Project Roadmap: Real-time Facebook Voting System with CasparCG

# 🎯 Goal

Let registered Facebook users vote during live shows via comments. Show results live on-screen via CasparCG. Ensure security, performance, and scalability (target: 100K+ votes per session).

# 🔷 PHASE 1: FOUNDATION — Define, Choose, Prepare

## 1. Define System Requirements

- Live Facebook comment ingestion  
- Vote processing with custom rules (e.g., first vote only)  
- Registered users only (optional)  
- Show real-time results via CasparCG  
- Send user feedback ('Vote accepted', 'Already voted', etc.)  
- Scale to 100K+ votes

## 2. Choose Technologies

Backend: .NET 8 (ASP.NET Core)  
Real-Time Messaging: SignalR  
Queue: Redis  
Database: PostgreSQL  
Frontend Admin Panel: React or Blazor (optional)  
CasparCG Integration: TCP/AMCP client in C#  
Facebook API: Graph API v19+ (Comments, Pages, Tokens)

# 🔷 PHASE 2: SYSTEM DESIGN — Blueprint & Data Models

## 3. Design Architecture

Split into:  
- Facebook Comment Listener  
- Vote Processor + Rule Validator  
- User Feedback Sender  
- CasparCG Result Publisher  
- Admin Panel (manual control)  
- Database & Logging Layer

## 4. Define Core Data Models

- User: Facebook ID, DisplayName, IsRegistered  
- Vote: UserId, QuestionId, Option, Timestamp  
- Question: Id, Text, Options[], IsActive, TimeWindow  
- VoteResult: QuestionId, OptionA%, OptionB%...  
- FeedbackLog: UserId, Message, SentAt

# 🔷 PHASE 3: DEVELOPMENT — Build System Components

Step 1: Setup Core Project  
- Create ASP.NET Core WebAPI project (.NET 8)  
- Add Redis and PostgreSQL  
- Setup logging with Serilog  
  
Step 2: Facebook Comment Listener  
- Use Facebook Graph API to poll live video comments  
- Push comments into Redis queue  
  
Step 3: Vote Processor + Rule Engine  
- Consume from Redis  
- Validate rules and store valid votes in DB  
  
Step 4: Feedback System  
- Send replies using Facebook API  
- Maintain feedback logs  
  
Step 5: CasparCG Integration  
- Push vote results to CasparCG using TCP client  
  
Step 6: Admin Panel (Optional)  
- Start/stop questions, view stats, manage users

# 🔷 PHASE 4: INFRASTRUCTURE — Hosting & Deployment

Step 7: Prepare Hosting Environment  
- Use Docker for backend, Redis, CasparCG client  
- Deploy on-premise or in cloud  
  
Step 8: Monitoring & Recovery  
- Use health checks, logs, and queue persistence

# 🔷 PHASE 5: TESTING — Simulate Load & Iterate

Step 9: Simulated Load Testing  
- Simulate votes and monitor performance  
  
Step 10: Go-Live Checklist  
- Facebook App ready  
- Page token valid  
- Backup plan in place

# ✅ Bonus: Features to Add Later

- Auto-ban for spam users  
- Chat-bot for feedback  
- Real-time charts  
- Multilingual UI support