Group 7:

Ali Farazmand

Jarod Jian Kang Hery Chen

Romil Lakhani

Dev Pradyuman Pandya

Haroon Waseem Baig

Business Case for Group 7  
 Communication Platform “MyChat”

\*Note: Project name may be subject to change. MyChat is not the final name for the platform.

# 1. Introduction/Background

In today's digital age, communication platforms like Discord and Microsoft Teams have become integral to both personal and professional interactions, for business use and for online friendships. Discord, initially designed for gamers, has expanded its user base to include diverse communities and organizations. With Discord approaching a rumored IPO and appointing Humam Sakhnini (read more [here](https://discord.com/blog/discord-appoints-new-ceo-humam-sakhnini)), former President of King Digital Entertainment, as its CEO, there are growing concerns about an impending shift toward aggressive monetization strategies—mirroring his history with microtransactions and ads in games like Candy Crush (*Discord's leadership shift: Humam Sakhnini takes Helm as CEO amid IPO rumors*).

Meanwhile, Microsoft Teams excels in offering structured collaboration tools for corporate environments but lacks the community-oriented flexibility and casual feel that has made Discord widely appealing.

As of early 2025, Discord boasts approximately 200 million monthly active users (About Discord). Microsoft Teams has around 320 million active monthly users (*Microsoft Teams Statistics - by revenue, Demographics, usage (2025)*). With their immense popularity, these two apps have set the standards that need to be followed for a communication platform to be useful for modern users. Each platform offers unique strengths but also leaves significant gaps, particularly when users seek a unified experience combining casual communication with structured collaboration.

This gap presents an opportunity to develop a communication platform that combines the best of both worlds: the community engagement of Discord and the organizational capabilities of Microsoft Teams that can be the best overall chatting and communication platform for general and casual use and for gamers with many of discord’s greatest features and several of Microsoft Team’s most useful features that are lacking in discord and other casual chatting apps.

# 2. Business Objective

Our objective is to develop a comprehensive communication platform that combines the strengths of Discord and Microsoft Teams. The platform will cater to both casual communities and small professional teams by offering robust customization, privacy controls, and productivity features.

Key objectives include:

* **Flexible Server Structures**: Allow users to create servers with either hierarchical team-based structures or simple role-based access, catering to both professional and casual use
* **Server Creation Templates:** Provide users with templates with preset roles and permissions for different types of servers such as small casual servers for friends and templates for bigger servers that require more strict role permissions and moderation
* **Integrated Call Recording**: Provide built-in voice and video call recording capabilities with customizable privacy and permission settings regarding whether each user consents and wants their voice and video feed to be included in the call recording. Call recording features are currently absent in Discord but valuable for meetings and content creation, and the recording feature in teams does not give individual users in a call the option for their voice and video to be excluded from the call recording.
* **User-Specific Storage Solutions**: Offer separate, scalable storage options for file uploads, enabling users to pay for the storage they need without being tied to a monthly subscription model, unlike Discord’s per-upload limits and nitro subscription model.
* **In-App File Interaction**: Enable users to open and edit common file types such as text files, word documents, spreadsheets, and code files directly within the app, streamlining collaboration for user convenience, reducing the need for external applications.
* **Third-party app integration:** Providing a feature allowing third-party app integration like Microsoft 365 Online apps such as Word Online or other productivity tools to create and share collaborative documents with friends, groups, and servers on the platform for increased engagement and collaboration and to further extend in-app file interaction capabilities.
* **Profile appearance customization:** Offer profile customization features for users such as how their name and card appears on a server user list, how it looks on other users’ friends lists, and how their full profile card appears. Discord has some of these features, but some of their animated profile effects do not meet good accessibility guidelines and make it difficult or impossible to view user bios. We will make sure that the designs for profile customization packs will not obscure text and will allow users to have both readable and beautiful profile bios.
* **App appearance customization:** Offer themes, icons, call ringtones, notification sounds that can be purchased from an in-app shop as it is one of the most requested features for discord.
* **Custom emotes and stickers:** Provide the ability for users to upload static or animated emotes and stickers to a server if they have permission, like on discord, but only if one of the server admins with the necessary emote and sticker permissions has enough free storage purchased from our platform. Users that have permission and upload emotes, and stickers will see a menu to choose which admin’s storage they want to upload to, and depending on the admin’s storage settings for that server, the admin will either be asked to authorize the upload or to automatically grant the permission and receive a notification instead.
* **Granular role and user permissions:** Provide individual role and user settings for admins to customize on servers. Discord currently has some role and permission settings, but they do not cover certain settings. Our platform will provide more granular role and user permissions for the app’s features as we roll them out.

