WebFrenzy 2026 - AI-Enhanced Customer Support Ticket System

Functional Requirements Document (Updated v2.1)

---

1. User Authentication & Authorization

1.1 Login System

- **Email/Password Authentication**: Users can sign up and log in using email and password
- **Google OAuth Integration**: Alternative login method via Google account
- **Session Management**: Users remain logged in after page refresh
- **Access Control**: Unauthorized users are redirected to login page

1.2 User Roles & Team Structure

Customer Role

- Can create, view, and edit their own tickets
- View ticket timeline and customer-facing comments
- No team assignment

Support Team Structure

Super Admin (Head of Support)

- Name: **Support Director** or **Ticket Manager**
- Can view all tickets across all teams
- Approves/modifies AI suggestions
- Assigns tickets to appropriate teams
- Has override permissions for all tickets
- Access to system-wide analytics

Team 1: Technical Support Team

- **Team Name**: Technical Support
- **Team Lead**: Technical Support Lead
- **Team Members**: Technical Support Agent
- **Handles**: Technical issues, bugs, system errors, integration problems, API issues

Team 2: Billing Support Team

- **Team Name**: Billing Support

- **Team Lead**: Billing Support Lead
- **Team Members**: Billing Support Agent
- **Handles**: Payment issues, invoicing, subscription management, refunds, pricing questions

Team 3: General Support Team

- **Team Name**: General Support (or Customer Success)
- **Team Lead**: General Support Lead
- **Team Members**: General Support Agent
- **Handles**: Account questions, general inquiries, feature requests, onboarding help

---

2. Ticket Management (Customer Side)

2.1 Create Ticket

- **Required Fields**:
    - Subject (non-empty validation, min 5 characters)
    - Description (detailed issue explanation, min 10 characters)
    - Category: **Technical**, **Billing**, **General** (required selection)
    - Priority: Low, Medium, High (required selection)
- **Auto-generation**: Timestamp captured at creation
- **Default Status**: "Open"
- **Auto-assigned**: No team assignment until Super Admin approval

2.2 Edit Ticket

- Customers can modify their ticket details (Subject, Description, Category, Priority)
- Edit history is tracked in timeline
- Editing restricted once ticket is marked "Resolved"

2.3 View Ticket

- Customers can view their submitted tickets
- Access to full ticket timeline and customer-facing comments
- Real-time status updates
- See which team is handling their ticket (after assignment)
- View Gemini AI's suggested solution (optional: shown to customer as "AI Assistant Suggestion")

---

3. AI-Powered Ticket Processing (Gemini Integration)

3.1 Gemini AI Automatic Analysis

- **Gemini API Integration**: Upon ticket creation, Google's Gemini AI model analyzes the ticket content
- **Processing Flow**:

    1. Ticket created by customer
    2. Backend sends ticket data to Gemini API
    3. Gemini processes and returns analysis
    4. Results stored in database and displayed to Super Admin

3.2 Gemini AI Generates

-

Suggested Solution:

-

    - Preliminary response or resolution steps (2-4 sentences)
    - Step-by-step troubleshooting guide (if applicable)
    - Relevant knowledge base article suggestions

-

Team Assignment Recommendation:

-

    - Analyzes subject and description to determine category
    - Recommends: **Technical Support**, **Billing Support**, or **General Support**
    - Provides reasoning for recommendation

-

Priority Validation:

-

    - Confirms or suggests adjusting the customer-selected priority
    - Detects urgency keywords (e.g., "urgent", "critical", "emergency", "down")
    - Provides confidence score for priority assessment

-

Sentiment Analysis (Bonus):

-

    - Detects customer frustration level (Low/Medium/High)
    - Flags potentially escalated situations

3.3 Gemini Prompt Structure

```
Analyze this customer support ticket:

Subject: {ticket.subject}
Description: {ticket.description}
Customer Priority: {ticket.priority}
Category: {ticket.category}

Please provide:
1. Recommended Team Assignment (Technical Support/Billing
Support/General Support) with reasoning
2. Priority Validation (confirm or suggest Low/Medium/High) with
explanation
3. Suggested Solution (2-4 sentence preliminary response or
resolution steps)
4. Confidence Score (0-100%) for your recommendations

Format your response as JSON:
{
  "recommended_team": "Technical Support",
  "team_reasoning": "...",
  "validated_priority": "High",
  "priority_reasoning": "...",
  "suggested_solution": "...",
  "confidence_score": 85
}
```

3.4 AI Response Storage

- Gemini-generated suggestions are stored with the ticket in database
- **Stored Fields**:
    - `ai_recommended_team`
    - `ai_team_reasoning`
    - `ai_validated_priority`
    - `ai_priority_reasoning`
    - `ai_suggested_solution`
    - `ai_confidence_score`
    - `ai_analysis_timestamp`
    - `ai_model_version` (e.g., "gemini-1.5-pro")
- Visible to Super Admin and assigned team members
- Optional: Show AI suggestion to customer as helpful guidance while they wait

3.5 Error Handling for Gemini API

- **Fallback Mechanism**: If Gemini API fails or times out:
    - Fall back to rule-based assignment:
        - Category "Technical" → Technical Support
        - Category "Billing" → Billing Support
        - Category "General" → General Support
    - Log error for Super Admin review
    - Display message: "AI analysis unavailable - using category-based assignment"
    - Super Admin still reviews and can override

