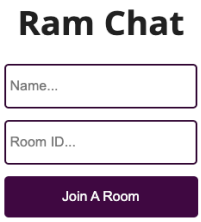
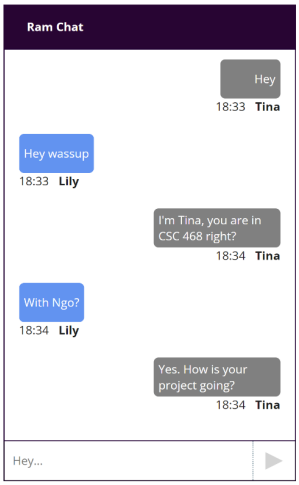
Ram Chat

Brendan Dill
Jessica Gorr
Kiah Johnson
Nathan Parlett
Zharia Washington

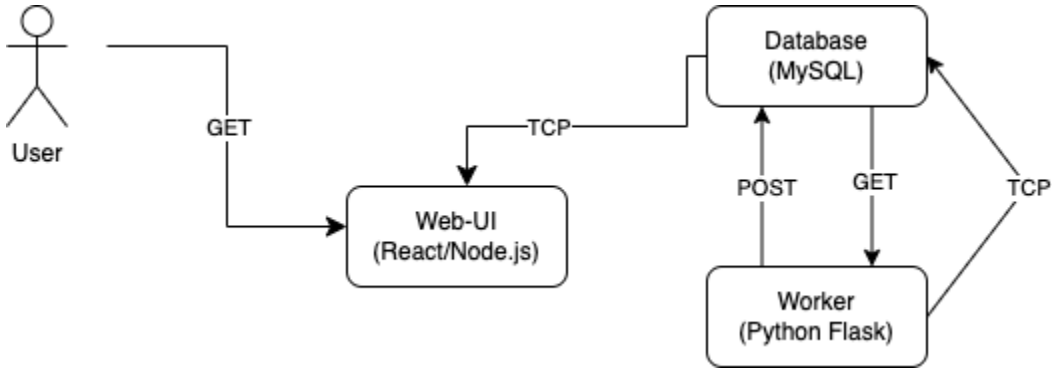
Chapter 1: Team's Vision





Ram Chat is a live chat room application that allows users to communicate in web chat rooms. Users can input their real name or remain completely anonymous; it's completely up to the user's experience. Once a chat room is created the user can share the Room ID with other users and send messages back and forth in the chatroom that is available. Ram Chat serves as a communication application for students to connect with each other socially and academically. In the future Ram Chat can be incorporated with D2L to enhance learning and communication amongst students and professors. In the future we would like to enhance the users experience by enabling functions such as sharing files, images, and supporting video chat within the app itself.

Chapter 2: Implementation



Ram Chat implements the following components (web-ui, worker, and database) together as stated:

WebUI

The web app itself is the only interface the users will interact with Ram Chat. The user interface uses Node.js and React.js to build the front end of the application, which contains a combination of JavaScript and JSX. In addition, there is also a simple server used as well, as it simply works with Socket.IO to establish a base connection between every platform. It detects if the user connects or disconnects to the server, and gives the user an automated ID, which is then saved. This interacts with the worker and the database upon user actions via TCP connections. This allows the web app to communicate with the simple server, the worker, and the database, exchanging and using the data. It relays the user information to the worker and database for storing. For example, any actions taken, such as messages sent or user credentials, will be triggered by an on-click event in the web app that the worker will retrieve and store into the database.

The web app offers the following key features:

1. *User Login*: The application when first opened, the users will be able to type in their name, or an alias if they choose. The web app ensures ambiguity if the user chooses, as they are allowed to pick any name of their choosing. It also has another input field where the user can choose a room ID, which can also be anything of their choosing. This just connects two or more users to the same room if they have the same room ID. The room ID is case sensitive, so if they do not match exactly, the users will not be able to communicate the way they want, for security purposes.

2. *Message Display*: The web app displays a simple messaging app design, so the user is aware of where to type the messages and how to send them. It has a responsive design which ensures the aesthetic is consistent and works across various devices and screen sizes. The user is allowed to type, delete, and edit input from the textbox. In addition, the user can also submit the text they wish to send to others by using their computer mouse or hitting the enter button on their keyboard. Once a message is sent, the user or ‘author’ of the text will have a blue message, while the responder(s) will have a gray message.

