ShipSpeak Phase 2 Phase Summary

Backend Integration & Production Infrastructure

Version: 1.0
Date: November 4, 2025
Phase: Phase 2 (Real Backend Integration)

Executive Overview

Phase 2 transforms ShipSpeak from a validated frontend prototype into a production-ready platform with real backend integrations. This phase maintains the **identical user interface** validated in Phase 1 while systematically replacing all mock data sources with live production services. The result is a fully functional platform that analyzes actual meetings, generates authentic AI feedback, and provides genuine coaching value to Product Managers.

The Transformation

Phase 1 Delivered: Complete user experience with mock data that validated product concept and user flows
Phase 2 Delivers: Same user experience powered by real authentication, meeting bots, transcription, AI analysis, and database persistence

Critical Success Factor

UI must remain unchanged from Phase 1. Users who participated in beta testing should not notice any visual differences—only that their real meetings are now being analyzed and their practice recordings are generating actual feedback.

Phase 2 Objectives

Primary Goal

Enable production usage with real users, real meetings, and paid subscriptions by replacing every mock integration point with corresponding production service.

Duration & Effort

- **Total Duration:** 4 weeks
- **Total Development Hours:** 36-44 hours
- **Total Slices:** 13 integration points
- **Team Size:** 1 developer (can be parallelized with 2)

Outcome Statement

By the end of Phase 2, the platform will:

- Authenticate users securely through Supabase
- Deploy meeting bots that join real Zoom/Google Meet/Teams meetings
- Transcribe audio with speaker attribution using Deepgram

- Generate AI-powered coaching feedback through OpenAI
- Process practice recordings with immediate feedback
- Persist all data reliably in PostgreSQL
- Update dashboards in real-time as processing completes
- Handle errors gracefully with user notifications
- Support paying customers at scale

What Gets Built

Infrastructure Foundation

✅ Production authentication (Supabase Auth)

✅ PostgreSQL database with complete schema

✅ Row-level security policies

✅ Session management with token refresh

✅ Protected route middleware

Meeting Intelligence Pipeline

✅ Calendar OAuth connections (Google, Microsoft, Zoom)

✅ Automated bot scheduling based on user criteria

✅ Meeting bot deployment and lifecycle (Recall.ai)

✅ Real-time audio capture with speaker separation

✅ Smart exit rules (participant-based, keyword-based, time-based)

✅ Audio transcription with Deepgram

✅ Comprehensive AI analysis with OpenAI GPT-4

✅ Structured feedback with scores, patterns, key moments

Practice & Learning System

✅ Browser audio recording and upload

✅ Practice transcription pipeline (Deepgram)

✅ Criteria-based AI feedback (OpenAI)

✅ Progress tracking across attempts

✅ Module completion persistence

Production Polish

✅ Real-time status updates (Supabase subscriptions)

✅ Comprehensive error handling and recovery

✅ Performance optimization (caching, indexing, job queuing)

✅ Monitoring and observability

Week-by-Week Breakdown

Week 1: Authentication & Data Foundation (Phase 2A)

Duration: 8-10 hours | **Slices:** 16-18

What Gets Built

- Supabase authentication replacing mock localStorage
- Complete database schema with all tables
- User profile and settings persistence
- Bot configuration storage

Key Milestones

- Users can sign up and log in with real credentials
- Sessions persist across browser restarts
- Password reset flow works end-to-end
- All user data saves to database
- RLS policies enforce data privacy

Validation Criteria

- □ Can create account with email/password
- □ Login persists across sessions
- □ Protected routes redirect properly
- □ Onboarding data saves to profiles table
- □ Bot configuration persists to database
- □ No localStorage usage for user data

Dependencies Established

This week establishes the foundation for all subsequent integrations. Every later slice depends on auth working correctly and database schema being complete.

Week 2-3: Meeting Intelligence Integration (Phase 2B)

Duration: 12-15 hours | **Slices:** 19-22

What Gets Built

- Calendar integrations (Google Calendar, Outlook, Zoom)
- Meeting bot scheduling and deployment (Recall.ai)
- Audio transcription pipeline (Deepgram)
- AI analysis and feedback generation (OpenAI)

Detailed Flow

1. **Calendar Monitoring:** Background job polls calendars every 15 minutes
2. **Bot Scheduling:** Matches meetings against user criteria, creates scheduling records