4. Super Admin Workflow (Support Director)

4.1 Review & Approval Dashboard

- **View New Tickets**: List of all newly created tickets with Gemini AI suggestions
  - **Gemini AI Suggestion Display Card**:

```
  Gemini AI Analysis━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
━━━Recommended Team: Technical SupportReasoning: "The ticket mentions 'API integration error' and 'webhook not triggering', indicating a technical backend issue."Priority Assessment: High (originally Medium)Reasoning: "Customer indicates service is completely down, affecting business operations. Recommend upgrading to High priority."Suggested Solution:"Check the webhook endpoint configuration in your dashboard. Verify the endpoint URL is correct and the server is responding. Test with our webhook testing tool at dashboard.example.com/webhooks."Confidence: 92%━━━━━━━━━━━━━━━
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━
```

  -

4.2 Approve or Modify

- ✅ **Approve AI Suggestion**: One-click assign to recommended team with AI solution
- **Modify Assignment**: Change team assignment manually:
  - Technical Support Team
  - Billing Support Team
  - General Support Team
- **Edit AI Solution**: Refine or completely rewrite the suggested solution
- **Override Priority**: Change priority if disagree with Gemini's assessment
- **Add Notes**: Include additional instructions for assigned team
- **Confirmation Action**: Final approval triggers:
  - Ticket assignment to team
  - Notification to Team Lead
  - Timeline event logged

4.3 Manual Assignment (Override AI)

- Ability to completely bypass Gemini AI and manually assign tickets
- Select team from dropdown
- Optionally assign to specific team member (Lead or Agent)
- Add custom notes for the assigned team
- Mark as "Manual Assignment" in timeline

4.4 Team Management View

- See workload distribution across teams
- View Gemini AI assignment accuracy per team
- Track how often AI recommendations are accepted vs modified
- View each team's open/in-progress tickets
- Reassign tickets between teams if needed

---

5. Team Member Workflow

5.1 Team Hierarchy

Team Leads (Technical/Billing/General Support Lead)

- View all tickets assigned to their team
- See Gemini AI suggested solutions for guidance
- Reassign tickets within their team
- Escalate tickets to Super Admin
- Access team performance metrics

Team Agents (Technical/Billing/General Support Agent)

- View tickets assigned to them or their team
- Access Gemini AI suggested solution as starting point
- Update ticket status
- Add comments and solutions
- Request help from Team Lead

5.2 Assigned Tickets View

- Team members see tickets assigned to their team
- **Gemini AI Badge**: Indicator showing AI confidence level
    - High Confidence (80-100%)
    - Medium Confidence (60-79%)
    - Low Confidence (<60%)
- Filter by status: Open, In Progress, Resolved
- Priority color coding: Red (High), Yellow (Medium), Green (Low)
- Filter by assigned team member

5.3 Ticket Handling with Gemini AI Assistance

- **View Full Ticket**: Opens detailed view with all information
    - **AI Suggested Solution Panel**:

```
Gemini AI Suggestion (Confidence: 92%)━━━━━━━━━━━━━━━━━━━━━━━━━━
━━━━━━━━━━━━━━━━━━━━[AI suggested solution text]━━━━━━━━━━━━━━━
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━  Mark as Helpful  |   Not
Helpful  |   Adapt Solution
```

- **Update Status**: Toggle between "Open" → "In Progress" → "Resolved"
- **Provide Feedback**: Add resolution notes or final solution in comments
- **Add Comments**: Internal team communication and customer-facing updates
- **AI Solution Feedback**: Track if AI suggestion was helpful (improves future accuracy)
- **Request Escalation**: Flag ticket for Team Lead or Super Admin review

---

6. Comments & Communication

6.1 Comment Section

- **Two Types**:
    - **Internal Comments**: Visible only to team members and super admin (marked with icon)
    - **Customer-Facing Comments**: Visible to customer and all staff (marked with icon)
- **Comment Features**:
    - Timestamp for each comment
    - Author identification (name + role badge):
        - "Support Director"
        - "Technical Support Lead"
        - "Billing Support Agent"
        - "Customer"
        - " Gemini AI" (for AI-generated suggestions)
    - Rich text support (optional: bold, italic, links)
    - Edit/Delete own comments (within 5 minutes of posting)
    - Reply threading (bonus feature)

6.2 AI-Generated Comments

- Gemini AI suggestion displayed as special comment type
- Clearly marked as AI-generated
- Cannot be deleted but can be marked as "not used"
- Team can reference AI suggestion in their customer-facing response

6.3 Notifications (Bonus)

- Customers notified when:
    - Team adds customer-facing comments
    - Ticket status changes
    - Ticket is resolved
- Team notified when:
    - New ticket assigned to their team

    - Customer edits ticket
    - Customer replies to comment

- Team Lead notified when:
    - Ticket assigned to their team by Super Admin
    - Team member escalates ticket

- Super Admin notified when:
    - New ticket created (with Gemini analysis complete)
    - Escalation requested

---

7. Ticket Timeline

7.1 Activity Log

- **Chronological Record** of all ticket events:
    - ✅ **Ticket created** (timestamp, customer name)
    - **Gemini AI analysis completed** (timestamp, confidence score)
        - Shows: Recommended team, suggested solution summary
    - **Super Admin reviewed** (timestamp, action taken: approved/modified)
    - **Team assignment** (timestamp, assigned team: Technical/Billing/General Support)
    - **Assigned to team member** (timestamp, team member name)
    - **Status changes** (Open → In Progress → Resolved)
    - **Comments added** (timestamp, author, role, comment preview)
    - **Ticket edited** (timestamp, what changed)
    - **Priority changed** (timestamp, old → new, changed by)
    - **Team reassignment** (timestamp, old team → new team)
    - / **AI feedback** (timestamp, agent marked AI suggestion helpful/not helpful)
    - **Ticket resolved** (timestamp, resolution notes, resolved by)
    - **Escalations** (timestamp, escalated to whom)

