TA Queue

Sam Lightfoot, Pasha Ranakusuma, Omar Elgeoushy, Thejus Unnivelan

What is TA Queue?

- Learning management system catered specifically for Virginia Tech
  - Utilities and tools that can be managed and altered at the Department level
  - Dedicated servers for Virginia Tech courses
  - Intuitive course structure with file management system
  - Seamlessly integrated channels of communication and familiar UI system

History of Learning Management Systems 1

- Paper Agenda
  - Great for young children
  - Easy for parents to check progress on
  - Not accessible online/easily lost
- Simple websites
  - Static HTML sites
  - Great for easy design, if HTML is known
  - Not accessible for all people, and requires internet access

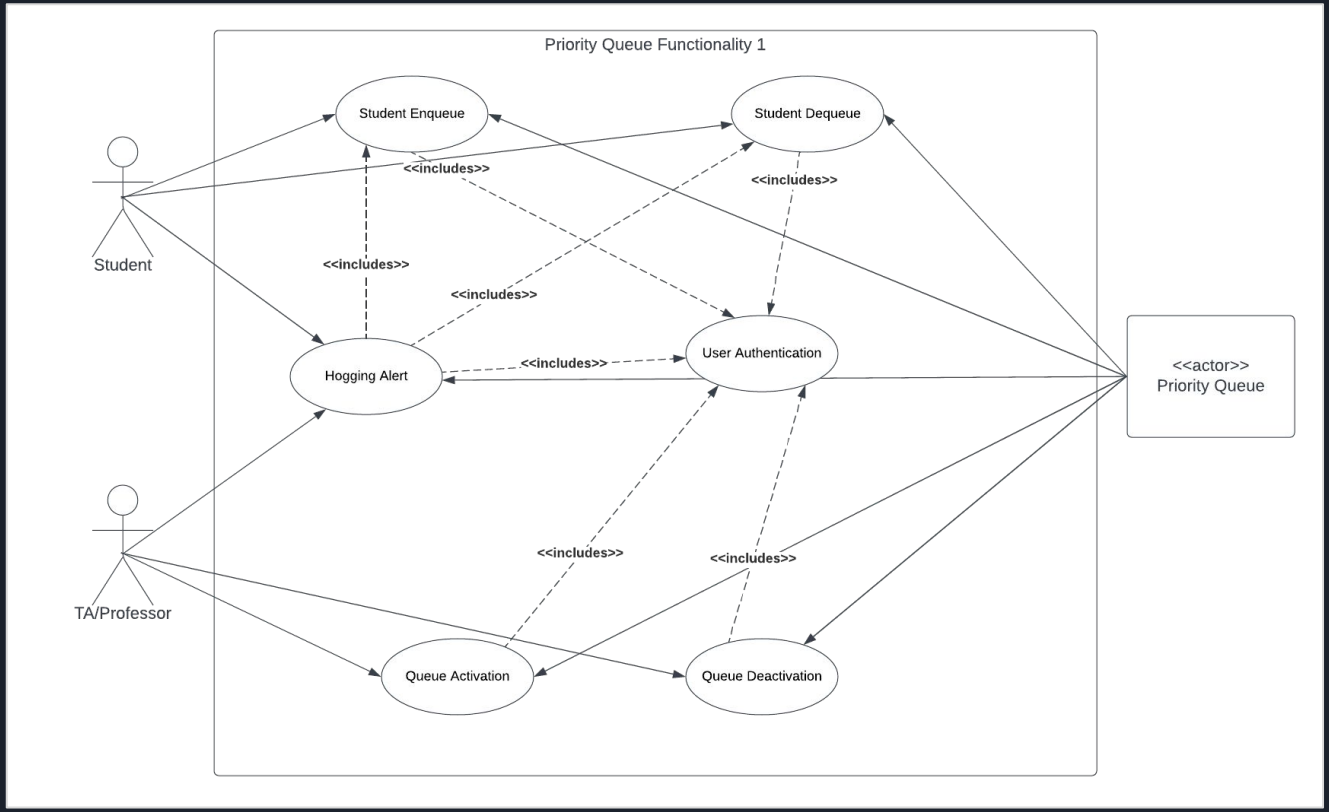
History of Learning Management Systems 2

