

# Presentation

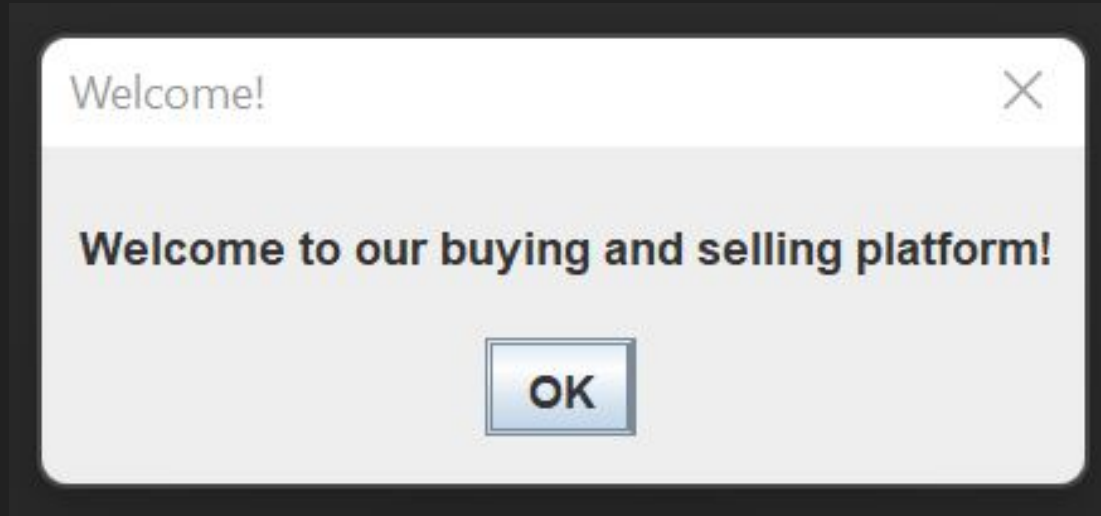
Your team will record and submit a video presentation as part of this project. The requirements are as follows:

- Each team member must actively contribute to the presentation for at least two minutes.
- The overall presentation will be a minimum of 10 minutes and a maximum of 15 minutes.
- The presentation must include the following:
  - Community Impact (Samson)
  - An overall pitch for the project (as in, try to sell the product to a prospective client). (Lincoln)
  - A demo of the project's required functionality. (Arthur)
  - A demo of any optional features. (Arthur)
  - An explanation of the testing done to ensure the project works reliably. (Alex)
  - An audio-visual component (Powerpoint, Google Slides, Prezi, movie, etc.)
  - 2 minutes of time allotted to an FAQ section of questions about the project and your implementation. Answer 2-3 questions a recruiter or business manager would be interested in. (Teresa)

# Marketplace Messaging System

Alex Roth, Arthur Ruano, Samson Tesfagiorgis,  
Teresa Wan, Yulin Lin

# Sales Pitch



Scalability - Modularity - Event-driven

# Java.nio.Selector

- “A **multiplexor** of SelectableChannel objects.” - Javadoc SE 18
- A channel that inherits SelectableChannel can **register** itself on the Selector and supply flags of types of events it will watch for - OP\_CONNECT, OP\_ACCEPT, OP\_READ, OP\_WRITE
- Selector will “select” the channel if the specific event is detected
- The selector.select() method will block until a channel is selected and selector.selectedKey() will provide a set of SelectionKey corresponding to each channels along with the flag detected for each
- Essentially, selector allows for multiple clients to be handle in a **single thread** with the cost that each channel at all time must execute the **same lines of code** for each IO operation and a **complete separation** of Read and Write operation(henced event-driven)

```
While (selector.isOpen) {
```

```
    Wait for selection
```

```
    For (each selected key)
```

```
        If (OP_CONNECT)
```

```
            Handles client connection and spawn and register SocketChannel
```

```
        If (OP_READ)
```

```
            Read and process the ByteBuffer and put the response in the  
            ArrayBlockingQueue attached to the SelectionKey
```

```
        If (OP_WRITE)
```

