# Tools for Teams

Anthony Fleury, Dikshyant Gyawali, Lucas Murray, Nathan Steele
*Department of Computer Science, University of New Hampshire, Durham, NH 03824*

## Introduction

As online collaboration tools have become increasingly important in education, the University of New Hampshire's Computer Science (CS) department encountered a challenge when it became clear that its Discord bot did not align with university IT policies. This led to the decision to transition to Microsoft Teams.

Our group, Tools for Teams, is focused on developing a Microsoft Teams bot that fulfills the Discord bot's functionality requirements while adhering to university standards. Our bot ensures students can reliably mark themselves present, TAs receive accurate attendance reports at the end of each lab, and all users can successfully create and join breakout rooms.

## Requirements

**Attendance Tracking: T**he bot must allow students to self-report attendance and generate accurate reports for instructors.

**Efficiency & Accuracy:** Attendance tracking should be easy, efficient, and error-free to minimize distractions and ensure reliable data.

**Breakout Room Management:** Faculty should have the ability to create and manage breakout rooms seamlessly, even for large numbers of users.

**Performance Considerations:** The bot must be scalable to handle high user loads without affecting performance.

**Security & Access Control:** Permissions must be properly managed to ensure that only authorized users can access, modify, or delete breakout rooms and attendance data.

## Project Design

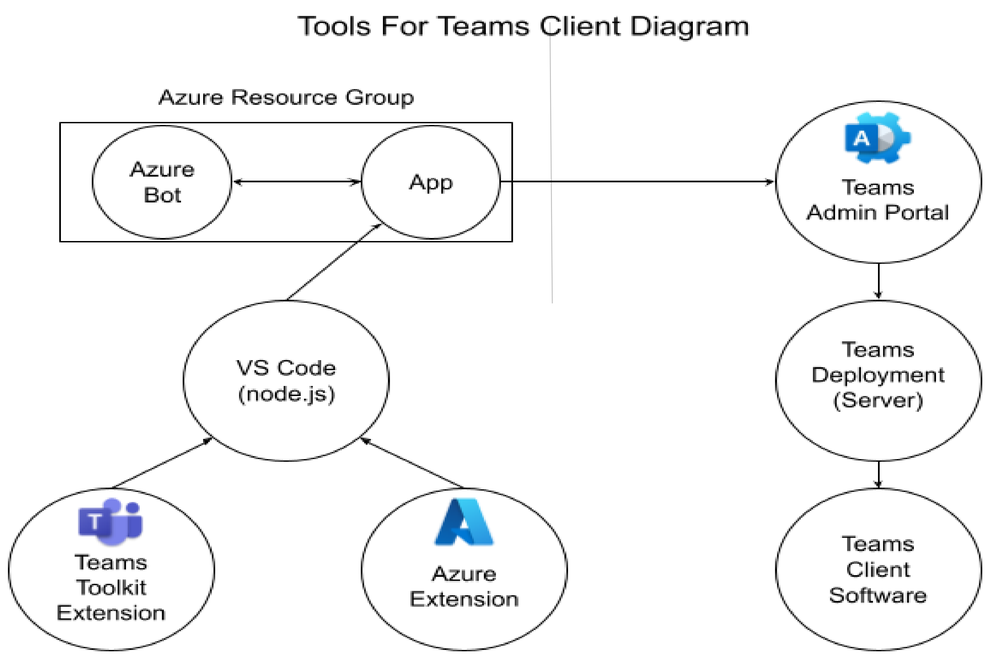
**Sandbox Testing Environment:** We set up a **Microsoft Teams sandbox** with ~20 dummy users to simulate real classroom interactions, with group members and our sponsor having full administrative access.

**Class Channel Simulation:** The sandbox includes **various Teams channels** designed to mimic those in the **UNH CS server**, ensuring realistic testing conditions.