- Blackboard
  - More intuitive system for all faculty
  - Not appealing nor easily navigable, with limited file sharing and plugins
- Canvas
  - Top of the line
  - Widely adopted
  - Easily accessible - Mobile apps
  - Limited channels of communication
  - No organization in terms of administering office hours

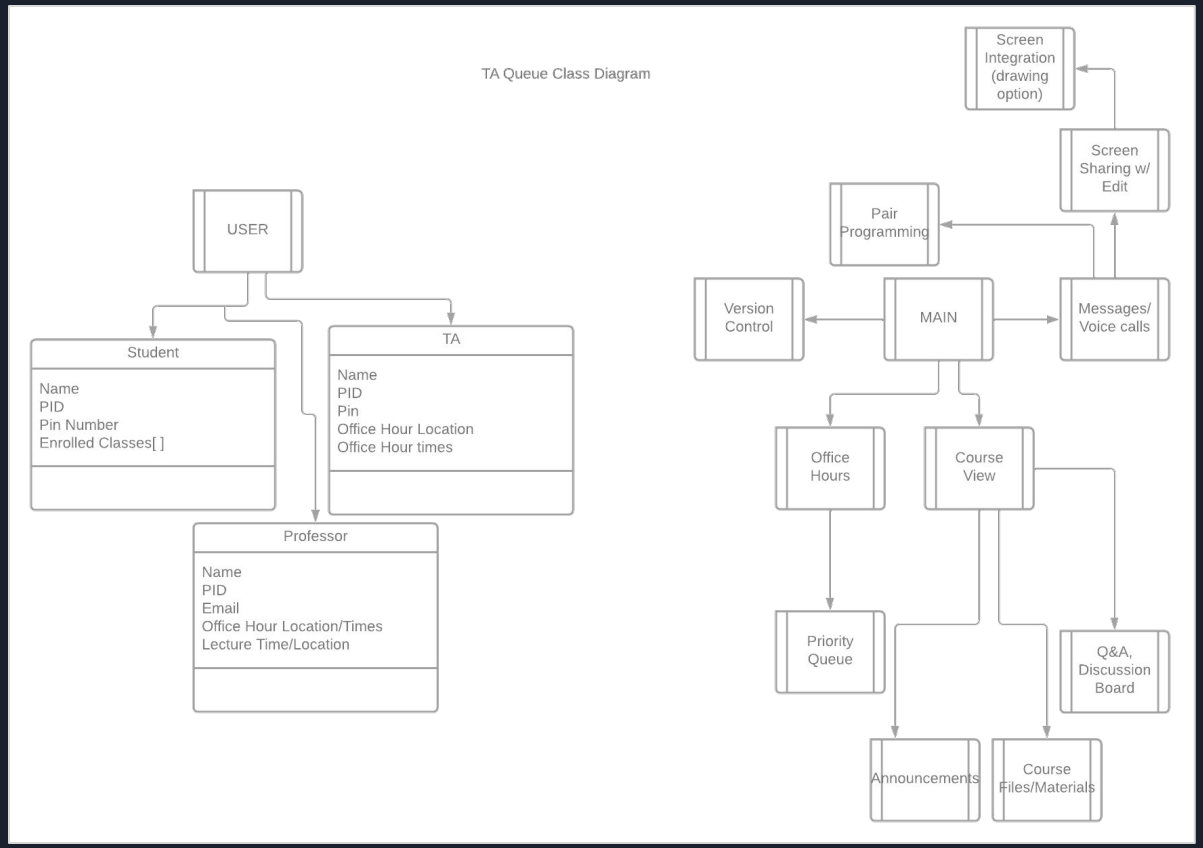
Why TA Queue

- What it borrows
  - Similar UI to currently-used platforms Canvas, Discord, GroupMe, Piazza, Slack
- What is new
  - Complete combination of all other systems
    - Canvas
    - Communication Channels
    - Hokie Spa
  - Addition of Priority Queue system to bring order to office hours

