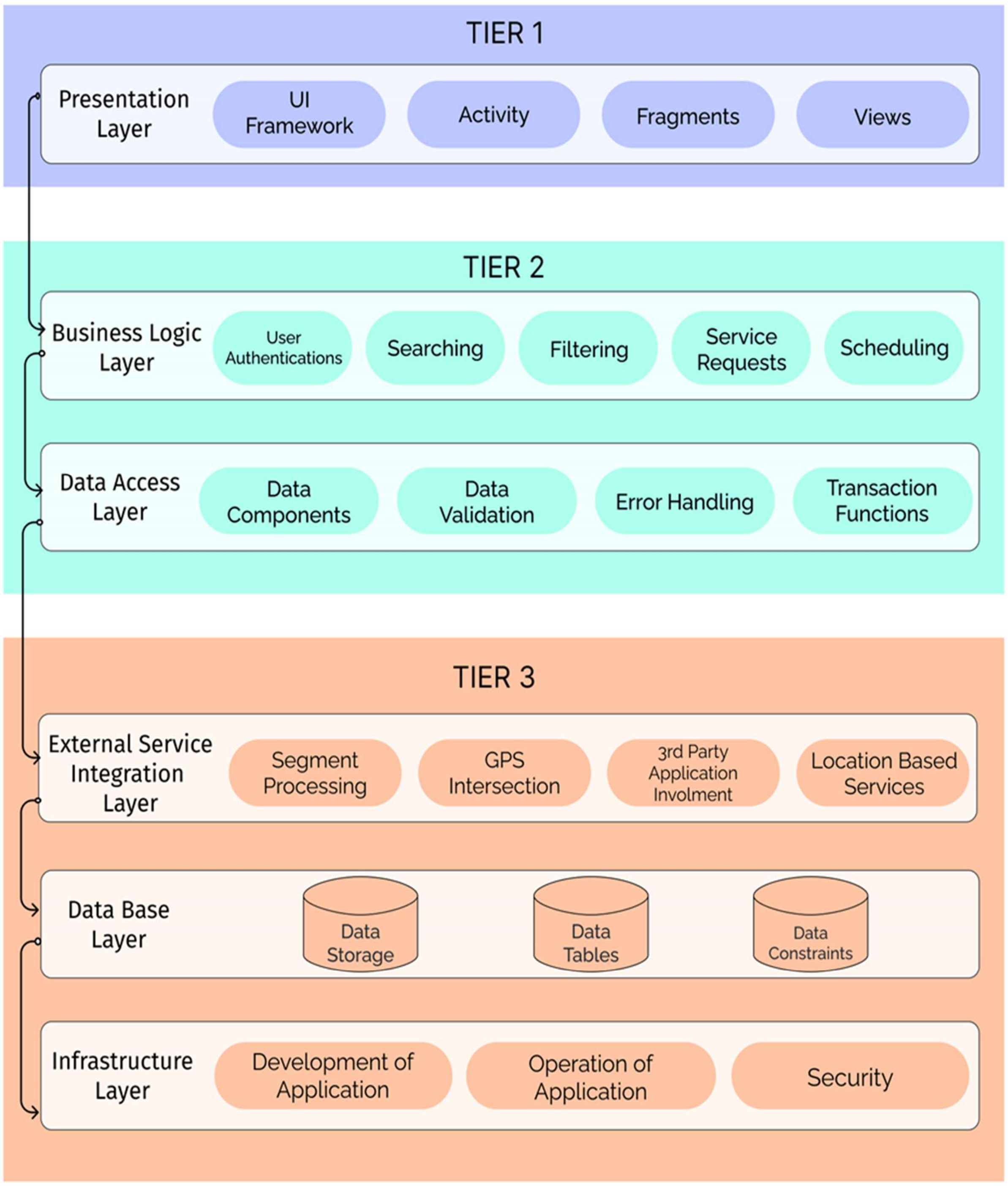
Overall, this architecture design enables the application to provide a seamless and reliable service to customers, with safe data storage and efficient lication is constructed using Java and Android Studio.

