**Architecture Overview**

The solution will be built using a **microservices** architecture, ensuring flexibility, scalability, and isolation of concerns. Each component of the system will be loosely coupled and independently deployable. Azure services such as Azure Kubernetes Service (AKS), Azure Functions, Azure API Management, and Azure CDN will be used to meet the requirements.

**1. Frontend/UI Components**

* **Web UI**: HTML5, CSS3, JavaScript (React.js, Vue.js, or Angular)
* **Mobile App**: (iOS/Android) built with React Native or Flutter
* **Communication Channels**: Integration with SMS and Email providers (Twilio, SendGrid)

**2. Backend Services (Microservices)**

* **Authentication and Authorization**: Managed via **Azure AD B2C** for user login and registration.
* **Conference Service**: Manages conference creation, schedule, and themes.
* **Session Service**: Manages individual session details and presentation uploads.
* **Voting Service**: Handles attendee votes via SMS, email, web, and phone.
* **Moderation Service**: Manages session moderation, speaker ratings, and voting results.
* **CDN Service**: Handles video streaming and media delivery through **Azure CDN**.

**3. Data Storage**

* **Azure SQL Database**: Stores relational data such as user profiles, conference schedules, speaker details, and votes.
* **Azure Cosmos DB**: Used for unstructured data, session feedback, and real-time analytics.
* **Blob Storage**: Used for storing media files such as slides, presentations, and conference video content.

**4. APIs and Services**

* **RESTful APIs** for CRUD operations on conference entities (attendees, speakers, sessions).
* **Azure Functions**: For real-time event handling (e.g., voting, session updates).
* **SignalR**: For real-time communication (like live voting updates or moderator interactions).
* **Azure Event Grid**: For event-driven architecture and handling real-time events (e.g., new votes, session updates).

**5. Scalability & High Availability**

* **Azure Kubernetes Service (AKS)**: Deploy backend microservices in scalable containers.
* **Azure Load Balancer**: Distributes traffic among the services.
* **Azure Application Gateway**: Used for web traffic routing and enhanced security.
* **Azure Traffic Manager**: Global distribution for redundancy and fast access for users worldwide.

**6. Analytics and Monitoring**

* **Azure Monitor**: For performance and health monitoring.
* **Azure Log Analytics**: For analyzing logs and real-time diagnostics.
* **Azure Application Insights**: For tracking user interactions and system performance.

**Detailed Component Breakdown**

1. **Authentication & User Management**:
   * **Azure Active Directory B2C** for user registration, login, and role-based authentication (attendee, admin, speaker, moderator).
   * Support for social login (Google, Facebook, etc.) and email/password authentication.
2. **Conference and Session Management**:
   * **Conference Service**: CRUD operations to create, update, and delete conferences, including the ability to set conference branding (background, logos).
   * **Session Service**: CRUD operations to create, update, delete, and schedule sessions.
   * **Moderation Service**: Moderators can change session order, remove sessions, or adjust the schedule dynamically.
3. **Voting & Feedback System**:
   * Multiple voting channels: Web, SMS, Email.
   * **Voting Service** will handle storing votes, real-time aggregation, and triggering actions when a vote threshold is reached.
   * **Real-time** voting results pushed via SignalR to the client application.
4. **Content Delivery Network (CDN)**:
   * **Azure CDN** will handle static content (slides, video streams, and images) for better global performance.
   * Dynamic content can be cached and served from the nearest Azure CDN edge location to minimize latency.
5. **Real-Time Communication**:
   * **Azure SignalR** for real-time updates such as live vote count, session changes, or new speaker announcements.
   * Integration with Azure Event Grid for event-driven interactions across services (e.g., attendee votes trigger immediate updates).

**Deployment to Azure**

1. **Azure Kubernetes Service (AKS)**:
   * Microservices (Auth, Conference, Session, Voting, Moderation) will be deployed as containers in AKS, ensuring auto-scaling based on load.
2. **Azure SQL Database**:
   * Relational data such as user profiles, conferences, and sessions will be stored in Azure SQL Database with replication for high availability.
3. **Azure Blob Storage**:
   * Media files such as conference videos, presentations, and other assets will be stored in Azure Blob Storage.
4. **Azure CDN**:
   * Media and static files will be cached and delivered from the Azure CDN to improve performance.
5. **Event-Driven Architecture**:
   * **Azure Event Grid** and **Azure Functions** will be used to handle asynchronous events, ensuring a responsive system even with a large number of concurrent users.

**Security Considerations**

1. **Data Encryption**: All sensitive data (e.g., user information, votes) will be encrypted at rest using Azure encryption mechanisms.
2. **Role-Based Access Control (RBAC)**: Azure Active Directory (AD) will handle user roles, ensuring proper access levels for admins, moderators, and other users.
3. **Network Security**: Utilize **Azure Virtual Networks**, **Network Security Groups**, and **Application Gateway** for secure communication between services.

**Monitoring & Maintenance**

* Use **Azure Monitor** for application performance insights, **Log Analytics** for troubleshooting, and **Application Insights** to track user behavior and interactions.
* Implement auto-scaling based on load and use **Azure Traffic Manager** for global traffic distribution.