7.2 Visual Timeline

- Display as vertical timeline or activity feed
- Icons for different event types
- Color coding for status changes
- Special styling for Gemini AI events (robot icon)
- Team badges showing which team handled each action
- Expandable entries for detailed information
- Hovering shows full details

7.3 Timeline Example

```
  Jan 3, 2026 10:30 AM
✅ Ticket #1234 created by John Doe
```

```
Jan 3, 2026 10:30 AM
Gemini AI analyzed ticket (Confidence: 92%)
 Recommended: Technical Support Team
 Priority: High (upgraded from Medium)

Jan 3, 2026 10:35 AM
Support Director approved AI recommendation
 Assigned to: Technical Support Team

Jan 3, 2026 10:40 AM
Technical Support Lead assigned to Sarah (Tech Agent)

Jan 3, 2026 10:45 AM
Status changed: Open → In Progress
 By: Sarah (Technical Support Agent)

Jan 3, 2026 11:00 AM
Comment added by Sarah (Tech Support Agent)
 "Following Gemini AI's suggestion, verified webhook config..."

Jan 3, 2026 11:30 AM
Ticket resolved by Sarah (Technical Support Agent)
 Resolution: "Issue fixed. Webhook endpoint updated."
```

---

8. Dashboard & Statistics

8.1 Statistics Header

Customer Dashboard:

- My Open tickets: 3
- My In Progress tickets: 1
- My Resolved tickets: 12

Team Member Dashboard (Technical/Billing/General Support Agent):

- Team's Open tickets: 8
- My assigned In Progress tickets: 2
- Team's Resolved tickets (this week): 15
- Gemini AI Helpful Rate: 85% (how often AI suggestions were useful)

Team Lead Dashboard (Technical/Billing/General Support Lead):

- Team Open tickets: 8
- Team In Progress tickets: 5
- Team Resolved tickets: 15
- Average resolution time for team: 2.5 hours
- AI Assignment Accuracy: 90% (tickets correctly assigned by Gemini)

Super Admin Dashboard (Support Director):

- **Total Tickets**:

    - Total Open: 24
    - Total In Progress: 12
    - Total Resolved: 150

- **Breakdown by Team**:
    - Technical Support: 10 open, 5 in progress, 60 resolved
    - Billing Support: 8 open, 4 in progress, 50 resolved
    - General Support: 6 open, 3 in progress, 40 resolved

- **Gemini AI Performance Metrics**:
    - Overall AI Assignment Accuracy: 88%
    - Average Confidence Score: 85%
    - AI Suggestions Marked Helpful: 82%
    - Priority Validation Accuracy: 91%
    - Times AI was overridden: 18 (12%)

- **System Metrics**:
    - Average resolution time across all teams: 3.2 hours
    - Tickets resolved within 24 hours: 94%
    - Customer satisfaction rate: 4.5/5 (if feedback system implemented)

8.2 Visual Indicators

- **Priority Color Coding**:
    - Red (High)
    - Yellow (Medium)
    - Green (Low)

- **Overdue Warning**:
    - Tickets open >24 hours flagged with icon/orange background
    - Blinking indicator for tickets >48 hours

- **Status Badges**:
    - Open (green circle)
    - In Progress (blue circle with progress ring)
    - ✓ Resolved (green checkmark)

- **Team Badges**:
    - Technical Support (blue badge)
    - Billing Support (green badge)
    - General Support (purple badge)

- **AI Confidence Indicators**:
    - High (80-100%): Solid green
    - Medium (60-79%): Yellow

- Low (<60%): Red, review recommended

---

9. Ticket List/Grid Display

9.1 Display Fields

- **Ticket ID** (auto-generated, e.g., #1234)
- **Subject** (truncated with tooltip on hover)
- **Priority** (color-coded badge)
- **Status** (badge: Open/In Progress/Resolved)
- **Category** (Technical/Billing/General)
- **Assigned Team** (with team badge icon)
- **Assigned To** (Team member name or "Unassigned")
- **AI Confidence** (only visible to team/admin)
- **Date Created** (e.g., "2 hours ago" or "Jan 3, 2026")
- **Last Updated** (relative time)
- **Created By** (Customer name)

9.2 Filtering Options (By Role)

Super Admin:

- Filter by team (Technical/Billing/General Support)
- Filter by status (Open/In Progress/Resolved)
- Filter by priority (High/Medium/Low)
- Filter by category
- Filter by AI confidence level
- Filter by date range
- Search by ticket ID, customer name, subject, description

Team Lead:

- Filter by team member (see individual workloads)
- Filter by status, priority
- Filter by AI confidence (find tickets needing review)
- View team performance metrics
- Filter by "AI Approved" vs "Manually Assigned"

Team Agent:

- Filter by "Assigned to Me" vs "Assigned to Team"
- Filter by status, priority
- Filter by "High AI Confidence" (easier tickets to start with)
- View own performance stats

Customer:

- View only their tickets
- Filter by status (Open/In Progress/Resolved)