The architecture consists of multiple layers.The presentation layer is responsible for the user interface of the mobile application.It has features that let users engage with the application, including as activities, fragments, and views.The application logic layer controls the business logic of the application.The data access layer communicates with the database to keep track of the inventory of damaged parts and service requests. The external services integration layer integrates external services such as payment processing and GPS integration.The database layer is responsible for managing the application's data storage. The infrastructure layer uses cloud-based services for load balancers, server infrastructure, and network security. It is in charge of deploying and running the application.

Ultimately, the system architecture should be created with scalability, dependability, and security in mind, as well as an easy-to-use user interface. The design should take the demands of the end users into account and offer effective search and filtering capabilities for broken components, simple inventory and service request management, and smooth connection with third-party services like payment processing and location-based services.

2.2 Architecture




TIER 1
Presentation Layer
UI Framework
Activity
Fragments
Views
TIER 2
Business Logic Layer
User Authentications
Searching
Filtering
Service Requests
Scheduling
Data Access Layer
Data Components
Data Validation
Error Handling
Transaction Functions
TIER 3
External Service Integration Layer
Segment Processing
GPS Intersection
3rd Party Application Involment
Location Based Services
Data Base Layer
Data Storage
Data Tables
Data Constraints
Infrastructure Layer
Development of Application
Operation of Application
Security

3. Application Architecture Design

Cipherserve is a mobile application designed for individuals or businesses that require computer or laptop repair services.The application architecture consists of two components: a frontend built using Flutter framework and a backend built using Firebase.

The frontend of the application can be implemented using a variety of methods and tools, including:

- User interface (UI) design: In order to give customers a comfortable experience, the user interface design should adhere to the Android design principles. Also, the layout should be simple to use and visually appealing. To allow users to interact with the programme, it has to have components like buttons, text fields, dropdown menus, and checkboxes.
- Navigation: The navigation should be made to be simple and straightforward to use. It should make switching between the application's many screens easy for users. Techniques like the drawer menu, bottom navigation, and tabs can be used to accomplish the navigation.
- Notifications: Users should be given notifications from the application to let them know when critical things like service requests, appointment confirmations, and payment reminders happen.
- Search and filter: The application should include search and filter functionality to enable users to quickly find the damaged parts they are looking for.
- Integration with external services:To give consumers a seamless experience, the application should be built to interface with external services like payment processing and GPS integration.

The backend of the application can be implemented using a variety of methods and tools including:

1. Application server: The application server will serve as the backbone of the system, handling all incoming requests from the front-end and interacting with the database to provide the necessary data. The server will be built using Java and the Spring Framework, with a RESTful API to facilitate communication between the front-end and back-end.
2. Database: The database will be used to store all the necessary data, such as part details, service availability, user accounts, and orders. The database will be designed using MySQL or PostgreSQL to ensure that it is secure, reliable, and scalable.
3. APIs: The back-end will expose APIs that enable the front-end to interact with the application server and retrieve the necessary data. These APIs will be secured using OAuth 2.0 or JWT (JSON Web Token) to ensure that only authorized users can access them.

4. Cloud services: The application may use cloud services to store data or perform certain tasks such as image recognition or payment processing. These services may include Amazon Web Services, Google Cloud Platform, or Microsoft Azure.
5. Payment gateway integration: The application will integrate with a payment gateway such as Stripe, PayPal, or Braintree to enable users to make payments for their orders. The payment gateway will be securely integrated into the application server using their respective APIs.
6. Security: The back-end will include various security features to ensure the integrity and confidentiality of user data. These features may include data encryption, firewall protection, and user authentication and authorization.
7. Caching: The application may use caching technologies such as Redis or Memcached to improve the performance of the application by reducing the number of requests to the database.

