Tech Stack:

Backend: The API has been developed using .NET 6 (ASP.NET Core).

Authentication and Authorization: JWT (JSON Web Tokens) have been implemented for user authentication and authorization.

Database: Microsoft SQL Server has been integrated for data storage.

API Documentation: Swagger has been used for API documentation and testing.

Frontend (Web): The landing page will be developed using React to showcase app features and provide user sign-up/login options.

Frontend (Mobile): The Flutter mobile app will be designed for Android and iOS platforms.

Backend Structure:

Authentication and Authorization:

JWT-based authentication has been implemented, ensuring secure user sessions.

.NET Identity has been used for user management, roles, and claims.

Repository Pattern:

Repository interfaces and implementations have been created for various entities, such as User, Points, DocumentScan, Company, and Ad.

CRUD operations and complex queries have been implemented in the repository classes.

API Controllers:

API controllers have been developed to handle different functionalities, including user management, Points management, DocumentScan, Ads, and more.

The repository pattern has been utilized to interact with the database.

Middleware:

Custom middleware has been implemented to handle logging, error management, and request/response transformations.

Swagger Documentation:

Swagger has been integrated to provide comprehensive API documentation and testing capabilities.

Authentication Middleware:

A custom middleware has been implemented to validate JWTs and enforce user authorization.

Frontend Structure:

Web (React):

The React landing page will be developed, highlighting app features and offering user sign-up/login options.

User interfaces for ticket and check scanning, as well as viewing user points, will be designed.

Mobile (Flutter):

The Flutter mobile app will be developed for both Android and iOS platforms.

Features for scanning tickets, checks, and viewing user points will be implemented.

UI components will be designed for displaying advertisements and products.

Hosting and Deployment:

Ubuntu Server 20.04 will be set up and configured on the hosting provider.

Nginx will act as a reverse proxy to manage routing for the .NET Core application.

Systemd will be employed to manage the .NET Core app as a service.

Endpoints will be secured with HTTPS through SSL certificates like Let's Encrypt.

Nginx will be configured to serve the React landing page and proxy API requests to the .NET Core app.

Development Workflow:

Backend Development:

The API has been developed in .NET 6, encompassing API controllers, repositories, and middleware.

JWT-based authentication and authorization mechanisms have been implemented.

Microsoft SQL Server has been integrated using Entity Framework Core for data management.

Frontend Development:

The React landing page will be crafted, featuring user authentication and registration functionality.

The Flutter app will be developed with screens for ticket and check scanning, and for viewing user points.

UI components will be designed for displaying advertisements and products.

Testing:

API endpoints have been thoroughly tested using tools like Postman and Swagger.

Frontend components will be tested to ensure seamless user interactions and responsiveness.

Deployment:

The .NET Core API will be deployed on the Ubuntu Server.

Nginx will handle routing and serve the React landing page.

The Flutter app will be deployed to both Google Play Store and Apple App Store.