3. **Bot Deployment:** Recall.ai dispatches bot 5 minutes before meeting
4. **Meeting Attendance:** Bot joins with configured identity, captures audio
5. **Exit Evaluation:** Monitors exit rules throughout meeting
6. **Audio Handoff:** Bot leaves, uploads audio, sends webhook
7. **Transcription:** Deepgram processes audio with speaker diarization
8. **Analysis:** OpenAI generates comprehensive feedback
9. **Storage:** All data persists to database with relationships

Key Milestones

- Calendar OAuth connections working
- Bot joins first real meeting successfully
- Transcript generates with speaker attribution
- AI feedback appears in UI (all 7 sections)
- Exit rules trigger correctly

Validation Criteria

- ☐ Calendar permissions grant successfully
- ☐ Bot appears in meeting with configured name
- ☐ Audio captures all participants clearly
- ☐ Exit rule triggers when CEO joins (test case)
- ☐ Transcript displays in UI within 5 minutes
- ☐ Feedback generates with specific examples
- ☐ All 7 feedback sections populate correctly
- ☐ Next Steps are actionable and personalized

Integration Challenges

- **Bot admission:** Some meetings require host approval—bot waits in lobby
- **Speaker identification:** Voice fingerprinting may need adjustment per user
- **Transcription accuracy:** Technical PM vocabulary requires custom dictionary
- **Analysis quality:** Prompt engineering for career-relevant feedback

---

Week 3: Practice & Learning Integration (Phase 2C)

Duration: 8-10 hours | **Slices:** 23-25

What Gets Built

- Practice recording upload to Supabase Storage
- Practice transcription optimized for single speaker
- Practice feedback generation with annotations
- Progress tracking across attempts

Detailed Flow

1. **Recording:** Browser captures audio via MediaRecorder API
2. **Upload:** Audio blob uploads to Supabase Storage (user-specific bucket)
3. **Transcription:** Deepgram processes with speed optimization
4. **Analysis:** OpenAI evaluates against exercise criteria

5. **Feedback:** Annotated transcript with strengths/improvements
6. **Progress:** Scores and attempt history update

Key Milestones

- Practice recording uploads successfully
- Transcription completes in <10 seconds
- Feedback displays with inline annotations
- Multiple attempts show improvement trajectory

Validation Criteria

- ☐ Recording captures clear audio
- ☐ Upload progress displays accurately
- ☐ Transcription appears within 10 seconds
- ☐ Feedback includes specific annotations
- ☐ Expert example comparison works
- ☐ Progress chart shows improvement
- ☐ Can re-record unlimited times

Performance Targets

- **Transcription latency:** <3 seconds average
- **Full feedback generation:** <10 seconds total
- **Upload reliability:** >99% success rate

Week 4: Real-Time Updates & Polish (Phase 2D)

Duration: 8-10 hours | **Slices:** 26-28

What Gets Built

- Supabase real-time subscriptions for status updates
- Comprehensive error handling with user notifications
- Performance optimizations (caching, indexing, job queuing)
- Monitoring dashboards for system health

Real-Time Update Points

- Meeting status changes (scheduled → in_progress → processing → completed)
- Transcript generation completion
- Feedback analysis completion
- Practice session completion
- Progress metric updates

Error Scenarios Handled

1. **Bot Failures:** Wrong meeting link, admission denied, network disconnection
2. **Transcription Errors:** Poor audio quality, API timeout, invalid format
3. **Analysis Failures:** OpenAI rate limits, token limit exceeded, API downtime
4. **Upload Issues:** Network interruption, storage quota exceeded

Performance Optimizations

- Database indexes on frequently queried columns
- API response caching for completed feedback
- Background job queuing for async operations
- Query result pagination for long transcripts

Key Milestones

- Dashboard updates without page refresh
- Error messages are user-friendly
- Retry logic recovers from transient failures
- Queries remain fast with growing data

Validation Criteria

- □ Meeting list updates live as meetings complete
- □ No page refresh needed for new feedback
- □ Error messages explain what happened
- □ Failed operations offer retry actions
- □ Dashboard loads in <2 seconds
- □ Transcript scrolling is smooth
- □ Background jobs complete reliably