By focusing on these objectives, we aim to deliver a platform that meets the diverse needs of modern users seeking both community engagement and efficient collaboration tools and to make users remain on our platform without the need to constantly switch to other external apps. We want to create a convenient and useful communication platform for casual, semi-casual, and gaming users.

# 3. Current Situation and Problem/Opportunity Statement

While Discord has great features for communication between friend groups, for gaming, and for content creators, it is missing key features that make it difficult and inconvenient to collaborate. Discord lacks built-in file editing and intuitive server templates which causes a lot of difficulty and deters users from setting up new servers, and discord also lacks a call recording system that lets each of the individual users in the call to decide if they want their audio or video included in the recording.

On the other hand, Microsoft Teams has powerful tools for business and education with its structured roles and hierarchical permissions, file collaboration, and scheduling but feels overly formal and lacks the personal feel and customization users love in Discord. Microsoft Teams also does not work well for casual or smaller groups who do not need enterprise-level complexity and those want the look and feel of a more casual chatting app without the boring, bland, and stressful feel of a corporate communication platform.

This creates an opportunity to build a hybrid platform that is as fun and community oriented as Discord, yet structured and efficient like Microsoft Teams.

This is the opportunity we are using to build that kind of platform which combines the best parts of both apps as well as providing convenience and features that both Discord and Teams are lacking.

Our platform will offer:

* Easy-to-configure servers with both casual and hierarchical structures
* Pre-built templates to simplify server creation
* Opt-in call recording with per-user controls and notifications
* Scalable storage purchases for uploads
* Real-time file collaboration and integration with third-party tools
* Extensive visual and profile customization options
* Advanced server administration tools with fine-grained permissions

This is the platform we would want to use ourselves; something powerful but flexible, professional when needed, but never boring. This is what our team is creating.

# 4. Critical Assumptions and Constraints

**Assumptions:**

There is dissatisfaction among Discord users due to changes driven by its IPO (initial public offering) trajectory, creating a market gap for a more user-focused communication platform. That is why users are willing to adopt a new platform that offers the combined strengths of Discord and Microsoft Teams, particularly if it improves limitations in file storage, media recording, and file viewing.

The necessary technical talent and infrastructure can be secured to build a scalable and secure platform capable of handling real-time voice, video, and chat communications.

Additionally, integrating hierarchical server structures, file viewing, and call recording features will provide significant competitive advantages that attract both people for casual and professional purposes.

**Constraints:**

**Scope:**

The platform must include:

* Basics communications functionalities: chat, video, voice, schedulable meetings...
* Hierarchical structure server with discords roles
* In-app file viewing
* Extensible file storing that is based on the user subscription type.

**Schedule:**

* The product must be delivered to beta testers within 8 months to take advantage of Discord's IPO timing and gaining public perspective.
* The new system must be implemented in a span of 4 months

**Budget:**

* The platform must be developed within a maximum budget of $800 000 to remain viable for the initial release.

**Quality:**

* The platform must meet or exceed Discord and Microsoft Teams in terms of:
* Call reliability and audio/video quality
* UI/UX responsiveness and experience
* It must hold security and privacy standards
* User satisfaction targets must remain above 90% in surveys.

**Resources:**

The project will be limited to

* 2 developers
* 2 UI/UX designers
* 1 Project Manager

# 6. Preliminary Project Requirements

**Day-One MVP (Minimum Viable Product) Functional Requirements**:

These features are required for the Minimum Viable Product (MVP) launch:

* **User Account System**: Email/password registration, login, password reset, and two-factor authentication (2FA). Basic profile customization interface.
* **Real-Time Communication Tools**: Direct messaging and channel-based communication with real-time updates via WebSocket. Basic message features including emojis, message editing/deletion, and typing indicators. WebRTC-based voice calls with optional high-fidelity audio mode.
* **Server Creation and Management**: Ability to create servers using basic templates (e.g., gaming, group, professional). Role-based permissions must be supported.
* **Storage and File Interaction**: Upload, download, and in-app viewing of common file types (images, documents, spreadsheets, code). Scalable server storage options available for purchase. Admin-controlled storage quotas.
* **Client Variants**:
  + *Full Client*: Electron or Flutter-based cross-platform app with full chat, server, and call functionality.
  + *Lightweight Windows Client*: Native C++/C# app with support for voice calls, private message notifications, and optional DirectX-based in-game overlay.
* **Notifications and Presence**: Real-time online status tracking and system notifications. Lightweight client must use Windows toast notifications.

