Project Design Phase-I
Solution Architecture

Date	16 October 2023
Team ID	Team-591008
Project Name	Money Matters
Maximum Marks	4 Marks

Solution Architecture:

- Solution architecture for the "Money Matters" project

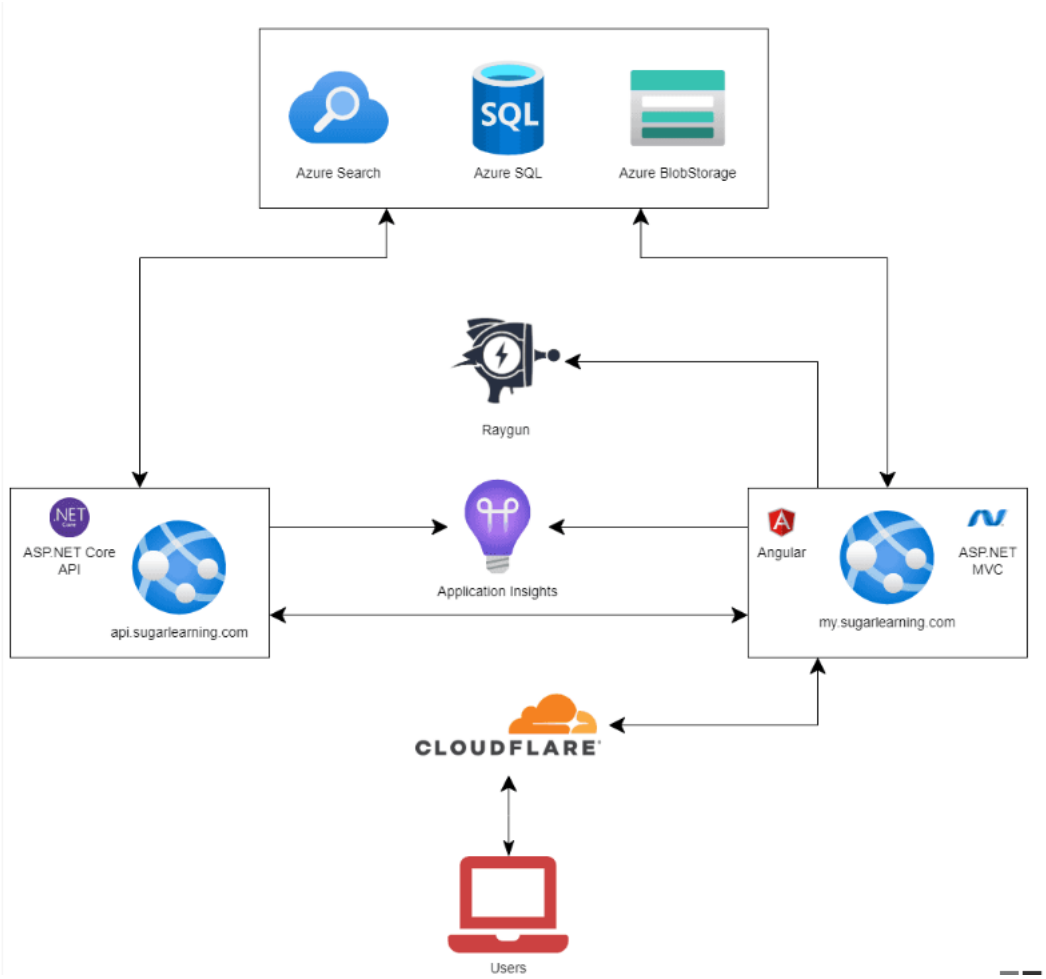
  is:(User Interface) Frontend

- The main tool for creating Android applications is Android Studio.
- Activities and Fragments: These parts are in charge of the navigation and userinterface.
- Define the visual organization of the app's screens using compose layouts.
- Buttons, text boxes, and other widgets for user interaction make up UI components.
- Server-side backend: Consider using a backend component if you wish to
- synchronize data between several devices, offer user accounts, or back up your data. This might be created using server-side frameworks like Node.js, Python,or other technologies.
- Database: Use the SQLite database to store local financial information, such astransactions, account balances, and user preferences.
- Firebase Realtime Database or Cloud Firestore: Firebase can be a fantasticoption if you require real-time updates and cloud syncing.
- User Identification: For managing user accounts, including sign-up, login, andpassword-reset features, utilize Firebase Authentication.
- Commercial Logic: To handle financial calculations, transaction categorization, budget management, and other essential app functionality, create Java or Kotlinclasses.
- To represent financial entities such as transactions, accounts, and categories,implement data models.

- Layer for Data Access: In order to communicate with the nearby SQLite database, create a data access layer. Methods for data retrieval, storage, andchange will be included in this layer.
- Network communication: If your application has a backend component, you'll need to create APIs and use tools like REST or GraphQL to communicate withthe server from the mobile app.
- Security: Encrypt critical information, including user credentials and financialtransactions.
- Securely manage user authentication.
- Interaction between user interface and business logic:
- To isolate UI from business logic, use the Model-View-Controller (MVC) orModel-View-ViewModel (MVVM) architectural patterns.
- For a more seamless connection between UI components and underlying data,use data binding.
- Testing: Perform unit testing on crucial elements like data access and financialcalculations.
- Test the user interface to make sure the app performs as planned.

  Think about frameworks for automated testing like Espresso.
- Documentation: Code, data models, and architecture documentation should bemade for future collaboration and reference.
- Deployment: Publish the Android app to the Google Play Store.

Solution Architecture Diagram:


Azure Search
SQL
Azure SQL
Azure BlobStorage
Raygun
.NET
ASP.NET Core API
api.sugarlearning.com
Application Insights
Angular
ASP.NET MVC
my.sugarlearning.com
CLOUDFLARE
Users