- Sort by date created or last updated

9.3 Actions

- **View Details**: Opens full ticket view with timeline and comments
- **Edit** (Customer only): Modify ticket before resolution
- **Delete** (Super Admin only): Remove ticket with confirmation popup
- **Quick Status Update** (Team members): Dropdown to change status
- **Reassign** (Team Lead and Super Admin): Change team or team member
- **View AI Analysis** (Team/Admin): Expand to see full Gemini suggestion
- **Copy AI Solution** (Team members): Copy AI suggestion to clipboard
- **Mark AI Helpful/Not Helpful** (Team members): Provide feedback

---

10. Team Assignment Logic

10.1 Gemini AI Recommendation Mapping

Primary Logic (based on ticket content analysis):

- Analyzes subject and description with Gemini AI
- Uses natural language understanding to determine intent
- **Category: Technical** + keywords (bug, error, API, integration) → **Technical Support Team**
- **Category: Billing** + keywords (payment, invoice, refund, subscription) → **Billing Support Team**
- **Category: General** + keywords (account, question, how-to, feature) → **General Support Team**

Secondary Logic (priority and urgency):

- High priority + critical keywords → Assign to Team Lead
- Medium/Low priority → Assign to Team Agent
- Considers current team workload from database

10.2 Manual Override by Super Admin

- Super Admin can override Gemini AI recommendation
- Tickets can be reassigned between teams anytime
- Reason for override is logged in timeline
- Feedback loop: overrides help improve AI accuracy

10.3 Team Lead Assignment

- Team Leads can assign tickets to specific team members
- Consider agent workload and expertise
- Can reassign within team without Super Admin approval

10.4 Load Balancing (Bonus)

- Gemini AI considers current team workload from database:

```
{  "technical_support_open_tickets": 10,
"billing_support_open_tickets": 5,
"general_support_open_tickets": 8}
```

- If recommended team is overloaded, suggest alternative
- Priority tickets still go to best-fit team regardless of load
- Display workload balance to Super Admin during approval

---

11. Technical Requirements

11.1 Backend & Database

- **Backend**: Node.js (Express) / Python (Flask/Django) / Go (Gin)
- **Database**: SQLite / MongoDB / PostgreSQL for persistent storage

Database Schema:

```
-- Users Table
users:
  - id (PRIMARY KEY)
  - email (UNIQUE)
  - password_hash
  - name
  - role (customer, team_agent, team_lead, super_admin)
  - team (technical_support, billing_support, general_support, null)
  - created_at

-- Tickets Table
tickets:
  - id (PRIMARY KEY)
  - ticket_number (UNIQUE, auto-generated)
  - subject
  - description
  - category (technical, billing, general)
  - priority (low, medium, high)
  - status (open, in_progress, resolved)
  - customer_id (FOREIGN KEY → users.id)
  - assigned_team (technical_support, billing_support,
general_support, null)
  - assigned_to (FOREIGN KEY → users.id, null)
  - created_at
  - updated_at
  - resolved_at (null until resolved)

  -- Gemini AI Fields
  - ai_recommended_team
  - ai_team_reasoning
  - ai_validated_priority
  - ai_priority_reasoning
  - ai_suggested_solution
  - ai_confidence_score
```

```
  - ai_analysis_timestamp
  - ai_model_version (e.g., "gemini-1.5-pro")
  - ai_feedback_helpful (boolean, null, set by team member)

-- Comments Table
comments:
  - id (PRIMARY KEY)
  - ticket_id (FOREIGN KEY → tickets.id)
  - author_id (FOREIGN KEY → users.id, null for AI)
  - content
  - comment_type (internal, customer_facing, ai_generated)
  - is_ai_comment (boolean)
  - created_at
  - updated_at
  - edited (boolean)

-- Timeline Events Table
timeline_events:
  - id (PRIMARY KEY)
  - ticket_id (FOREIGN KEY → tickets.id)
  - event_type (created, ai_analyzed, admin_reviewed, assigned,
status_changed,
                commented, edited, priority_changed, reassigned,
resolved, escalated)
  - event_description
  - performed_by (FOREIGN KEY → users.id, null for system/AI)
  - details (JSON field for additional data)
  - timestamp

-- Teams Table (optional for team management)
teams:
  - id (PRIMARY KEY)
  - team_name (technical_support, billing_support, general_support)
  - team_lead_id (FOREIGN KEY → users.id)
  - description
```

- **API Design**: RESTful endpoints:

```
POST   /api/auth/signupPOST   /api/auth/loginPOST
/api/auth/google-oauthGET    /api/auth/mePOST   /api/tickets
(create ticket, triggers Gemini analysis)GET    /api/tickets
(list tickets, role-based filtering)GET    /api/tickets/:id
(get ticket details)PUT    /api/tickets/:id      (update
ticket)DELETE /api/tickets/:id      (delete ticket, super admin
only)POST   /api/tickets/:id/approve-ai   (super admin approves
Gemini suggestion)POST   /api/tickets/:id/assign       (assign
to team/member)PUT    /api/tickets/:id/status       (update
status)POST   /api/tickets/:id/comments     (add comment)GET
/api/tickets/:id/comments     (get comments)PUT
/api/comments/:id             (edit comment)GET
/api/tickets/:id/timeline     (get timeline events)POST
/api/ai/feedback              (mark AI suggestion
helpful/not)GET    /api/stats/dashboard          (role-based
dashboard stats)GET    /api/stats/ai-performance     (Gemini AI
metrics, admin only)
```

-

11.2 Gemini AI Integration

API Setup:

- **API Key**: Store in environment variable `GEMINI_API_KEY`
- **Model**: Use `gemini-1.5-pro` or `gemini-1.5-flash` (faster for hackathon)
- **Endpoint**: Google AI Studio or Vertex AI