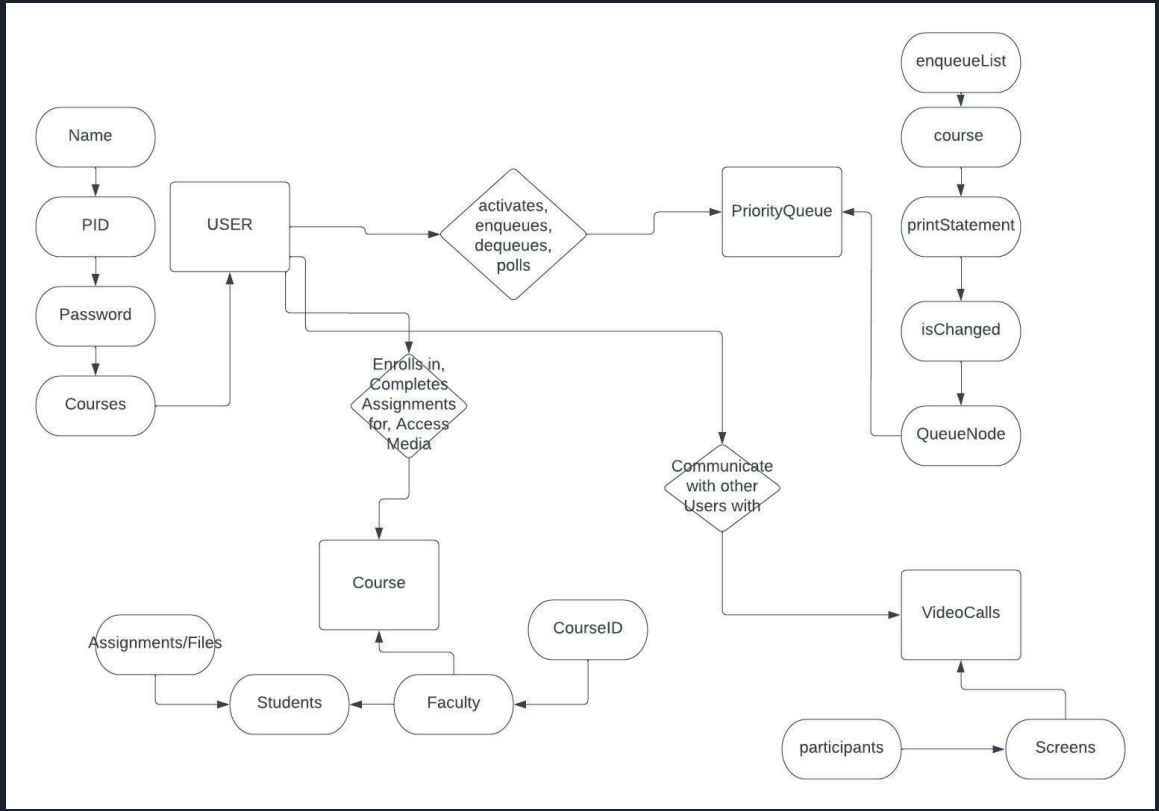
Simple Use Case Diagram for Priority Queue System