3. *Message statuses*: Within the web app once messages are sent, the users will be able to see the messages they sent and receive immediately once they are online. The web app

displays what time the message was sent to the user under where the message was sent, and it is displayed in military time.

4. *Conversation History*: On the web app, the users can see their past conversation history stored on their devices, if they use the same exact room ID as they used before. The chat history can be easily seen by simply scrolling through on the message display.

Worker

The worker utilizes Python Flask to connect to the database. In which the worker can retrieve and store messages, user information, etc. upon user actions. The worker can then communicate with the database via TCP connections, where this will allow the retrieval and storing of user and chat information. It uses a REST API to develop a backend API. Where it can then use GET and POST requests to store and retrieve user and chat information. The REST API provides a great deal of flexibility and security to exchange important information for the application.

```
    from flask import Flask, render_template, request, redirect, url_for, session
    from flask_mysqldb import MySQL
    import MySQLdb.cursors
    import re

    app = Flask(__name__)

    app.secret_key = "secret"

    app.config['MYSQL_HOST'] = 'localhost'
    app.config['MYSQL_USER'] = 'cloud'
    app.config['MYSQL_PASSWORD'] = 'cloud'
    app.config['MYSQL_DB'] = 'CloudChat'

    mySQL = MySQL(app)

def getMessages():
    msg = ''
    if request.method == 'GET' and 'username' in request.form and 'RoomID' in request.form and 'Messages' in request.form:
        username = request.form['username']
        roomID = request.form['RoomID']
        messages = request.form['Messages']
        time = request.form['Time']
        cursor = mySQL.connection.cursor(MySQLdb.cursors.DictCursor)
        cursor.execute(
            'SELECT UserName, Messages, Time FROM RoomMessages WHERE RoomID = RmID', (username, roomID, messages, time)
        )
    return render_template('index.html', msg=msg)

@app.route('/addMessage', methods=['GET', 'POST'])
def addMessages():
    msg = ''
    if request.method == 'GET' and 'username' in request.form and 'RoomID' in request.form and 'Messages' in request.form:
        username = request.form['username']
        roomID = request.form['RoomID']
        messages = request.form['Messages']
        time = request.form['Time']
        cursor = mySQL.connection.cursor(MySQLdb.cursors.DictCursor)
        cursor.execute(
            'INSERT INTO RoomMessages WHERE UserName = %s \ AND Messages = %s \ AND Time = %s' (username, roomID, messages, time)
        )
    return render_template('index.html', msg=msg)

if __name__ == "__main__":
    app.run(host="localhost", port=int("5000"))
```

Database

The database is built using MySQL. It is used to store messages and user information. It communicates with the worker and web-ui via GET and POST requests, where it is then able to communicate and exchange user and chat information.

The database has four tables.

The user table stores usernames, IP addresses, and whether or not the user is registered or a guest.

UserName	IPAddress	UserType
Test	TestIP	Guest

The password table stores usernames and passwords for registered users.

UserName	PasswordHash
Test2	TestHash
Test23	TestHash

The banlist table stores the IP addresses of banned users.

IPAddress
TestIP

The messages table stores user messages and the room each message belongs to. It also stores the username of the poster and the time the message was sent.

RoomID	UserName	Messages	Time
1	tu	Test	2023-05-03 02:30:11

The database also contains many MySQL procedures to make it easier for the worker to interact with the database.