Implementation Example (Node.js):

```
const { GoogleGenerativeAI } = require("@google/generative-ai");

const genAI = new GoogleGenerativeAI(process.env.GEMINI_API_KEY);

async function analyzeTicket(ticket) {
  try {
    const model = genAI.getGenerativeModel({ model: "gemini-1.5-
flash" });

    const prompt = `Analyze this customer support ticket:

Subject: ${ticket.subject}
Description: ${ticket.description}
Customer Priority: ${ticket.priority}
Category: ${ticket.category}

Provide:
1. Recommended Team (Technical Support/Billing Support/General
Support) with reasoning
2. Priority Validation (confirm or suggest Low/Medium/High) with
explanation
3. Suggested Solution (2-4 sentences)
4. Confidence Score (0-100%)

Respond in JSON format.`;

    const result = await model.generateContent(prompt);
    const response = await result.response;
    const text = response.text();

    // Parse JSON response from Gemini
    const aiAnalysis = JSON.parse(text);

    return {
      recommended_team: aiAnalysis.recommended_team,
      team_reasoning: aiAnalysis.team_reasoning,
      validated_priority: aiAnalysis.validated_priority,
      priority_reasoning: aiAnalysis.priority_reasoning,
      suggested_solution: aiAnalysis.suggested_solution,
      confidence_score: aiAnalysis.confidence_score
    };
  } catch (error) {
    console.error("Gemini AI Error:", error);
    // Fallback to rule-based assignment
    return fallbackAssignment(ticket);
  }
}

function fallbackAssignment(ticket) {
  const teamMap = {
    'technical': 'Technical Support',
    'billing': 'Billing Support',
```

```
    'general': 'General Support'
  };

  return {
    recommended_team: teamMap[ticket.category],
    team_reasoning: "Automatic assignment based on category (AI
unavailable)",
    validated_priority: ticket.priority,
    priority_reasoning: "Customer-selected priority maintained",
    suggested_solution: "Please review this ticket manually.",
    confidence_score: 50
  };
}
```

Async Processing:

- Ticket creation returns immediately to customer
- Gemini AI analysis runs in background (async job/queue)
- Timeline shows "AI analyzing..." → "AI analysis complete"
- Super Admin notified when analysis is ready for review

Rate Limiting:

- Implement request throttling to avoid API quota limits
- Cache similar tickets to reduce API calls (bonus)
- Queue system for high-volume scenarios

Error Handling:

- Timeout after 10 seconds → fallback to rule-based
- API error → fallback to rule-based
- Invalid JSON response → retry once, then fallback
- Log all errors for debugging

11.3 Authentication

- **Secure Password Storage**: bcrypt with salt rounds 10+
- **Google OAuth 2.0**:
    - Use official Google OAuth library
    - Store OAuth tokens securely
    - Link OAuth accounts to existing emails
- **JWT Tokens**:
    - Include: user ID, role, team
    - Expiry: 24 hours
    - Refresh token mechanism (bonus)
- **Protected Routes**:
    - Middleware checks JWT token
    - Role-based access control (RBAC)
    - Team-based data filtering

Role-Based Access Control (RBAC):

```
// Middleware example
function authorize(allowedRoles) {
  return (req, res, next) => {
    if (!allowedRoles.includes(req.user.role)) {
      return res.status(403).json({ error: "Access denied" });
    }
    next();
  };
}

// Usage
app.get('/api/stats/ai-performance',
  authenticate,
  authorize(['super_admin']),
  getAIPerformance
);
```

11.4 Stability & Performance

- **Error Handling**:
    - Graceful database connection errors
    - Input validation on frontend AND backend
    - Handle empty result sets
    - Try-catch blocks around all Gemini API calls
- **Loading States**:
    - Show spinner during AI analysis
    - "AI analyzing ticket..." message
    - Progressive UI updates as data loads
- **Retry Logic**:
    - Retry Gemini API calls once on failure
    - Exponential backoff for rate limit errors
- **Validation**:
    - Frontend: Immediate feedback
    - Backend: Sanitize inputs, prevent SQL injection
    - Gemini response validation: ensure JSON is parseable

11.5 Environment Variables

```
# Database
DATABASE_URL=postgresql://localhost/webfrenzy_db

# Gemini AI
GEMINI_API_KEY=your_gemini_api_key_here
GEMINI_MODEL=gemini-1.5-flash

# Authentication
JWT_SECRET=your_jwt_secret_here
```

```
GOOGLE_CLIENT_ID=your_google_oauth_client_id
GOOGLE_CLIENT_SECRET=your_google_oauth_secret

# App Config
PORT=3000
NODE_ENV=development
```

---

12. Bonus Features

12.1 Search & Filter

- Full-text search across subject and description
- Filter by priority, category, status, team, assigned member
- Combined filters (e.g., "High priority Technical tickets")
- Save filter presets (e.g., "My urgent tickets")

12.2 Dark Mode

- Theme toggle in user settings or header
- Persistent theme preference per user (stored in DB)
- Smooth transition animation
- Adjust all colors, including charts and badges

12.3 Export

- **CSV Export**: Download filtered ticket list
- **PDF Export**: Generate detailed ticket report with:
    - Ticket details
    - Full timeline
    - All comments
    - Gemini AI analysis summary
- Export options:
    - Single ticket (PDF)
    - Filtered tickets (CSV)
    - Team performance report (CSV/PDF for Team Leads)
    - AI performance report (CSV for Super Admin)