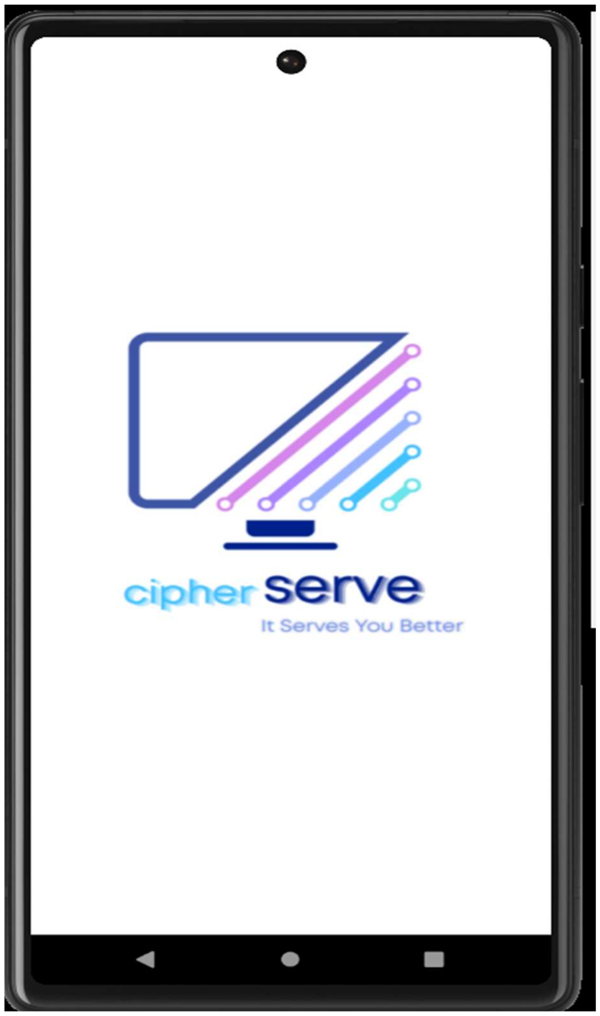
Overall, the back-end of the application architecture will be designed to provide a secure, reliable, and scalable infrastructure for the application. It will handle all the complex business logic and data management required for the application to function properly, while exposing APIs that enable the front-end to communicate with the back-end seamlessly.