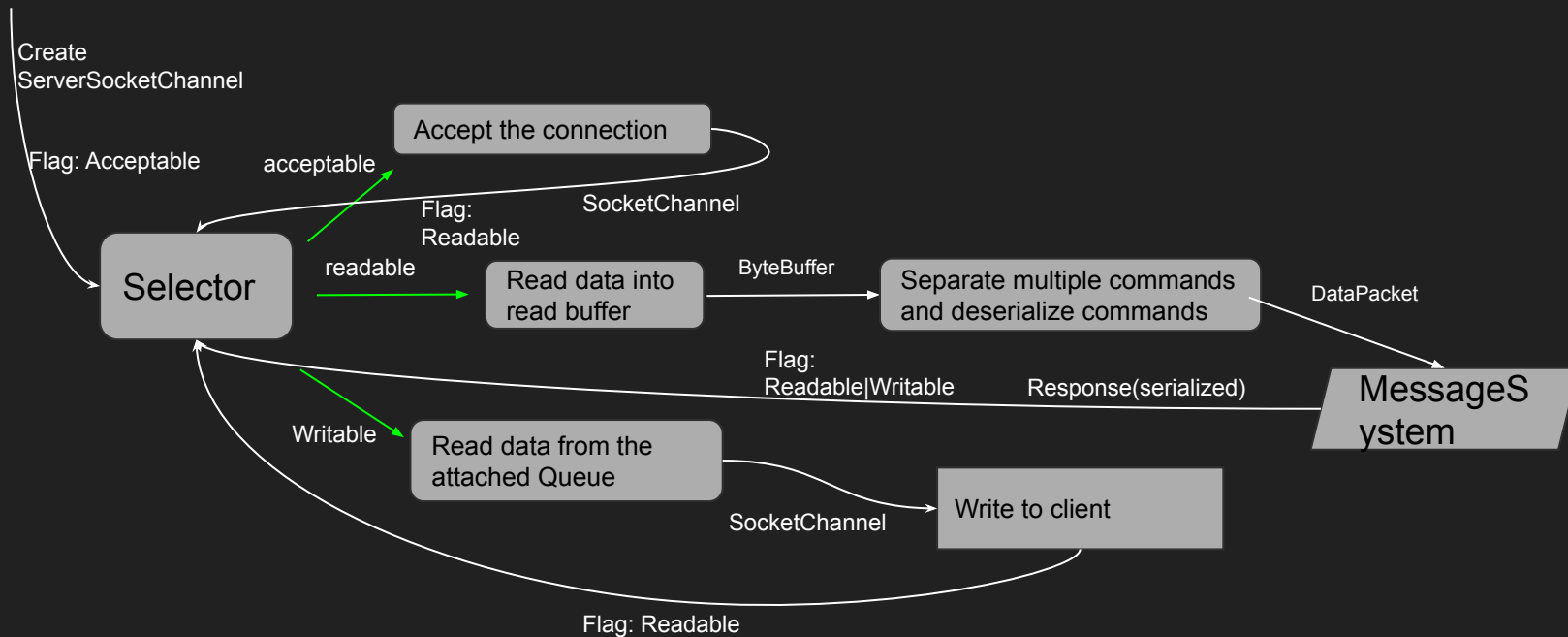
```
            Pop from Queue and write it to the client
```

# NotificationFactory

- Handles communication between channels
- Spawn a thread that add a specific bytearray to the designated channel or broadcast it to all channels