**Post-Launch Staged Rollout Features**:

These features are planned for future phases after MVP launch:

* Server discovery and public community channels
* Built-in calendar, scheduling, and reminders
* Third-party integrations with Notion, Slack, Google Forms, Evernote, and Scrivener
* Emote/sticker creation tools and marketplace
* In-app purchases for profile themes, sound packs, and ringtones
* Screen sharing and live streaming with server audience control
* Spatial audio support for immersive group conversations
* Full third-party app collaboration (Microsoft 365, Google Docs) embedded in app

**Non-Functional Requirements**:

* **Performance**: Low latency with dynamic backend scaling. Lightweight client must operate efficiently with minimal CPU and RAM consumption.
* **Security & Privacy**: End-to-end encryption for private messages. User-controlled privacy settings. All traffic is secured via HTTPS.
* **User Satisfaction**: Implement feedback systems (surveys, ratings). Target at least 90% positive user experience in early testing.

**Technology Stack:**

* **Frontend Frameworks**: React.js (web), Flutter or React Native (mobile), Electron (desktop), C++/C# (Windows lightweight client)
* **Backend Frameworks**: Node.js with Express.js, Socket.IO, NestJS; Go with Gin, Gorilla WebSocket, NATS
* **Languages**: TypeScript, Go, JavaScript, C++/C#
* **Database**: Apache Cassandra; managed options include DataStax Astra DB, Amazon Keyspaces, Azure Cosmos DB (Cassandra API)
* **Redis** for caching and lightweight pub/sub

**Cloud Deployment**

* Google Cloud Run, AWS Fargate, or Azure Container Apps for containerized services
* CI/CD via GitHub Actions, Google Cloud Build, or Docker-based pipelines
* Microservices with autoscaling, stateless design, Redis or NATS for messaging

**Media Servers and Voice Infrastructure**

* Client Audio: Google WebRTC Native Library (C++), Opus codec with adjustable bitrate and complexity
* Voice Signaling: JSON over WebSocket, using libraries like Socket.IO (Node), Gorilla WebSocket (Go), ws or uWebSockets.js
* Media Relay: LiveKit Cloud, Cloudflare Realtime, self-hosted coturn; future support for Jitsi/Janus
* Overlay Rendering: DirectX 11/12 hook using C++ and libraries like DirectXHook

**Monitoring and Testing**

* Monitoring: Prometheus + Grafana, Google Cloud Monitoring, AWS CloudWatch
* Testing: k6, JMeter, Windows Performance Analyzer, Visual Studio Diagnostics, Valgrind

**API and Integration**

* REST and WebSocket APIs for client-backend communication
* gRPC-Web for internal service efficiency (optional)
* Future integrations with Google Docs, Microsoft 365, Notion, Slack, Evernote

**Operating System Compatibility**

* Desktop: Windows 10+, macOS 11+ (Electron/Flutter)
* Mobile: Android 10+, iOS 14+ (Flutter/React Native)
* Lightweight Client: Windows 10+ (native C++/C#)

**Security Protocols**

* OAuth 2.0 authentication with Auth0, Firebase Auth, or custom solution
* JWT for stateless session handling
* HTTPS/TLS for encrypted data transmission
* End-to-end encryption for private messaging (planned)

**Business Requirements**:

* **Product-Market Fit**: The platform must fill the usability gap between Discord’s casual appeal and Microsoft Teams’ productivity. It should cater to gamers, creators, and small teams.
* **Monetization Strategy**: Non-subscription monetization through one-time purchases, including storage upgrades, profile customizations, and premium themes.
* **Customer Retention and Engagement**: Encourage repeat engagement through voice chat hangouts, emote/sticker personalization, and social collaboration tools.
* **Budget and Team Structure**:
  + Project must be delivered under $800,000
  + Team: 2 developers, 2 UI/UX designers, 1 project manager

# 9. Potential Risks

The risks involved are:

* User Adoption Challenge – Many may show resistance to switch from familiar platforms like Discord or Teams, even though our app offers better functionality.