NewGuestUser takes in a username and an IP address and writes them into the Users table along with the string “Guest”.

```
CREATE PROCEDURE NewGuestUser (IN UName nvarchar(30), IPAddr nvarchar(255))
```

NewRegisteredUser takes in a username, an IP address, and a password hash. The username and IP are written into the user table.

```
CREATE PROCEDURE NewRegisteredUser (UName nvarchar(30),  IPAddr
nvarchar(255), PsswrdHash nvarchar(255))
```

The username and password hash are written into the password table. BanUser takes in an IP address and writes it to the banlist table.

```
CREATE PROCEDURE BanUser (IN IPAddr nvarchar(255))
```

IsBanned takes in an IP address and will return, “User is banned” if the user is in the banlist table.

```
CREATE PROCEDURE IsBanned (IN IPAddr nvarchar(255))
```

UserLogin takes in a username and password hash. It finds the row in the password table corresponding to the username and checks if the password hashes match.If there is a match, it will return, “Valid Credentials”.Otherwise, it will return, “Invalid Credentials”.

```
CREATE PROCEDURE UserLogin (IN UName nvarchar(30), PsswrdHash
nvarchar(255))
```

AddMessage takes in a Room ID, username, and message. These are then written into the messages table (the table also stores the time the message was received in the Time column).

```
CREATE PROCEDURE AddMessage (IN RmID nvarchar(30), UName nvarchar(30),
Mssg nvarchar(255))
```

GetMessagesFromRoomN will return all the messages from a specific room. It does this by selecting all the rows in the messages table that have the entered ID. This can be used for displaying the room to users when they first enter the chatroom.

```
CREATE PROCEDURE GetMessagesFromRoomN (IN RmID nvarchar(30))
```

Chapter 3: Building Images

The docker images were all originally made using Dockerfiles and uploaded to DockerHub. The MySQL docker image was pulled from the base image and modified to expose the specific port for the web-ui and the database to communicate. The Dockerfile for the web-ui is utilizing Alpine as a service and incorporates Node.js and JSON. The exposed port is the same as the database, allowing for open communication through TCP connections. Finally, the Dockerfile for the worker incorporates Alpine, and installing pip requests in order to pull from the database.

Currently, our project is able to deploy a container for the web-ui and database. The next step is setting up and deploying a container for the worker. A docker-compose file is currently in development in order to deploy all the containers and services for the project on Kubernetes.

Here is an example of the docker-compose.yaml file that creates the containers for the separate components. There are additional yaml files for services and components. Shown below the docker-compose.yml file is the Dockerfile for the worker component.

```yaml
version: '3'

services:
#webui
        webui:
                build: webui
                image: cloudChat/webui
                container_name: webui
                volumes:
                        - ./webui/logs:/var/logs
                restart: unless-stopped
                ports:
                        - "80:80"
                        - "443:443"
#database
        mydb:
                restart: always
                container_name: mysql-docker-container
                image: mysql
                environment:
                        MYSQL_ROOT_PASSWORD: cloudroot
                        MYSQL_ROOT_HOST: 'cloud'
                        MYSQL_DATABASE: CloudChat
                ports:
                        - "3308:3306"
                volumes:
                        - mysql:/var/lib/mysql
                        - mysql_config:/etc/mysql
volumes:
        mysql:
        mysql_config:
```

```dockerfile
FROM python:alpine
RUN pip install redis
RUN pip install requests
COPY worker.py /
CMD ["python", "worker.py"]
```

Chapter 4: Connecting Components

Initially, getting the MySQL server and the web-ui to communicate with each other has been difficult due to the lack of familiarity with concepts such as containerization and cluster networking. Another difficulty was ensuring that the database and the web-ui were able to ping each other, and it was recognized that different ports were being utilized, disabling communication. The fix was successfully made, and now the two components were able to successfully ping each other. However, the Webui no longer connects directly to the database. Rather, it connects to the worker which mediates communication between the Webui and database. Understanding and practicing these techniques was a priority so they could be recognized sooner rather than at the end of the project. Currently, the project has achieved connecting local containers together on a network.This includes the worker, database, and the web-ui.The worker has been set up to use a REST API to pull the needed information and works as the middle-man in conjunction with the web-ui and database components.

Chapter 5: Final State

Progress

Throughout the duration of building Ram Chat there was a lot of trial and error. Because for most of the group members this was new technology that most weren't familiar with. Through trial and error group members were able to make progress.To start off, the group initially had to get the cloud infrastructure up and running successfully with Kubernetes, CloudLab, Docker, and also add a CI/CD pipeline such as Jenkins for continuous integration and deployment.