12.4 Gemini AI Learning Dashboard (Advanced)

- **AI Accuracy Tracking**:
    - Show trend over time (improving/declining)
    - Break down by team (which team has best AI accuracy)
    - Compare AI priority validation vs actual outcomes
- **Common Overrides Analysis**:
    - Show top reasons Super Admin overrides AI

    - Display frequently misclassified ticket types
    - Use insights to improve AI prompts

- **Helpful Suggestion Rate**:

    - Track which AI suggestions were marked helpful by team
    - Show examples of most helpful AI responses
    - Identify patterns in unhelpful suggestions

12.5 Team Performance Analytics

- Average resolution time per team
- Tickets resolved per team member
- Comparison: tickets handled with AI help vs without
- "Top Performer" badge for fastest/most tickets resolved
- Team leaderboard (friendly competition)

.6 Real-time Updates (WebSocket)

- Live updates when:
    - New ticket assigned to team
    - Ticket status changes
    - New comments added
    - Gemini AI completes analysis
- Live notification bell icon with count
- Toast notifications for important events
- No need to refresh page

12.7 Customer Satisfaction Feedback

- After ticket is resolved, customer can rate (1-5 stars)
- Optional comment on resolution quality
- Track satisfaction by team
- Show on Super Admin dashboard
- Badge system: "95% satisfaction rate"

12.8 Gemini AI Continuous Improvement

- **Feedback Loop**:

    - When team marks AI suggestion "not helpful", ask why
    - Collect feedback reasons:

        - Incorrect team assignment
        - Wrong priority assessment
        - Unhelpful solution
        - Missing context

    - Log feedback to improve future prompts

- **A/B Testing** (Advanced):
    - Test different Gemini prompt structures

- Compare accuracy of different models
- Choose best-performing approach

12.9 Mobile App (Stretch Goal)

- Responsive web design for mobile browsers
- Native mobile notifications
- Quick actions from notification
- Simplified mobile UI for on-the-go management

---

13. User Flow Summary

Customer Journey

1. **Sign up / Login** (email+password or Google OAuth)
2. **Create ticket** → Gemini AI starts analyzing immediately
3. **See confirmation**: "Ticket submitted! AI is analyzing your request..."
4. **Optional**: View AI suggested solution while waiting
5. **Notification**: "Your ticket has been assigned to Technical Support Team"
6. **Track status**: See timeline and team comments
7. **Read solution**: Team provides resolution based on AI + expertise
8. **Edit if needed**: Modify ticket before resolution
9. **Ticket resolved**: View final solution and timeline
10. **Rate experience** (bonus): Provide feedback on resolution

Super Admin Journey (Support Director)

1. **Login** → Dashboard shows pending tickets with Gemini AI analysis
2. **Review new tickets**:

```
Ticket #1234 - "API webhook not triggering"  Gemini AI (92% confidence):→ Technical Support Team→ Priority: High→ Solution: [AI suggestion preview]
```

3.
4. **Decision**:

   - ✓ **Approve**: Ticket instantly assigned to Technical Support
   - **Modify**: Change team to Billing Support, add note
   - **Edit Solution**: Refine AI suggestion before assignment

5. **Monitor workload**: Check team distribution
6. **View AI metrics**: Track Gemini accuracy (88% this week)
7. **Handle escalations**: Review flagged tickets
8. **Export reports**: Generate team performance PDF

Team Lead Journey (Technical/Billing/General Support Lead)

1. **Login** → See new assignments from Super Admin
    2. **Review ticket** + Gemini AI suggestion:

```
  AI suggests: "Check webhook endpoint config..."Confidence:
92%
```

    3.
4. **Assign to team member**: "Sarah has API expertise, assign to her"
5. **Monitor team progress**: 5 tickets in progress, 3 resolved today
6. **Help when needed**: Team member asks for guidance
7. **Track AI usefulness**: 85% of AI suggestions were helpful to team
8. **Report to Super Admin**: Weekly team performance summary

Team Agent Journey (Technical/Billing/General Support Agent)

1. **Login** → View tickets assigned to me
    2. **Open ticket #1234**:

```
Customer: "API webhook stopped working"  Gemini AI Suggestion
(92% confidence):"Check webhook endpoint configuration..."[Full
AI suggested solution displayed]
```

    3.
4. **Update status**: "Open" → "In Progress"
5. **Follow AI guidance**: Check webhook config as suggested
6. **Find issue**: Endpoint URL was incorrect
7. **Add internal comment**: "AI suggestion was spot-on! Fixed URL."
8. **Mark AI helpful**: "This AI suggestion helped resolve the issue"
9. **Customer-facing comment**: "Issue resolved! Your webhook endpoint..."
10. **Mark resolved**: Status → "Resolved"
11. **Move to next ticket**: Repeat process

---

14. Validation Rules

Input Validation

- **Subject**:
    - Cannot be empty
    - Minimum 5 characters
    - Maximum 200 characters
- **Description**:
    - Cannot be empty
    - Minimum 10 characters
    - Maximum 5000 characters

- **Category**:
    - Must select one: Technical, Billing, General
    - Required field
- **Priority**:
    - Must select one: Low, Medium, High
    - Required field
- **Email** (Sign up/Login):
    - Valid email format (regex validation)
    - Cannot be empty
- **Password** (Sign up):
    - Minimum 8 characters
    - At least 1 uppercase letter
    - At least 1 number
    - At least 1 special character (optional but recommended)