Technical Architecture

Service Integration Map

```
┌─────────────────────────────────────────────────┐
│              User Interface                │
│      (Next.js 14 - No Changes from Phase 1)        │
└─────────────────────────────────────────────────┘
                  ↓↑
┌─────────────────────────────────────────────────┐
│              Vercel API Routes                  │
│       (Orchestration & Business Logic)             │
└─────────────────────────────────────────────────┘
      ↓↑        ↓↑        ↓↑        ↓↑
┌───────────┐ ┌───────────┐ ┌───────────┐ ┌───────────┐
│  Supabase  │ │ Recall.ai │ │  Deepgram  │ │  OpenAI   │
│         │ │         │ │         │ │         │
│ • Auth      │ │ • Bot Deploy │ │ • Meeting    │ │ • Meeting    │
│ • PostgreSQL │ │ • Audio      │ │  Transcript │ │  Analysis   │
│ • Storage    │ │  Capture     │ │ • Practice   │ │ • Practice   │
│ • Real-time  │ │ • Webhooks   │ │  Transcript │ │  Feedback    │
└───────────┘ └───────────┘ └───────────┘ └───────────┘
```

Data Flow Examples

Meeting Analysis Flow

```
User connects calendar → Calendar polls for meetings → Bot scheduled
    ↓
Bot joins meeting → Records audio → Evaluates exit rules
    ↓
Meeting ends → Audio upload → Webhook fires
    ↓
Deepgram transcription → Database storage → Real-time update
    ↓
OpenAI analysis → Feedback generation → Database storage
    ↓
User dashboard updates → Notification sent → User reviews feedback
```

Practice Flow

```
User starts exercise → Browser records audio → Upload to Supabase Storage
    ↓
Deepgram transcription (single-speaker optimized)
    ↓
OpenAI feedback (criteria-based evaluation)
    ↓
Annotated transcript + scores → Database storage
    ↓
UI updates with feedback (<10 seconds total)
```

Database Schema Overview

Core Tables

User Management:

- `profiles` - User profile data and career information
- `bot_configs` - Meeting bot configuration and preferences

Meeting Intelligence:

- `meetings` - Meeting metadata and status
- `transcript_segments` - Individual speaker turns
- `meeting_feedback` - AI analysis and scores

Learning & Practice:

- `modules` - Learning module definitions
- `exercises` - Practice exercise specifications
- `practice_sessions` - User practice attempts
- `module_progress` - Learning completion tracking

Progress Analytics:

- `progress_snapshots` - Daily aggregated metrics

Key Relationships

- `profiles.user_id` → `auth.users.id`
- `meetings.user_id` → `profiles.id`
- `transcript_segments.meeting_id` → `meetings.id`
- `meeting_feedback.meeting_id` → `meetings.id` (one-to-one)
- `practice_sessions.user_id` → `profiles.id`
- `practice_sessions.exercise_id` → `exercises.id`

Row-Level Security

Every table implements RLS policies ensuring users can only access their own data. Policies check `auth.uid()` against `user_id` columns.

Success Metrics

Technical Reliability

- **Bot Join Success Rate:** >95%
- **Transcription Completion Time:** <5 minutes per hour of audio
- **Analysis Generation Time:** <2 minutes per meeting
- **Practice Feedback Latency:** <10 seconds end-to-end
- **System Uptime:** >99%
- **Error Rate:** <5% across all integrations

User Experience

- **Meeting Analysis Engagement:** 50%+ of users analyze meetings weekly
- **Practice Exercise Completion:** 30%+ complete exercises monthly
- **Skill Improvement:** Measurable score increases over 3 months
- **Feature Satisfaction:** Positive feedback on bot discretion
- **Session Duration:** Average 15+ minutes per visit

Business Viability

- **Conversion Rate:** 20%+ trial to paid conversion
- **Churn Rate:** <10% monthly
- **Time to First Value:** <24 hours (first meeting analyzed)
- **User Retention:** 60%+ 90-day retention

Risk Mitigation

Integration Risks

Risk: Recall.ai bot fails to join meetings
Mitigation: Implement retry logic, provide manual meeting link sharing, log all failures with context
Fallback: Allow users to upload their own recordings

Risk: Deepgram transcription accuracy issues
Mitigation: Custom vocabulary for PM terminology, confidence score thresholds, option to edit transcripts
Fallback: Display original audio with playback controls

Risk: OpenAI rate limits or API downtime
Mitigation: Exponential backoff, job queuing, cached results for re-analysis
Fallback: Queue for off-peak processing, notify users of delay

Risk: Calendar OAuth permissions revoked
Mitigation: Token refresh handling, re-authorization prompts, graceful degradation
Fallback: Manual meeting scheduling through bot interface

Data Privacy Risks

Risk: Meeting audio containing sensitive information
Mitigation: User-controlled bot identity, smart exit rules, audio encryption at rest
Fallback: Immediate audio deletion option, no cloud storage mode

Risk: Unauthorized access to transcripts
Mitigation: Row-level security policies, encrypted connections, audit logging
Fallback: Emergency data deletion procedures