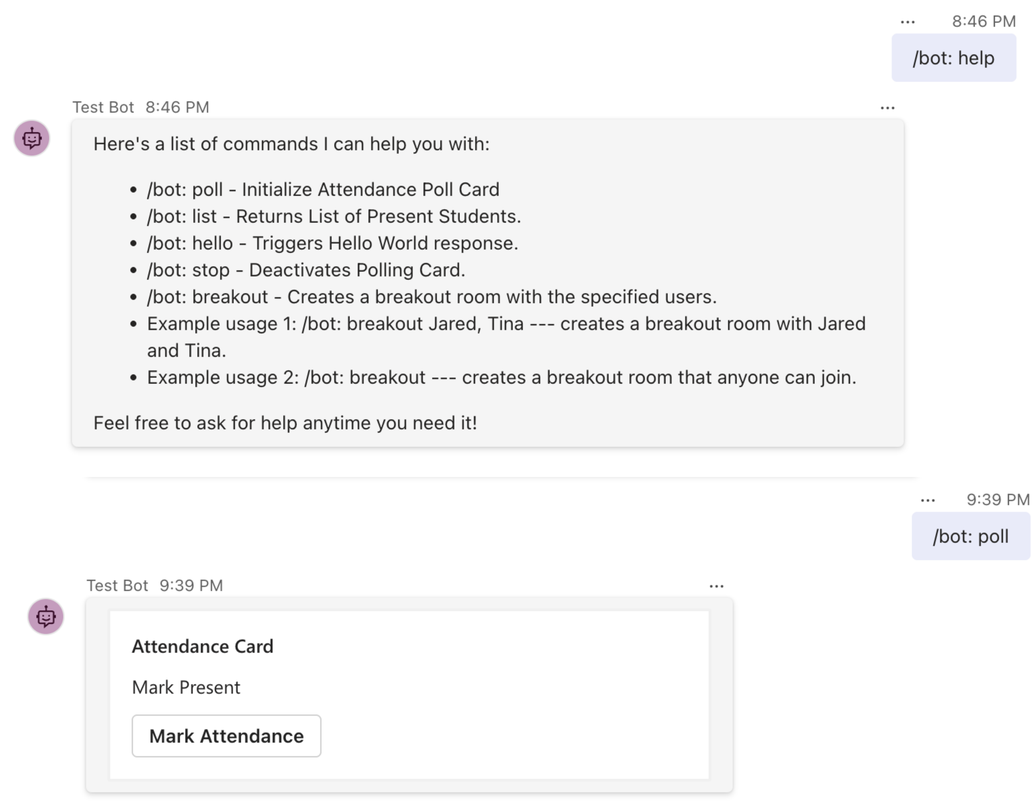
**Bot Command Processing:** The bot listens for messages in **teamsBot.js**, verifies user credentials, and processes commands formatted as <command> <inputs>.

**Command Handling & Responses:** Recognized commands are routed through **attendanceCommandHandler.js** and **genericCommandHandler.js**, which execute the appropriate functions and return responses to the Teams channel.

## Architecture/Sequence Diagrams




## User Interface and Bot Functionality




Screenshot #1 List of bot commands

Screenshot #2 Attendance Tracking Feature

## Implementation/Testing

- Bot tested in a Microsoft Teams sandbox simulating classroom interaction.
- Successfully deployed to Microsoft Azure and integrated into a live Teams environment.
- Functional verification included attendance tracking and message handling.
- Performance checked using Azure Log Stream to ensure system stability.
- Live testing ensures the bot handles real-time Teams interactions reliably.

## Evaluations/Conclusions

- **Attendance Tracking**: Bot records student attendance; instructors receive accurate reports.
- **Deployed & Functional**: Bot is currently live and working in our developer sandbox Teams instance.
- **Breakout Room Feature**: Not yet implemented, but access to Azure and Teams enables future development.
- **Next Steps**: Finalize breakout room functionality, continue deployment efforts within the UNH Teams environment, and develop additional features.
- Our bot currently satisfies two of our three measures of value by ensuring students can mark themselves present with confirmation and TAs receive complete attendance reports. While breakout rooms are still in development, the foundation is in place to meet this final objective.

## Acknowledgements

**Project Sponsor:** Prof. Matthew Plumlee

**Capstone Advisor:** Prof. Craig Smith

**Discord Staff:** Jayme Brannan, Declan Baker