Business Rules

- Team assignment required before status can move to "In Progress"
- Ticket cannot be edited by customer after status is "Resolved"
- Only ticket owner (customer) can edit their ticket details
- Only Super Admin can permanently delete tickets
- Only assigned team members can update ticket status
- Comments cannot be deleted after 5 minutes (only edited)
- Internal comments invisible to customers (enforced at API level)
- Delete action requires confirmation popup with "Are you sure?"
- Team reassignment requires:
    - Super Admin role, OR
    - Team Lead role (within same team)
- Gemini AI analysis must complete before Super Admin can review (or use fallback)

Gemini AI Rules

- AI analysis timeout: 15 seconds max
- If AI fails → automatic fallback to rule-based assignment
- AI confidence <60% → flag for manual review by Super Admin
- AI suggested solution max length: 500 characters
- Store AI model version for future reference

---

15. Team & Role Summary Table

Role	Display Name	Team	Permissions	Access Level

Role	Display Name	Team	Permissions	Access Level
Super Admin	Support Director	N/A (oversees all)	View all tickets, approve/modify Gemini AI, assign teams, reassign tickets, system analytics, delete tickets, view AI performance	Full Access
Team Lead	Technical Support Lead	Technical Support	View team tickets, assign within team, escalate, team analytics, view AI suggestions	Team Access
Team Lead	Billing Support Lead	Billing Support	View team tickets, assign within team, escalate, team analytics, view AI suggestions	Team Access
Team Lead	General Support Lead	General Support	View team tickets, assign within team, escalate, team analytics, view AI suggestions	Team Access
Team Agent	Technical Support Agent	Technical Support	View assigned tickets, update status, add comments, mark AI helpful, view AI suggestions	Assigned Tickets
Team Agent	Billing Support Agent	Billing Support	View assigned tickets, update status, add comments, mark AI helpful, view AI suggestions	Assigned Tickets
Team Agent	General Support Agent	General Support	View assigned tickets, update status, add comments, mark AI helpful, view AI suggestions	Assigned Tickets
Customer	[Customer Name]	N/A	Create tickets, view own tickets, edit before resolution, view customer-facing comments, view timeline	Own Tickets Only

---

16. Non-Functional Requirements

Performance

- **UI Response Time**: UI updates reflect database state immediately
- **API Response Time**:
    - Most endpoints: <200ms
    - Gemini AI analysis: <10 seconds (with loading indicator)
- **Database Queries**: Optimized with indexes on frequently queried fields
- **Frontend Load Time**: Initial page load <2 seconds

Scalability

- Handle 50+ concurrent users during demo
- Support 1000+ tickets in database without performance degradation
- Efficient Gemini API usage (avoid unnecessary calls)

Security

- No exposed API keys (use environment variables)
- Secure authentication (JWT + bcrypt)
- Role-based access control enforced at API level
- SQL injection prevention (parameterized queries)
- XSS protection (sanitize inputs)
- CORS configured properly
- HTTPS in production (for hackathon, HTTP acceptable)

Accessibility

- Basic keyboard navigation support
- Proper semantic HTML
- ARIA labels for screen readers
- Color contrast ratios meet WCAG 2.1 AA
- Focus indicators visible

Browser Compatibility

- Works on modern browsers:
    - Chrome 90+
    - Firefox 88+
    - Safari 14+
    - Edge 90+
- Responsive design for tablets (768px+) and desktops

Data Integrity

- All data persists across:
    - Browser refresh
    - Server restart
    - Database reconnection
- Foreign key constraints in database

- Transaction support for critical operations

---

17. Gemini AI Prompt Engineering Tips

Effective Prompts for Ticket Analysis

```
You are a customer support AI assistant analyzing tickets for routing
and prioritization.

Analyze this ticket:
Subject: {subject}
Description: {description}
Customer Priority: {priority}
Category: {category}

Tasks:
1. TEAM ASSIGNMENT: Choose the best team
   - Technical Support: handles bugs, errors, integrations, API
issues, system problems
   - Billing Support: handles payments, invoices, refunds,
subscriptions, pricing
   - General Support: handles accounts, features, questions,
onboarding, misc

2. PRIORITY VALIDATION:
   - Assess urgency based on business impact keywords
   - Consider: "critical", "urgent", "down", "not working",
"emergency"
   - Recommend Low/Medium/High

3. SUGGESTED SOLUTION:
   - Provide 2-4 sentence actionable response
   - Be specific and helpful
   - Include troubleshooting steps if applicable

4. CONFIDENCE: Rate your confidence 0-100%

Respond ONLY in this JSON format:
{
  "recommended_team": "Technical Support",
  "team_reasoning": "Ticket mentions API integration error requiring
technical expertise",
  "validated_priority": "High",
  "priority_reasoning": "Service is completely down affecting
business operations",
  "suggested_solution": "Check your API endpoint configuration in the
dashboard...",
  "confidence_score": 92
}
```

Handling Edge Cases

- Train Gemini with few-shot examples for better accuracy
- Handle ambiguous tickets (e.g., "need help") → default to General Support
- Detect spam or inappropriate content → flag for Super Admin

- Multi-category tickets → choose primary focus

---

18. Submission Checklist

Required Files

- ✅ `README.md` with:
    - Tech stack (e.g., React + Node.js + PostgreSQL + Gemini AI)
    - Setup instructions
    - Environment variables needed
    - Gemini API key setup instructions
    - How to run locally
- ✅ Source code (exclude `node_modules/`, `dist/`, `venv/`, etc.)
- ✅ Database schema file or migrations
- ✅ `.env.example` file showing required environment variables
- ✅ Gemini AI integration code