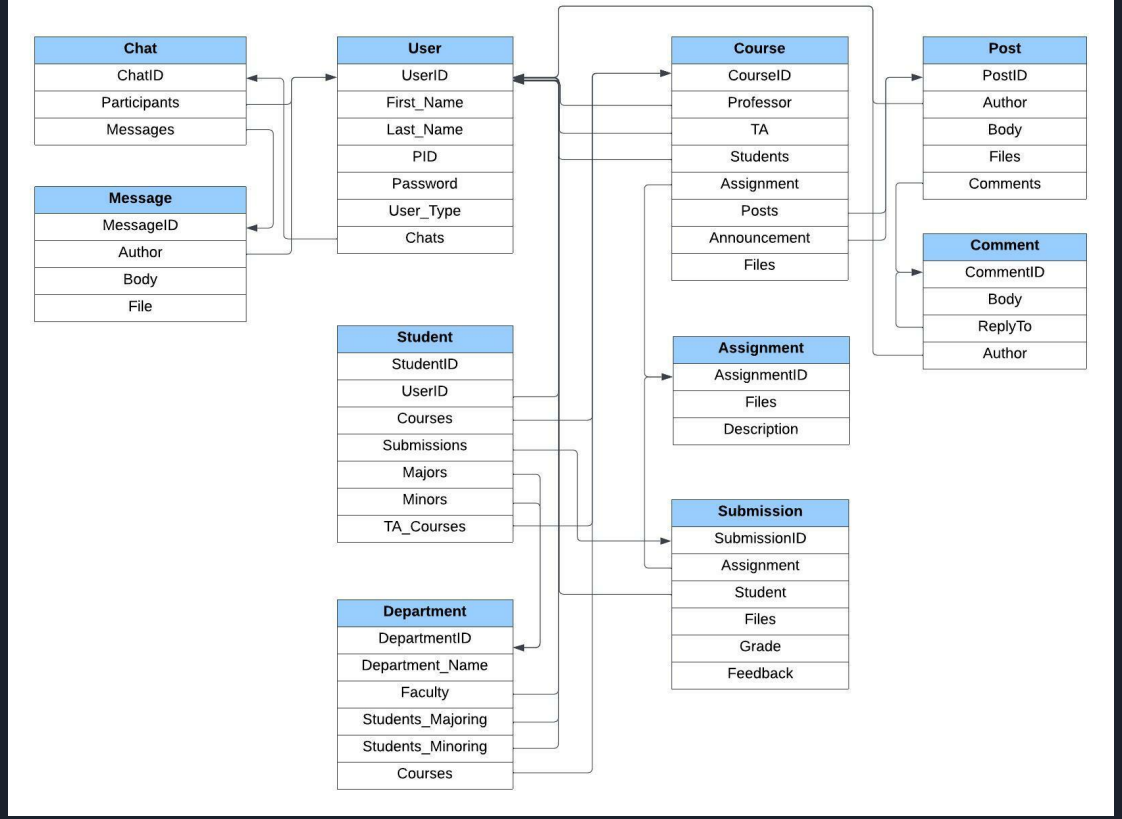
Class Diagram



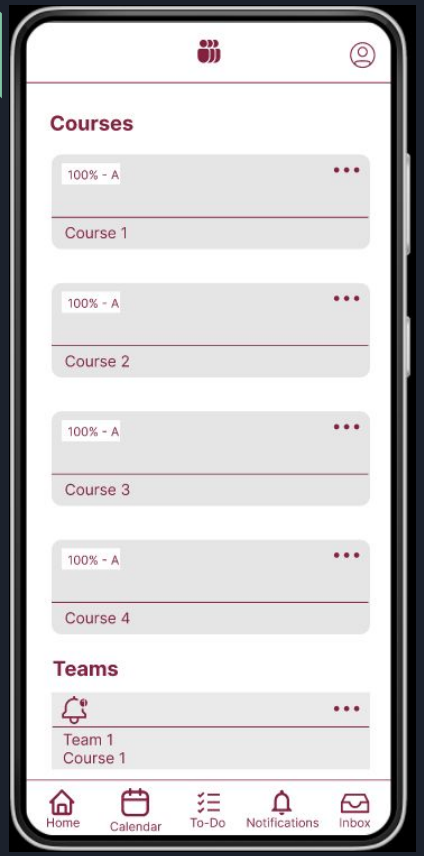
Entity-Relationship Diagram

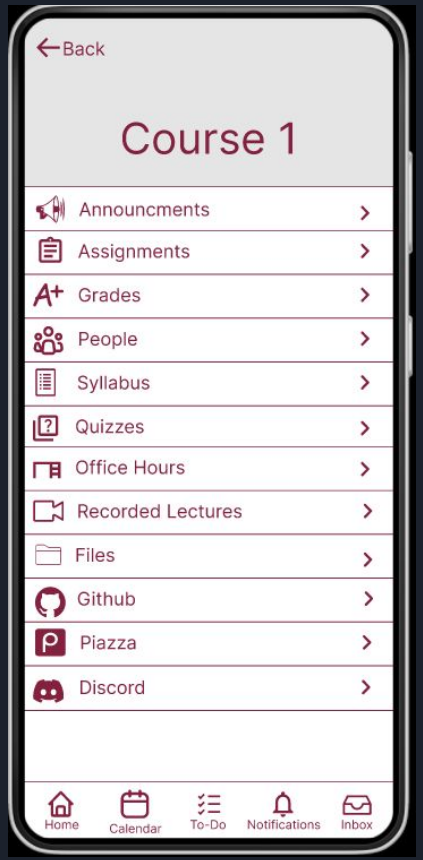


Data Tables


Chat
ChatID
Participants
Messages
Message
MessageID
Author
Body
File
User
UserID
First_Name
Last_Name
PID
Password
User_Type
Chats
Course
CourseID
Professor
TA
Students
Assignment
Posts
Announcement
Files
Post
PostID
Author
Body
Files
Comments
Comment
CommentID
Body
ReplyTo
Author
Student
StudentID