As for the user interface our team had some connectivity issues. Ideally when two users enter the same chat room, they would be able to chat freely, but this was not the case, instead a user could only see the message they sent so the connection was broken somewhere, an example of this was when a user would periodically refresh the page the message would disappear.

The database development was fairly simple. The main issues were figuring out the unfamiliar syntax of Mysql, and how to have multiple rooms in one database. This was initially going to be solved by having separate message tables for each room. However, the procedures were unable to dynamically reference a table based on user input. Instead, all the messages are stored in one table with each message having a room ID associated with it. When using procedures like AddMessage the procedure caller can specify which room the message should be added to.

For the worker there were ups and downs as making your own REST API can be challenging. The main struggle for the API being developed was successfully integrating it with Node.js/the front-end so it can communicate between the database and the front end successfully.

Was the Final Deliverable Met? -

The Final Deliverable was not completely met. It not being completely met was due to a couple reasons. One reason why it wasn't completely met was due to not being able to integrate Jenkins in the project completely. This was because of the lack of knowledge of implementing a continuous integration/continuous delivery tool such as Jenkins. As this was an unfamiliar tool. The problem of integrating a tool like Jenkins into the project was developing the proper Jenkinsfile with the correct source code so everything could compile correctly. A main problem encountered while trying to incorporate Jenkins was getting error responses in the console output such as, "`hudson.remoting.ProxyException: io.fabric8.kubernetes.client.KubernetesClientException: container webui/worker/mysql not found in pod ramchat-11-5t3kn-dbqff-rhhwh`." Meaning that the Jenkinsfile wasn't set up correctly to find the proper containers in the pod. The team was able to deploy the project on Kubernetes and have all docker images in the Docker Hub registry. The team was also able to deploy the web-ui, worker, and the mysql database successfully.

Reference

https://github.com/jessgorr01/cloudChat

Brendan Dill

Current Address: 434 S Walnut St., West Chester, PA 19382
https://brdxn.github.io/ - bd985346@wcupa.edu - (610)-306-8108

EDUCATION

Anticipated May 2023

West Chester University, West Chester, PA
Bachelor of Science in Computer Science, *cum laude*
Certificate in Computer Security

- GPA: 3.43

PROJECTS

Fishing Log Application April 2022

- Used Java and MySQL to develop an application where users can log their fishing catches.
- Users can log their catches into a crowdsourcing database where they can access the information and use it for their next fishing trip.

Developed a portfolio website (https://brdxn.github.io/)

- Developed using NodeJS, ReactJS, Bootstrap, JSX, CSS, and HTML.

WORK EXPERIENCE

Software Engineer Intern - *ATPCO*, Dulles, VA August 2022 – Present

- Technologies used during my experience were Java, Springboot, Redis, MongoDB, PostgreSQL, IBM DB2, AWS, Kubernetes, Docker, and COBOL.
- Contributed with the migration to MongoDB.
- Converted a COBOL/DB2 program to Java in order to work with PostgreSQL.
- Used AWS EKS and microservices to develop a self-service application.
- Conducted smoke tests, unit tests (JUnit), and integration tests.

Server - *Cheesecake Factory*, King of Prussia, PA March 2021 – May 2022

- Used excellent customer service, communication, and group member skills.
- Processed payments and aimed for at least $1200 in sales per a shift.
- Trained new employees and showed them their responsibilities.

SKILLS

- Java, Springboot, Python, and Apache PySpark
- AWS, Kubernetes, Helm, and Docker
- JavaScript (NodeJS and ReactJS)
- SQL (SQL: MySQL, PostgreSQL, PL/SQL, TSQL; NoSQL: Redis, MongoDB)
- HTML, CSS, Bootstrap, C Programming, Jenkins, and Git
- Cybersecurity, Big Data Engineering, Agile, and Batch Processing

Jessica Gorr

Exton, PA | (610) 836-2943 | Jessgorr01@gmail.com | https://www.linkedin.com/in/jessica-gorr-973b40192/