# Server Model

Start

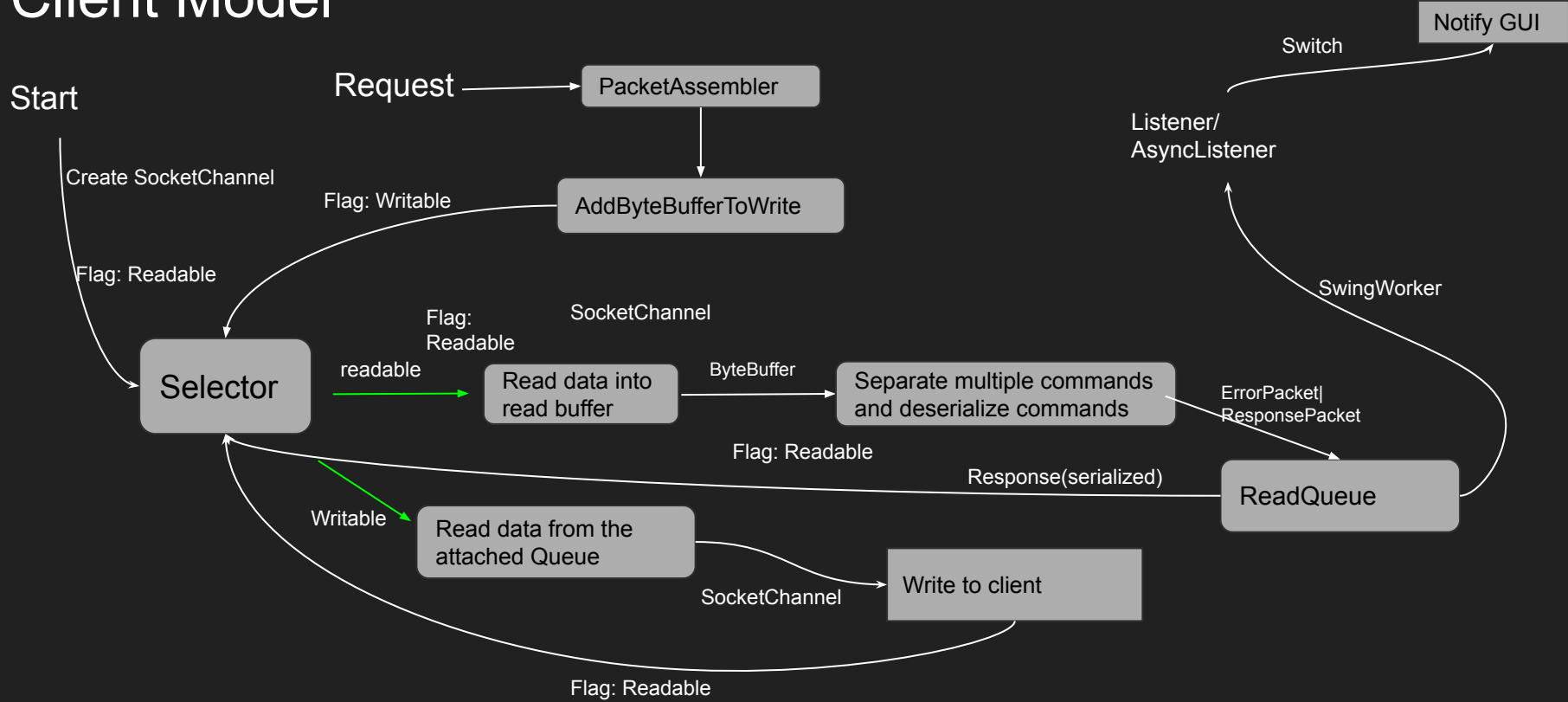


# javax.swing.SwingWorker<T,V>

- “An abstract class to perform lengthy GUI-interaction tasks in a background thread.” - Javadoc SE 18
- SwingWorker class allows the programmer to spawn a thread running in the background and have the EDT monitor it
- Basically spawning a separate thread and a listener attach to it so edt will know its states and get notified when it's done
- In the client model, swing worker is being used to listen to the client read queue and return the queue to the done() method(the done() method runs on the EDT and calls specific functions to update the screen)
- Since listeners simply blocks until the read queue has a packet, doing it inside the EDT would make the GUI unresponsive



# Client Model



# Real-Time Update


- The nature of event-driven in both client and server allow for real-time update of the client
- Once the server receive the packet and processing it in MessageSystem class, it uses notification factory to send update packet to the target client.
- When the target client receive the new conversation, the listener will catch it and invoke the method to rerender the screen

-DEMO-

Someone pull up the project on their computer and screen record and talk

# Testing

# Community Need

 Purdue Marketplace



Mute



 1,069 Members



- Promotes Small Businesses
- Connects Community
- Availability



[purdue.edu](http://purdue.edu)

# FAQ

- What is the cost?
- How well does it scale?
- How maintainable is the code?

Thanks for  
Listening!