UserID
Courses
Submissions
Majors
Minors
TA_Courses
Assignment
AssignmentID
Files
Description
Submission
SubmissionID
Assignment
Student
Files
Grade
Feedback
Department
DepartmentID
Department_Name
Faculty
Students_Majoring
Students_Minoring
Courses

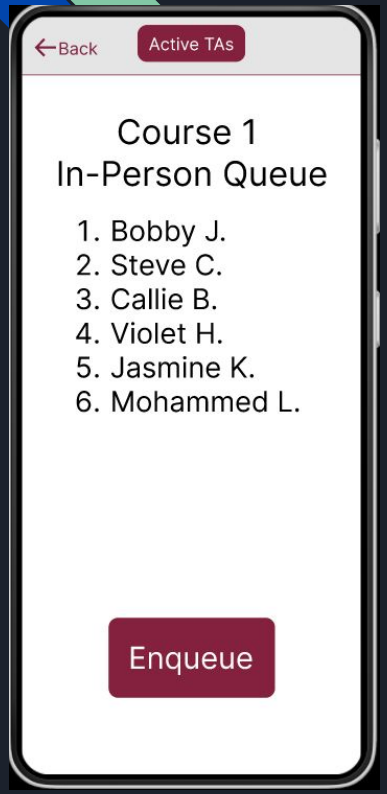
Web App UI

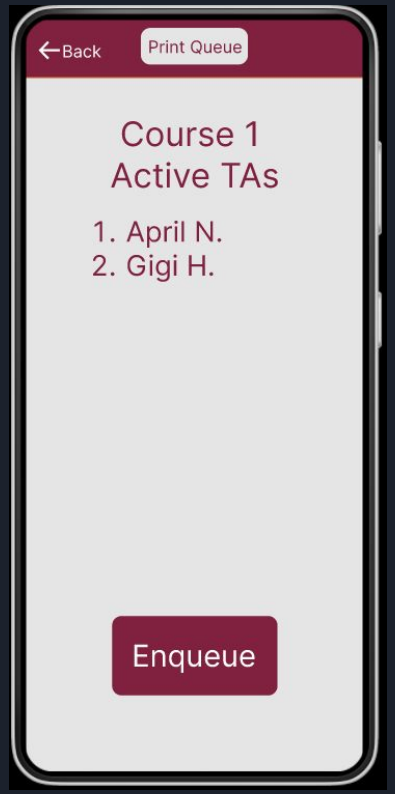

Courses
100% - A
Course 1
100% - A
Course 2
100% - A
Course 3
100% - A
Course 4
Teams
Team 1
Course 1
Home
Calendar
To-Do
Notifications
Inbox