Performance Risks

Risk: Database queries slow down with data accumulation
Mitigation: Indexes on critical paths, query pagination, caching layer
Fallback: Database read replicas, query optimization

Risk: Real-time updates causing connection overload
Mitigation: Connection pooling, selective subscriptions, debouncing
Fallback: Polling fallback for unsupported browsers

Testing Strategy

Unit Testing

- Test each API endpoint independently
- Mock external service calls (Recall.ai, Deepgram, OpenAI)
- Validate data transformations and business logic
- Test error handling for each failure scenario

Integration Testing

- Test service-to-service communication
- Validate webhook processing
- Test database transactions and rollbacks
- Verify real-time subscription behavior

End-to-End Testing

- Complete user journeys from signup to feedback
- Path A (meeting-first) and Path B (practice-first)
- Multi-device testing (desktop, mobile, tablet)
- Cross-browser compatibility (Chrome, Safari, Firefox)

Load Testing

- Simulate 50 concurrent users
- Test bot scheduling at scale (100+ meetings per day)
- Database query performance under load
- API rate limit handling

Resilience Testing

- Deliberately fail each external service
- Test network interruptions mid-process
- Validate retry and recovery mechanisms
- Confirm user notifications work correctly

Launch Preparation

Pre-Launch Checklist

Infrastructure:

- □ Production environment configured
- □ Environment variables secured
- □ API keys rotated to production values
- □ Database backups automated
- □ Monitoring dashboards configured
- □ Error alerting set up

Security:

- □ RLS policies tested thoroughly
- □ API authentication verified
- □ HTTPS enforced everywhere
- □ Session timeout configured
- □ Rate limiting enabled
- □ Audit logging active

Data:

- □ Phase 1 beta users migrated
- □ Mock data cleared from production
- □ Database migrations version controlled
- □ Rollback procedures documented

Support:

- □ Help documentation written
- □ Common issues troubleshooting guide
- □ Support email configured
- □ Internal knowledge base complete

Phased Rollout Strategy

Phase 1: Beta User Migration (Week 1)

- Migrate 10 Phase 1 beta users
- Validate all integrations with real usage
- Gather feedback on new real-data experience
- Fix critical issues before broader launch

Phase 2: Controlled Launch (Weeks 2-3)

- Direct outreach to 25-50 target users
- Monitor system health closely
- Iterate on AI prompt quality
- Build case studies and testimonials

Phase 3: Public Launch (Week 4+)

- Open signup with broader marketing
- Scale infrastructure as needed
- Continue feature refinement
- Expand to 100+ active users

Success Criteria for Each Phase

Beta Migration Success:

- All 10 users successfully analyze real meetings
- No data loss from migration
- Bot join rate >90%
- Feedback quality meets expectations

Controlled Launch Success:

- 50%+ of new users complete onboarding
- 30%+ analyze first meeting within 48 hours
- <5% error rate across integrations
- Positive qualitative feedback

Public Launch Success:

- 100+ active users within first month
- 20%+ trial to paid conversion
- <10% monthly churn
- System remains stable under load

Post-Phase 2 Roadmap (Phase 3)

Immediate Priorities (Weeks 5-6)

- Payment integration (Stripe)
- Enhanced onboarding flow based on feedback
- Additional learning modules
- Mobile app development begins

Short-Term Enhancements (Weeks 7-10)

- Fine-tune AI prompts based on user feedback
- Expand meeting platform support (Webex, Slack Huddles)
- Community features (peer learning, discussion forums)
- Advanced analytics dashboard

Medium-Term Goals (Months 4-6)

- Team accounts and admin features
- Integration with product management tools (Jira, Linear)
- Custom coaching programs
- API for third-party integrations

Dependencies & Prerequisites

External Accounts Required

- Supabase project (with database, auth, storage, real-time)
- Recall.ai API account with bot deployment credits
- Deepgram API account with transcription credits
- OpenAI API account with GPT-4 access
- Vercel hosting account
- Google Cloud Console (for calendar OAuth)
- Microsoft Azure (for Outlook OAuth)
- Zoom Developer account (for Zoom OAuth)

Environment Setup

- Node.js 18+ installed
- Next.js 14 project configured
- TypeScript compilation working
- Tailwind CSS build pipeline
- Environment variables configured
- Database connection tested

Skills Required

- Next.js/React development
- PostgreSQL and SQL
- REST API integration
- OAuth 2.0 flows
- Webhook handling
- Real-time subscriptions
- Error handling patterns
- Performance optimization

