Here is a development plan for a SaaS microservice project for Insurance brokers, based on the provided features and technologies:

Initial Stage - **Infrastructure and Core Services**

In this phase, we will establish the basic infrastructure and core services required for the project.

* Choose the appropriate cloud provider, such as AWS, and set up the necessary infrastructure.
* Set up a Kubernetes cluster for container orchestration and management.
* Implement an API Gateway using Nginx to handle incoming requests.
* Choose a message broker such as Apache Kafka or RabbitMQ and implement a message-based communication pattern between services.
* Set up Docker for containerization of microservices.
* Implement Mass Transit to handle messaging between microservices.

Phase 1: **Minimum Viable Product (MVP)**

1. Automated Underwriting Microservice

* Develop automated underwriting microservice using C#, Docker, and Kubernetes.
* Integrate with Apache Kafka or RabbitMQ for messaging and event-driven architecture.
* Utilize AWS services for scalability, security, and reliability.
* Develop simple UI for manual underwriting.

1. Document Management Microservice

* Develop document management microservice using C#, Docker and Kubernetes.
* Utilize Mass Transit as the messaging framework.
* Integrate with AWS S3 for storage.
* Develop a simple UI for document upload and retrieval.

1. Payment Gateway Microservice

* Develop payment gateway microservice using C#, Docker and Kubernetes.
* Integrate with AWS Payment Gateway.
* Implement fraud prevention measures.
* Develop a simple UI for payment processing.

1. Self-Service Portal

* Develop self-service portal using React/Typescript.
* Utilize Nginix as a reverse proxy.
* Integrate with the previously developed microservices for underwriting, document management, and payment gateway.
* Implement multi-language and localization support.

Phase 2: Enhancements

1. Notification Service Microservice

* Develop notification service microservice using C#, Docker and Kubernetes.
* Utilize AWS SNS for messaging.
* Implement different notification channels (email, SMS, etc.).
* Integrate with the previously developed microservices.

1. Blockchain Smart Contract Microservice

* Develop blockchain smart contract microservice using C#, Docker and Kubernetes.
* Utilize Ethereum blockchain network.
* Implement smart contracts for insurance policies and claims.
* Integrate with the previously developed microservices.

1. Claim Processing Microservice

* Develop claim processing microservice using C#, Docker and Kubernetes.
* Utilize AWS Lambda for serverless processing.
* Integrate with the previously developed microservices.
* Develop a simple UI for claim submission and tracking.

1. Mobile App

* Develop a mobile app using React Native.
* Utilize the previously developed microservices for underwriting, document management, payment gateway, and claim processing.
* Implement AI-powered chatbot for customer support.

Phase 3: Advanced Features

1. Customizable Workflow Microservice

* Develop customizable workflow microservice using C#, Docker and Kubernetes.
* Utilize AWS Step Functions for workflow orchestration.
* Integrate with the previously developed microservices.

1. Advance Analytics and Reporting Microservice

* Develop advance analytics and reporting microservice using C#, Docker and Kubernetes.
* Utilize AWS Athena and QuickSight for data analysis and visualization.
* Integrate with the previously developed microservices.

1. CRM Microservice

* Develop CRM microservice using C#, Docker and Kubernetes.
* Utilize AWS DynamoDB for storage.
* Integrate with the previously developed microservices.

1. External Integration Microservice

* Develop external integration microservice using C#, Docker and Kubernetes.
* Provide an API Gateway using AWS API Gateway.
* Enable third-party integration with insurance companies as an extension to add to the app by the user.

This development plan follows an iterative approach, with each phase building upon the previous phase. It starts with developing the MVP, which includes the core features of the application. In the second phase, enhancements are made to the existing microservices to add more functionality. Finally, in the third phase, advanced features are added to the application to provide a complete solution for insurance brokers