* The influence of the IPO may be limited; if the IPO by Discord fails to bring about significant changes to its users, the urgency to find an alternative platform may not be immediate.

* Competitive feature redundancy poses the risk that either Teams or Discord could bring similar features before or shortly after our release, thus reducing our competitive advantage.

* Monetization Acceptance – Some users may be unwilling to pay for add-ons like increased storage or tailored functions, especially if they have been using them at no cost.

* The feature integration complexity – It necessitates careful design when combining professional and social features so that, together, they can be useable and convenient for multiple user groups.

# 10. Why is this a substantial project that warrants to be a 2-semester project.

Provide a substantiated argument.

The **MyChat communication platform** represents a large-scale, technically complex, and market-sensitive software development effort that goes far beyond the scope of a single-semester academic project. It warrants a **2-semester duration** for the following reasons:

#### ***1. Scope and Feature Set***

The platform aims to merge the community-first experience of Discord with the productivity-focused structure of Microsoft Teams. This includes:

* Real-time chat, voice, and video functionality using **WebRTC** and **Socket.IO**
* Server creation with **role-based and hierarchical permissions**
* **File collaboration tools**, in-app document viewing and editing
* **Opt-in call recording** with individual consent features
* Profile, theme, emote, and sticker customization systems
* Lightweight and full desktop clients with **DirectX-based overlay**
* Integrated **third-party tools** (Microsoft 365, Notion, etc.)  
   This level of functionality matches or exceeds commercial platforms, making it a substantial software project.

#### ***2. Technical Complexity***

The project requires the integration of multiple technologies, including:

* **Frontend frameworks** like React.js and Flutter
* **Backend microservices** using Node.js, Go, Redis, Cassandra
* **Secure real-time communications**, call recording, and media relay servers (LiveKit, coturn)
* Multi-platform deployment (Electron, native Windows app)
* **End-to-end encryption**, authentication via OAuth 2.0, JWT, and compliance with **privacy standards**  
   The architectural planning, implementation, testing, and security hardening of such features demands extensive time.

#### ***3. Multi-Phase Development and Testing***

This project must go through:

* Requirements gathering and system design
* Wireframing and UI/UX prototyping
* MVP development (core chat, call, server creation)
* Internal testing, bug fixing, feedback cycles
* Post-MVP enhancements like recording, file collaboration, third-party integrations
* Performance benchmarking, security testing, and quality assurance

A one-semester timeline would force either incomplete implementation or under-tested, unstable features. A **two-semester schedule** allows the team to properly develop, test, and iterate on a real-world scale platform.

#### ***4. Business Planning and Market Strategy***

Beyond technical development, the platform includes:

* A defined **monetization strategy** (non-subscription, one-time purchases)
* **Market timing goals** (launch during Discord’s IPO hype window)
* **User feedback loops** for refinement and retention
* Strategic planning for **competitive differentiation**  
   These require market research, UX studies, and user engagement plans that must evolve over time, making a longer development cycle essential.

#### ***5. Limited Team and Budget Constraints***

The project has a **small team (2 developers, 2 designers, 1 PM)** and a strict **$800,000 budget cap**. A two-semester timeline enables:

* Better task distribution and workload management
* Time for learning and upskilling where needed
* Controlled feature rollout and budget-aware planning

# MyChat Project Timeline

This is an optimistic timeline for the project features. Actual implementation may take longer for the team, as we will need to learn to use new frameworks and technologies.