4. GUI Design

4.1 User Interface Design

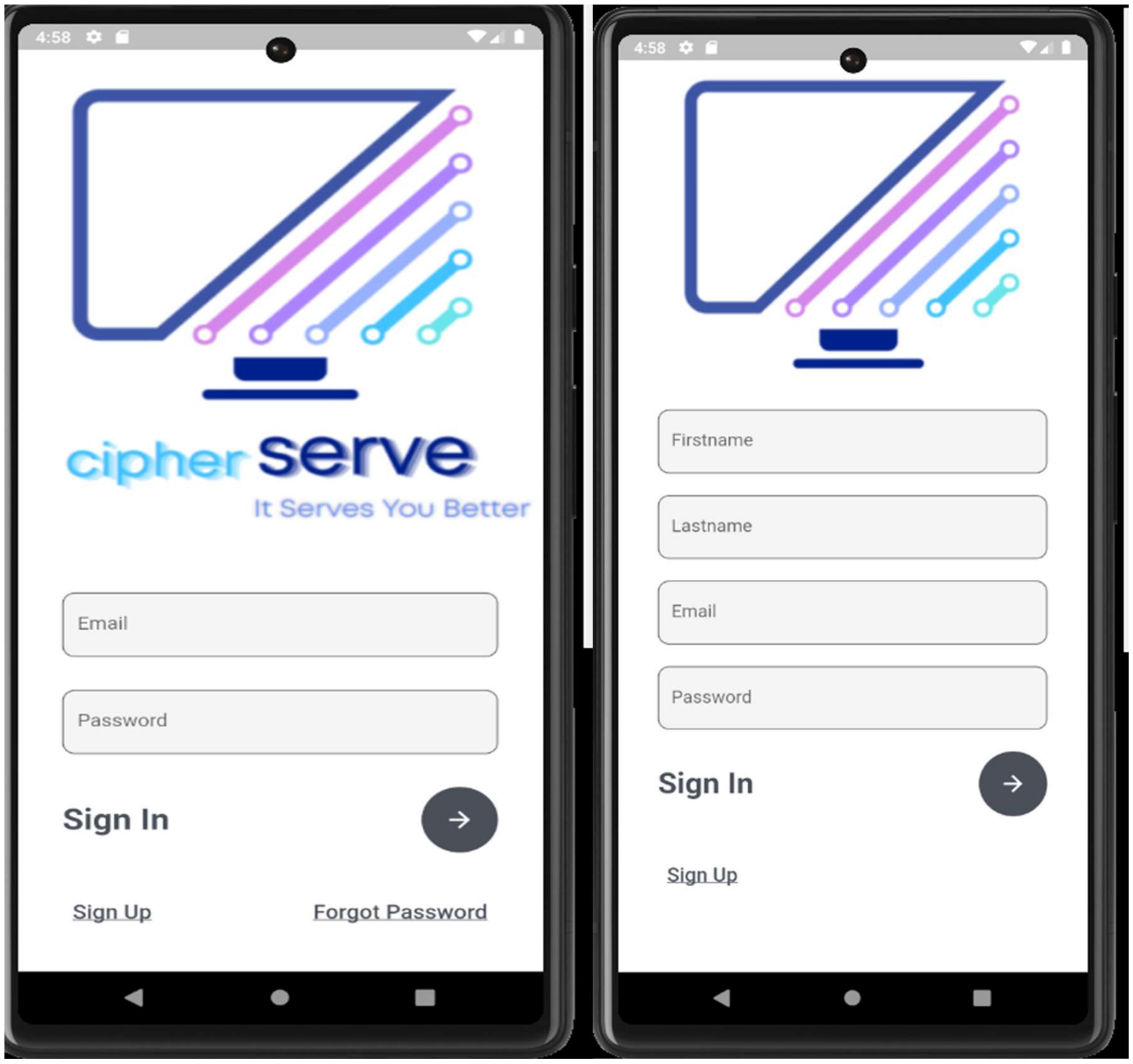
a.Splash Screen

- Displaying the application logo and name for 4 seconds.



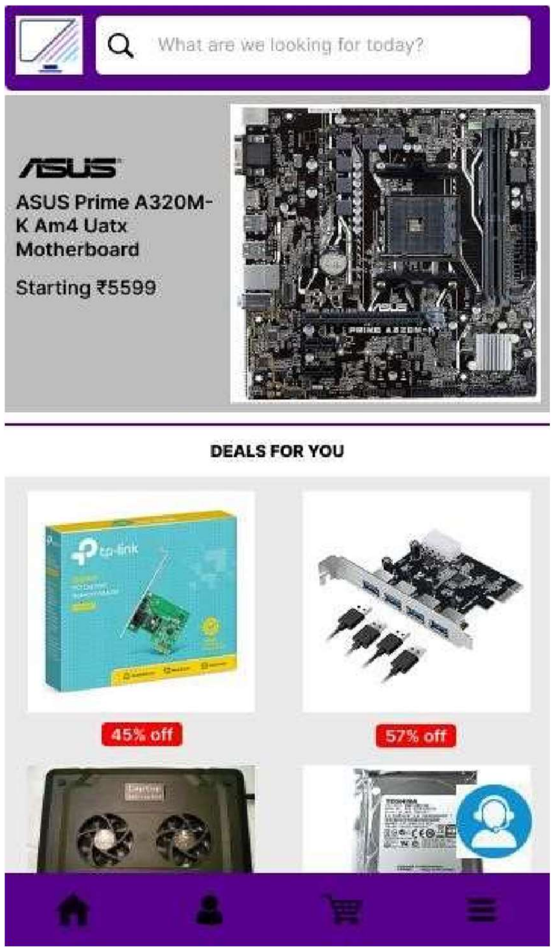
b.Login Screen

- Login screen includes these buttons:Email,password,sign in,sign up and forgot password button.
- Sign up button is connected to the register screen
- Sign in button will help the user to enter the home screen.



c.Home screen

- The home screen inludes a seach button(for searching damaged or required parts),home button (represented in house shape),service button(to navigate to service screen),cart button(to navigate to cart screen),category button.