Time Estimates by Experience

Experienced Full-Stack Developer:

- Phase 2A: 8 hours
- Phase 2B: 12 hours
- Phase 2C: 8 hours
- Phase 2D: 8 hours
- **Total:** 36 hours

Mid-Level Developer:

- Phase 2A: 10 hours

- Phase 2B: 15 hours
- Phase 2C: 10 hours
- Phase 2D: 9 hours
- **Total:** 44 hours

With Documentation/Learning Time:

- Add 20-30% for API documentation reading
- Add 10-20% for debugging integration issues
- **Realistic Total:** 45-60 hours

Appendix: Quick Reference

API Endpoints Created

Authentication:

- `POST /api/auth/signup` - Create new user
- `POST /api/auth/login` - Authenticate user
- `POST /api/auth/reset-password` - Request password reset
- `POST /api/auth/logout` - End session

Profile:

- `GET /api/profile` - Get current user profile
- `PUT /api/profile` - Update user profile
- `GET /api/bot-config` - Get bot configuration
- `POST /api/bot-config` - Update bot configuration

Meetings:

- `GET /api/meetings` - List user's meetings
- `GET /api/meetings/:id` - Get meeting details
- `GET /api/meetings/:id/transcript` - Get transcript
- `GET /api/meetings/:id/feedback` - Get feedback
- `POST /api/meetings/schedule` - Manually schedule bot

Calendar:

- `POST /api/calendar/connect` - Initiate OAuth flow
- `GET /api/calendar/callback` - Handle OAuth callback
- `GET /api/calendar/status` - Check connection status
- `DELETE /api/calendar/disconnect` - Revoke permissions

Practice:

- `POST /api/practice/sessions` - Create practice session
- `GET /api/practice/sessions/:id` - Get session details
- `GET /api/practice/sessions/:id/feedback` - Get feedback

Webhooks:

- `POST /api/webhooks/recall` - Receive Recall.ai notifications
- `POST /api/webhooks/deepgram` - Receive transcription updates

Environment Variables

bash

```
# Supabase
NEXT_PUBLIC_SUPABASE_URL=https://your-project.supabase.co
NEXT_PUBLIC_SUPABASE_ANON_KEY=your-anon-key
SUPABASE_SERVICE_ROLE_KEY=your-service-role-key

# Recall.ai
RECALL_API_KEY=your-recall-api-key
RECALL_WEBHOOK_SECRET=your-webhook-secret

# Deepgram
DEEPGRAM_API_KEY=your-deepgram-api-key

# OpenAI
OPENAI_API_KEY=your-openai-api-key

# OAuth Providers
GOOGLE_CLIENT_ID=your-google-client-id
GOOGLE_CLIENT_SECRET=your-google-client-secret
MICROSOFT_CLIENT_ID=your-microsoft-client-id
MICROSOFT_CLIENT_SECRET=your-microsoft-client-secret
ZOOM_CLIENT_ID=your-zoom-client-id
ZOOM_CLIENT_SECRET=your-zoom-client-secret

# Application
NEXT_PUBLIC_APP_URL=https://app.shipspeak.com
WEBHOOK_BASE_URL=https://app.shipspeak.com/api/webhooks
```

Key Configuration Values

Recall.ai Bot Settings:

- Audio quality: High (48kHz)
- Speaker diarization: Enabled
- Video: Disabled
- Reconnection attempts: 3
- Max duration: 180 minutes

Deepgram Settings:

- Model: Nova (latest)
- Language: en-US
- Punctuation: Enabled
- Diarization: Enabled
- Custom vocabulary: Product management terms

OpenAI Settings:

- Model: gpt-4-turbo
- Temperature: 0.7 (analysis), 0.3 (scoring)
- Max tokens: 4000
- Response format: JSON (structured)

Conclusion

Phase 2 represents the critical transition from validated prototype to production platform. By maintaining the exact user interface from Phase 1 while systematically replacing mock data with real integrations, we minimize risk while maximizing the value of our Phase 1 validation work.

The phased rollout strategy ensures we catch issues early with beta users before scaling to broader audiences. Comprehensive error handling and monitoring give us confidence in system reliability. The technical architecture is designed for the first 100-500 users, with clear paths to scale as the platform grows.

Upon Phase 2 completion, ShipSpeak will be ready to serve paying customers, generate authentic testimonials, and begin the journey toward sustainable growth and impact on Product Manager career development.

Document Version: 1.0
Last Updated: November 4, 2025
Next Review: Upon Phase 2 completion