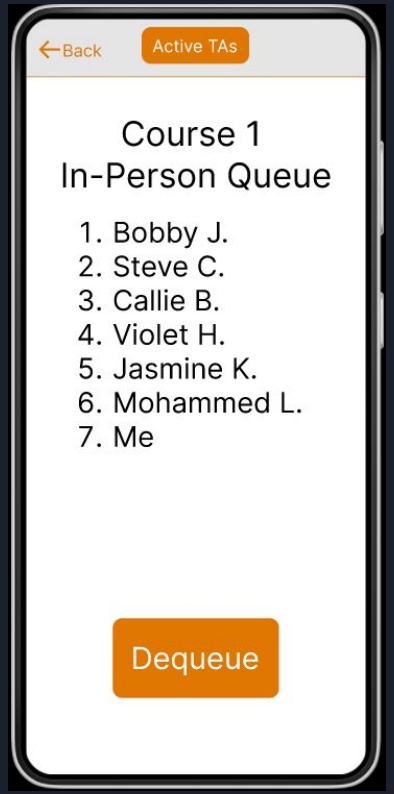

Back
Course 1
Announcments
Assignments
Grades
People
Syllabus
Quizzes
Office Hours
Recorded Lectures
Files
Github
Piazza
Discord
Home
Calendar
To-Do
Notifications
Inbox

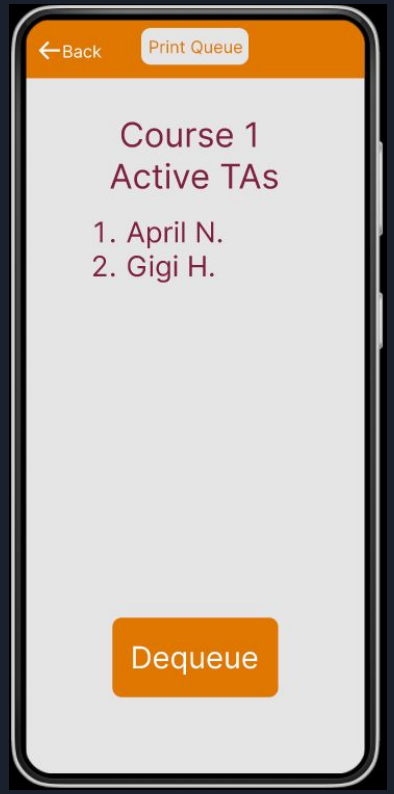

Back
Team 1
Jim L.
Hey!
Hello! When is the assignment due?
Me
Kyle C.
In two weeks
Me
We better start early

Priority Queue UI (Student View)