What are we looking for today?
ASUS
ASUS Prime A320M-K Am4 Uatx Motherboard
Starting ₹5599
DEALS FOR YOU
45% off
57% off

d.Parts Screen

e.Cart Screen

f.Checkout Screen

g.Service Screen

4.2 Visual Design

- The visual design of the mobile android application should be user-friendly and easy to navigate. The overall design should be simple and contemporary, with an emphasis on use and functionality. The colour scheme should be uniform across the application and comprise of eye-pleasing, complimentary hues.
- The GUI design will use high-contrast colours and larger font sizes for better visibility and readability.
- The icons used in the application are simple and easily recognizable, with each icon representing a different function or feature of the application.
- The buttons used in the application are designed to be responsive and easy to use, with clear text labels that describe their function.

4.3 Navigation Design

- The application will use a intuitive and user-friendly navigation structure to ensure a seamless user experience.
- The navigation design has the ability to easily navigate back to previous screens or pages, either through a back button or a swipe gesture. This will allow users to easily backtrack if they accidentally navigate to the wrong screen or page.
- The application will use a Consistent navigation structure throughout.

5. API Design

The Cipherserve mobile application utilizes two main APIs: Flutter and Firebase. Flutter is a UI toolkit for building natively compiled applications for mobile, web, and desktop from a single codebase. Flutter does not have its own API, but it provides tools and packages for building APIs in Dart, the programming language used by Flutter,while Firebase is used as the back-end framework for the application.

5.1 Flutter API

With the help of the Flutter mobile app SDK (Software Development Kit), programmers can produce native apps for both the iOS and Android platforms from a single codebase. The Flutter application for Android mobile devices is intended to be quick, simple, and effective in providing information about the availability of damaged software or hardware components for a computer or laptop.In general, the Flutter application for the mobile Android application is designed to be user-friendly, effective, and aesthetically pleasing, giving users a seamless experience while using the app. It provides information about the availability of the damaged software or hardware parts of a computer or laptop and, thus, providing services.

5.2 Firebase API

A backend-as-a-service platform called Firebase offers a variety of tools and services to help mobile app developers create scalable and reliable applications. Firebase can be used to provide the backend functionality including authentication, data storage, cloud functions, and messaging in the context of the mobile android application that provides the availability of the details of the damaged software or hardware parts of a computer or laptop and thus providing services. In conclusion, Firebase offers a wide range of backend services that may be utilised to create a reliable and scalable mobile Android application that specifies the availability of the faulty software or hardware components of a computer or laptop and thus offers services.

6. Technology Stack

The Cipherserve mobile application is built using the following technology stacks:

- Front-End Development: Flutter
  Flutter is an open-source mobile application development framework created by Google. To create Android applications, Flutter employs a single codebase. It offers a wide range of pre-made widgets and utilities for creating user interfaces, controlling state, and processing user input.

- Back-End Development: Firebase
  Firebase is a mobile and web application development platform provided by Google. It provides a set of services for building, managing, and scaling mobile and web applications. Firebase includes features such as real-time databases, authentication, cloud storage, and hosting.

- Database:MySqL
  MySQL is an open-source relational database management system that is widely used for web-based applications.MySQL allows developers to store and retrieve data using a wide range of programming languages, including PHP, Java, and Python. It uses the SQL language for interacting with data, allowing developers to create, read, update, and delete data from the database.

- Cloud Services:Firebase
  Firebase is a cloud-based service that provides developers with various tools and services to build, test, and deploy mobile and web applications. It offers features like real-time databases, authentication, storage, hosting, and messaging, which can be used to build scalable and robust applications.

- Payment Gateway:Stripe
  A worldwide payment gateway called Stripe enables companies to take payments online. For secure and convenient payment processing, it offers a collection of APIs and tools. Credit cards, bank transfers, and virtual wallets are just a few of the payment options supported by Stripe.

- Third Party APIs:Google Maps API
  Google Maps API is a collection of APIs that enables programmers to include navigation, location, and map services into their applications. It offers a variety of features, including routing, instructions, geocoding, and reverse geocoding.