**Month 1: Planning & Foundations (Weeks 1–4)**

|  |  |  |
| --- | --- | --- |
| Week | Tasks | Team Involved |
| 1–2 | Requirements finalization (MVP scope lock), project kickoff | PM, All |
| 2–3 | Wireframing + UI/UX design concepts | Designers |
| 2–4 | Technology stack setup (Repo, CI/CD, DevOps tools) | Developers |
| 3–4 | Finalize database schema & microservices structure (Cassandra, Redis) | Developers, PM |

**Month 2: MVP Development Phase 1 (Weeks 5–8)**

|  |  |  |
| --- | --- | --- |
| Week | Tasks | Team Involved |
| 5–6 | User account system (Auth, Registration, 2FA, JWT, Auth0/Firebase setup) | Developers |
| 5–6 | Real-time chat core (Socket.IO setup, message handling, typing indicators) | Developers |
| 6–7 | Server creation tools (basic templates, role management logic) | Developers |
| 6–8 | UI/UX implementation for chat, login/registration pages | Designers, Developers |
| 7–8 | Voice/video call setup (WebRTC + Opus integration, basic room logic) | Developers |

**Month 3: MVP Development Phase 2 (Weeks 9–12)**

|  |  |  |
| --- | --- | --- |
| Week | Tasks | Team Involved |
| 9–10 | In-app file viewing & uploads (PDF, DOCX, image viewer) | Developers |
| 10–11 | Lightweight Windows client (C++/C# base, toast notifications, basic chat UI) | Developer 2 |
| 9–11 | Presence & notification logic (online status, toast notifications) | Developers |
| 11–12 | Storage system with admin quota controls | Developers, PM |
| 11–12 | UI testing & unit tests setup (Jest, k6, JMeter) | Developers, PM |

**Month 4: MVP Finalization & Internal Testing (Weeks 13–16)**

|  |  |  |
| --- | --- | --- |
| Week | Tasks | Team Involved |
| 13 | Internal QA, bug fixing, and integration testing | All |
| 14 | MVP feature freeze. Final UI polish, accessibility passes | Designers, Developers |
| 15 | Prepare installer for Windows lightweight client | Developer 2 |
| 15–16 | MVP Beta Deployment (via Google Cloud Run/AWS Fargate + CDN + DNS) | Developers, PM |
| 16 | Internal beta testing + feedback from small group | All |

**Post-MVP Rollout: Month 5–8**

|  |  |  |
| --- | --- | --- |
| Month | Tasks | Team Involved |
| Month 5 | Call recording system (user opt-in controls, notification logic) | Developers |
| Month 5 | Public server discovery, onboarding UX | Designers, Developers |
| Month 6 | Emotes, stickers, and marketplace system (upload UI, validation, link to admin storage) | Developers |
| Month 6 | Cloud voice relay testing: coturn, LiveKit | Developers |
| Month 7 | Integrations: Microsoft 365 / Notion API testing & embedding | Developers |
| Month 7 | Screen sharing & spatial audio support | Developers |
| Month 8 | Pre-launch user feedback loop, performance benchmarking, security testing | PM, Developers |

**Milestones**

|  |  |
| --- | --- |
| Milestone | Target Date |
| Project Kickoff | Month 1, Week 1 |
| MVP Feature Freeze | Month 4, Week 14 |
| Internal MVP Beta Release | Month 4, Week 16 |
| Staged Feature Rollouts Begin | Month 5 |
| Final Public Beta Launch | End of Month 8 |

#### ***Conclusion***

Given its **breadth, technical depth, market relevance, and real-world applicability**, this project aligns more with a **startup-grade software product** than a typical course assignment. A single semester would compromise quality and depth, whereas two semesters allow for full-cycle development—from idea to stable MVP—ensuring both educational value and project viability.

Therefore, **the MyChat platform deserves to be a 2-semester capstone or senior project** due to its scale, innovation, and complexity.

Works Cited:  
   
“About Discord.” *Discord*, [discord.com/company](https://discord.com/company). Accessed 15 May 2025.

Balasubramanian, Suvedha. “Microsoft Teams Statistics - by Revenue, Demographics, Usage (2025).” *Desk365*, 23 Dec. 2024, [www.desk365.io/blog/microsoft-teams-statistics/](https://www.desk365.io/blog/microsoft-teams-statistics/).

Lath, Anuja. “Discord’s Leadership Shift: Humam Sakhnini Takes Helm as CEO amid IPO Rumors.” *BBN Times*, 23 Apr. 2025, [www.bbntimes.com/technology/discord-s-leadership-shift-humam-sakhnini-takes-helm-as-ceo-amid-ipo-rumors](https://www.bbntimes.com/technology/discord-s-leadership-shift-humam-sakhnini-takes-helm-as-ceo-amid-ipo-rumors).