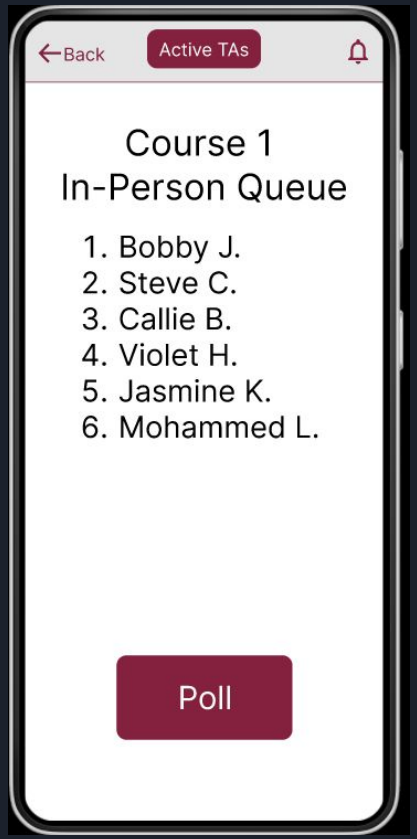


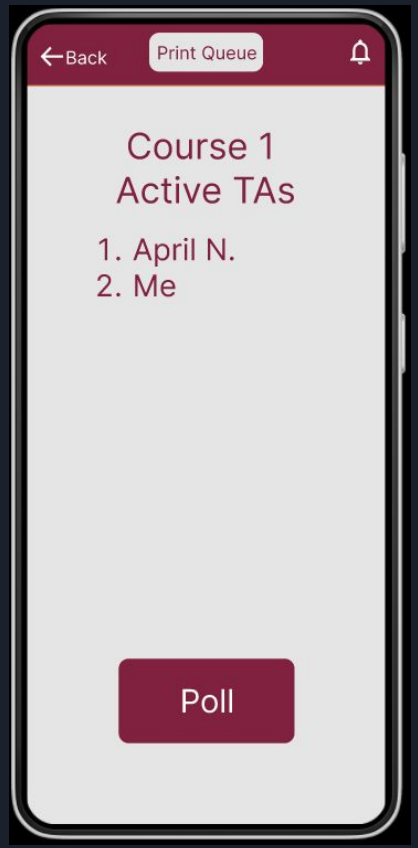






Priority Queue UI (TA View)





Screen Sharing

```
// Use of zero-parameter Lambda Expression
case "0":
    empty.run();
    break;

// Use of single-parameter Lambda Expression
case "1":
    String message = writeMessage.run("Tada!  This is the result of
    another Lambda Expression");
    message = exclaim.run(message);
    System.out.println(message);
    break;

// Use of two-parameter Lambda Expression
case "2":
    int a = 99;
    int b = 101;
    int product = multiplication.run(a, b);
    System.out.println("The product of " + a + " and " + b + " is "
    + product);
```

Screen Sharing (Different Menus)

```java
// Use of zero-parameter Lambda Expression
case "0":
    empty.run();
    break;

// Use of single-parameter Lambda Expression
case "1":
    String message = writeMessage.run("Tada!  This is the result of
    another Lambda Expression");
    message = exclaim.run(message);
    System.out.println(message);
    break;

// Use of two-parameter Lambda Expression
case "2":
    int a = 99;
    int b = 101;
    int product = multiplication.run(a, b);
    System.out.println("The product of " + a + " and " + b + " is "
    + product);
```


```java
// Use of zero-parameter Lambda Expression
case "0":
    empty.run();
    break;

// Use of single-parameter Lambda Expression
case "1":
    String message = writeMessage.run("Tada!  This is the result of
    another Lambda Expression");
    message = exclaim.run(message);
    System.out.println(message);
    break;

// Use of two-parameter Lambda Expression
case "2":
    int a = 99;
    int b = 101;
    int product = multiplication.run(a, b);
    System.out.println("The product of " + a + " and " + b + " is "
    + product);
```

Algorithm Design (Heapify)

- Highlight the priority queue system
  - Priority queue uses a min-heap structure in its design

```
heapify() : void
    for (int i = queue.size() / 2; i >= 0; i--):
        siftDown(i)
```

Algorithm Design (siftDown)

```
siftDown(int pos) : void
    while (!isLeaf(pos)):
        int j = 2*pos + 1
        if (j > queue.size()):
            break
        if (j < last) && (arr[j].compareTo(arr[j + 1] > 0)):
            j++
        if (arr[pos].compareTo(arr[j]) <= 0):
            return
        swap(arr, pos, j)
        pos = j
```

Major Challenges & Solutions

- Integrating all systems into an intuitive structure
  - Easy-to-use interface
  - Great server-side organization and use of data structures
- Cross-platform capabilities
  - Developing systems as a web-app; adapting to different OSs and generations
  - Testing before release, being aware and responsive of user-feedback throughout lifetime
- Debugging the entire system
  - Rigorous testing with very large roster sizes for classes, students, course faculty, etc.
    - n >= 100,000

Summary

- TA Queue is a revolutionary tool for higher education, specialized for Virginia Tech
- Uses cutting edge data structures and channels of communication
  - Efficient algorithms
  - Simple class structure
  - Tight-knit linkage

GitHub

https://github.com/Intermediate-Software-Design/ISD-Group-Project