Education

BACHELOR OF SCIENCE | MAY 2023 | WEST CHESTER UNIVERSITY **MAY 15, 2023**
MAJOR: COMPUTER SCIENCE

- Minor: Applied Statistics, ABET Certificate in Cybersecurity
- Dean's List (GPA per semester > 3.670) fall 2019 – spring 2023.
- Member of Upsilon Phi Epsilon Honors Fraternity
- PSAC Scholar Athlete Award- Years 2019-2020, 2020-2021, 2021-2022
- IWLCA Academic Honor Roll (Cumulative GPA > 3.5, through minimum six semesters)

Work and Leadership Experience

SOFTWARE DEVELOPMENT INTERN | IBM | MAY 2022 – AUGUST 2022

- Streamlined and automated processes using JIRA and Python OOP practices to reinvent the current dump management program.
- Implemented several test plans and coordinated testing for projects throughout the development cycle and provided accurate documentation of limiting factors in the code.
- Lead communications between beta testing groups and outlined future improvements to next team using various platforms such as GitHub, Linux, Slack, PowerPoint, etc.
- Ascertained credentials pertaining to data science using R, Python, and Statical Analysis techniques.

TECHNOLOGY INTERN | SAINTS PHILIP AND JAMES SCHOOL | JUNE 2020 – AUGUST 2021

- Instructed students on software programs such as: Microsoft Excel, Office, & PowerPoint, programming skills, typing, & basic components of the computer and how they work.
- Facilitated the creation of current curriculum for students at each grade level according to the standards set by the Archdioceses of Philadelphia.
- Troubleshooted any technology issues within the school pertaining to hardware and software including laptops, smartboards, servers, and programs.
- Redesigned school website and refreshed school website according to feedback from faculty and upkept with current events of the school and achievements of students.

WOMEN IN STEM WORKSHOP | WEST CHESTER UNIVERSITY | OCTOBER 2021, 2022

- Mentored students in various STEM fields in areas pertaining to computer science, mathematics, and computer engineering with stations and continued communication with outreach programs.
- Designed multiple factor experiments and analysis plans for young women to perform and learn skills in statistics, focusing on t-tests and ANOVA analysis.
- Instructed students on analysis techniques and useful programs such as R, SPSS, and SAS to facilitate the analysis and how to communicate results effectively to an audience.

ADDITIONAL SKILLS

- Programming languages: Java, Python, R, C, SQL, HTML, CSS.
- LARC I Tutor Certification – October 2020.
- Member of West Chester University Varsity Women's Lacrosse Team (August 2019-2023).
- Captain's Leadership Advancement Series – Fall 2022

KIAH JOHNSON

Contact

kiahj003@gmail.com
@Kiah Johnson
(267) 239-6995
Philadelphia, PA

Skills

Proficient
- MATLAB
- Java
- C++

Familiar
- JavaScript
- Python
- HTML/CSS

Relevant Software

GitHub
Kubernetes
Bootstrap
Wireshark

Relevant Courses

- Computer Algebra
- Computer Science I, II, & III
- Computer Security
- Computer Systems
- Data Communications & Networking I
- Data Structure & Algorithms
- Intro to Cloud Computing
- Modern Malware Analysis
- Programming Language Concepts/Paradigms
- Software Engineering
- Software Security
- User Interfaces

Education

West Chester University of Pennsylvania
Bachelor of Science in Computer Science, Minor in Mathematics, Certificate in Computer Security NSA/ABET
- Cumulative GPA: 3.689 | Expected Graduation: May 2023

Honors & Awards

- College of the Sciences & Mathematics Dean's List: Fall '19, '20, & '22; Spring '20 & '21
- PSAC Scholar Athlete: '19-'20 & '21-'22 Seasons
- All America Scholar Athlete: '19-'20 & '21-'22 Seasons

Professional Experience

Gymnastics Coach *April '17 - Present*
Philadelphia Gymnastics Academy; Philadelphia, PA
- Part-time as needed work, to instruct and coach gymnasts between the age of 2-16 in both recreational and team capacities
- Safety & Risk Management Certified through USA Gymnastics

Projects & Notable Coursework

Cloud Computing: Student FAQ Chatbot
- Worked with Python, JavaScript, MySQL, & HTML/CSS to create a Cloud-base Chatbot Application

Modern Malware Analysis: Malware Analysis Tool
- Utilized Windows XP environment, Virtual Box, WinDbg, Rootkits, & Volatility
- Built a runnable Python Script to determine if a PC is infected by its memory dump & built a dynamic heuristic analysis tool to detect unknown malware