Archive Format

- File name: `[YourName]_WebFrenzy2026.zip`
- Max size: Keep manageable (exclude dependencies)

Pre-Demo Checklist

- ✅ Gemini API key is working and has quota
- ✅ Database seeded with sample data:
    - 1 Super Admin account
    - 3 Team Leads (one per team)
    - 6 Team Agents (two per team)
    - 5 Customer accounts
    - 10-15 sample tickets in various states
    - Some tickets with AI analysis completed
- ✅ Google OAuth configured (if implementing)
- ✅ Test all user flows
- ✅ Verify AI analysis works end-to-end
- ✅ Check timeline displays correctly
- ✅ Ensure overdue warnings appear
- ✅ Test persistence (refresh browser, restart server)

---

19. Live Demo Script

Flow to Demonstrate (8-10 minutes)

1. Introduction (30 seconds)

- "We built an AI-powered ticket system using Gemini AI for intelligent routing"

2. Customer Creates Ticket (1 minute)

- Show sign up/login
- Create ticket: "Our payment gateway integration stopped working"
- Category: Technical, Priority: High
- Show "AI is analyzing..." loading state
- Gemini completes analysis (show in console/network tab)

3. Super Admin Reviews AI Suggestion (2 minutes)

- Login as Support Director
- Show dashboard with new ticket
    - Display Gemini AI analysis panel:

```
  Gemini AI Analysis (95% confidence)Recommended: Technical
Support TeamReasoning: "Payment gateway integration is
technical issue..."Priority: High (confirmed)Solution: "Check
API credentials and webhook settings..."
```

    -
- Approve AI suggestion
- Ticket assigned to Technical Support Team

4. Team Member Handles Ticket (2 minutes)

- Login as Technical Support Agent
- View assigned ticket
- Show AI suggested solution
- Update status: Open → In Progress
- Add internal comment: "Following AI suggestion..."
- Add customer-facing comment with resolution
- Mark AI suggestion as helpful ( )
- Resolve ticket

5. Show Dashboard & Analytics (1.5 minutes)

- Switch back to Super Admin
- Show updated statistics:
    - Tickets resolved: +1
    - AI accuracy: 88%
    - Team performance metrics
- Show Gemini AI performance dashboard:
    - AI suggestions marked helpful: 85%
    - Average confidence score: 87%

6. Demonstrate UI Features (1.5 minutes)

- Show priority color coding (Red/Yellow/Green)
- Show overdue warning (if any tickets >24 hours)
- Show timeline with all events including AI analysis
- Show comment section (internal vs customer-facing)
- Show filter/search (bonus)
- Show dark mode toggle (bonus)

7. Persistence Check (1 minute)

- Refresh browser → data persists
- Show timeline still intact
- Restart backend server (if time permits)
- Verify data still present

8. Code Review (1.5 minutes)

- Show Gemini API integration code
- Show database schema with AI fields
- Show API routes structure
- Explain role-based access control

Total: ~10 minutes

---

20. Evaluation Criteria Alignment

Functional Completeness (40%)

✅ All core requirements met:

- User authentication (email + Google OAuth)
- Ticket CRUD operations
- Status tracking (Open/In Progress/Resolved)
- Comment system (internal + customer-facing)
- Timeline with all events
- **BONUS: Gemini AI integration** for intelligent routing and solutions

UI/UX & Usability (20%)

✅ Professional interface with:

- Color-coded priorities
- Statistics dashboard
- Overdue warnings
- Team badges
- AI confidence indicators
- Responsive design
- Dark mode (bonus)

Code Quality & Architecture (20%)

✅ Well-structured codebase:

- Clean separation: Frontend / Backend / Database
- RESTful API design
- Modular Gemini AI integration
- Error handling and fallbacks
- Role-based access control
- Environment variables for config

Stability & Validation (10%)

✅ Robust application:

- Input validation (frontend + backend)
- Graceful error handling for AI failures
- Database persistence across restarts
- Fallback to rule-based assignment
- Loading states during AI processing

Bonus Features (10%)

✅ Implemented extras:

- **Gemini AI integration** (main differentiator!)
- Search & filter
- Dark mode
- Export (CSV/PDF)
- AI performance analytics
- Real-time updates (if time permits)
- Customer satisfaction ratings

Document Version: 2.1 (Gemini AI Integration)
Event: WebFrenzy 2026 Hackathon
Duration: 3 Hours
AI Model: Google Gemini (gemini-1.5-flash or gemini-1.5-pro)
AI Assistance: Allowed (no AI IDEs/auto-coding agents)
Last Updated: January 2026

Quick Reference: Gemini AI Integration

Environment Setup

```
npm install @google/generative-ai
# or
pip install google-generativeai
```

API Key

Get from: https://aistudio.google.com/app/apikey

Basic Usage

```
const { GoogleGenerativeAI } = require("@google/generative-ai");
const genAI = new GoogleGenerativeAI(process.env.GEMINI_API_KEY);
const model = genAI.getGenerativeModel({ model: "gemini-1.5-flash" });

const result = await model.generateContent(prompt);
const response = await result.response;
const text = response.text();
```

Key Gemini Features to Showcase

1. ✅ Intelligent team assignment based on ticket content
2. ✅ Priority validation and urgency detection
3. ✅ Automated solution suggestions
4. ✅ Confidence scoring
5. ✅ Learning from feedback (helpful/not helpful)
6. ✅ Performance analytics dashboard

This makes your project stand out with cutting-edge AI integration!