Software Security: Vulnerability Attacks
- Utilized Unix/Linux & Command Line Shell
- Exploited attacks such as Stack Buffer Overflow, Return-Oriented Program, SQL Injection, & Session Hijacking Vulnerabilities

Activities

Women in Business Member *Jan '22 - Present*
West Chester University of Pennsylvania
- Attend bi-weekly meetings to develop professional knowledge & create connections with other students

Women in Computer Science Member *Sept '19 - Present*
West Chester University of Pennsylvania
- Attend bi-weekly meetings to discuss relevant information & create connections with other students

NCAA Women's Gymnastics Team Member *Aug '19 - Present*
West Chester University of Pennsylvania
- Devote 17.5-20 hours weekly for practices & competitions
- Developed skills such as leadership, teamwork, & time management

Nathan B. Parlett

1254 Bowman Ave | West Chester, PA 19380 | 610-436-4921 | NP983807@wcupa.edu

EDUCATION

West Chester University, Pursuing a Bachelor's Degree In **Computer Science** & Certificate In Computer Security. **GPA** 3.75
2021-Present, Expected Graduation **May 2023.**
Delaware County Community College,
2019-2021

TECHNICAL SKILLS

Java, C, Python, C++, Object Oriented Programming, Linux, Unified Modeling Language, Statistics, Calculus.

CLASS HIGHLIGHTS

CSC 231 Computer Systems: Learned the basics of C, Assembly, and computer hardware. Transcribed assembly code into C.

CSC 472 Software Security: Analyzed assembly code in depth. Learned about the operation of the stack. Utilized stack overflow, return oriented programming, and ASLR bypass via GOT overwrite in hacking labs. Used python to write exploit scripts.

CSC 401 Software Engineering: Learned higher principles of Object Programming and Object Oriented Design. Implemented design patterns. Learned Unified Modeling Language.

CSC 301 Cyber Security and Ethics: Discussed and wrote about important ethical issues related to privacy, technology, and Cyber Security. Gave group presentation on smart home technology.

INTERESTS

Linux, Programming, Cyber Security, Open Source Software, Piano, Music Theory, Philosophy.

WORK HISTORY

United Parcel Service August 2019 - Present
Package Handler

- Handle customer packages during shipping process.
- Load Trucks, Sort Packages, Tend to Belts.

Chic-fil-A September 2017 – July 2018
Cashier:

- Take and process customer orders, Resolve issues with customer orders.

Zhayria Washington

Exton, PA | 610-818-9870 | Zhayriaw80@gmail.com

Education

B.S COMPUTER SCIENCE | JUN 2023 | WEST CHESTER UNIVERSITY

- ABET Certificate in Cybersecurity
- PSAC Scholar-Athlete 2020-2021, 2021-2022
- Resident Assistant Advisory Board 2022-2023
- Captain of Women's Track and Field 2022 -2023

Coursework

PROGRAM LANGUAGES

- C, Java, Java Script, Python,

SOFTWARE SECURITY:

- Performed Session Hijacking, SQL Injection, and Stack Buffer Overflow attacks
- PowerShell Commands Unix/Linux

MALWARE: ANALYSIS

- Utilized heuristic analysis tools such as winDbg and Rootkits to detect unknown malware
- Set up a virtual machine and develop Python Script to detect infected files

CLOUD COMPUTING: RAM CHAT

- Utilized Python, SQL, and React to develop the front and backend of a university chat application.
- Connected containers for each component and use Kubernetes for automation

Experience

MOBILE TECHNICIAN | ASURION | MAY 2022 – AUG 2022

- Functional test on handsets to ensure correct diagnoses
- Cosmetic repairs
- Specialized repair of Apple, Samsung, and Google mobile devices
- Troubleshoot and update software

MOBILE EXPERT | T-MOBILE | AUG 2021 – MAY 2022

- Delivering the Un-carrier experience by promoting sales and service expertise
- Side-by-side selling to attain personalized solutions
- Troubleshoot hardware and software issues
- Data recovery and Data transfers