SmartClass

sddec25-10

Team Members:
Josh Dwight, Logan Pfantz, Matthew Gudenkauf, Michael Becker, Michael Geltz, Ryan Lin

Client/Advisor:
Maruf Ahamed

01
SmartClass Mission

SmartClass' Mission

- **Empower educators and students** by bridging communication gaps in large lecture settings
- **Strengthen classroom relationships** through enhanced interaction tools
- **Increase student engagement** with real-time feedback mechanisms
- **Develop a scalable platform** to support diverse educational environments

What is SmartClass?

What are we creating?

- An interactive classroom platform that connects students and teachers
- Real-time polling and Q&A system for immediate feedback
- Interactive lectures activities to ensure student engagement
- Discussion space that persists beyond class time
- Analytics dashboard for tracking student engagement

Who are our intended users?

- Students
  - Shy - Doesn't like to ask questions
  - Confused - Professor went to fast through content
- TA
  - Wants to help students in/outside of the class
- Professors
  - Large classroom
  - Wants to make lectures engaging
  - Wants feedback

Development Style

- Agile Workflow
  - Weekly faculty advisor meetings
  - Weekly meeting with all team members
  - Decided what needed to be implemented & what needed change

- Sprint Work Week
  - 1 week long sprints
  - Utilized Linear for organization

- Collaborative Environment
  - All ideas valued and considered

02
SmartClass Design

Initial Mockups

- Lectures
  - Upload lecture notes and follow along
  - Take in-class quizzes
  - Live-Chat feature with anonymous mode
  - Chat saved for future reference

Initial Mockups

- Quizzes
  - Test knowledge on the fly
  - Create quizzes for use during lecture
  - Formats
    - Multiple choice
    - Fill in the blank
    - True/False
    - Short Answer

SmartClass

Hello, Kelly ▾   Sign Out

Live ● Classes

Home   Classes   Chat   Schedule   People   Support   Search   About

COMS:352 | 02/14/25 9:30AM - 10:20AM:

Live View:

1. For each of the following memory allocation schemes explain why it does or does not experience external fragmentation or internal fragmentation.

- a. Base and bounds
- b. Segmentation
- c. Paging

2. Most systems allow a program to allocate more memory to its address space during execution. Allocation of data in the heap segments of programs is an example of such allocated memory. What is required to support dynamic memory allocation in the following schemes?

- a. Base and bounds
- b. Segmentation
- c. Paging

3. For each of the following virtual addresses, compute the page number and offset when using 4KB pages.

- a. 22,000
- b. 77,056
- c. 197,012

page 0

page 1

frame number

0

1 page 0

Initial Mockups

- Personal Profile
  - Join Classes
  - Review previous lectures/quizzes
  - See upcoming classes
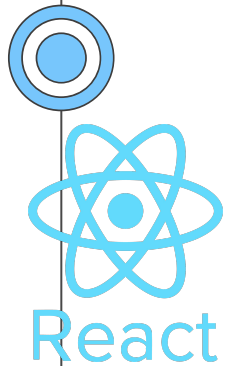
Frontend

- React JS
  - Component based
  - Reduced redundancy
  - Dynamic and Interactive interfaces
- Page hierarchy
  - Login to enter class
  - Navigation Bar is always present

Navigation Bar

Home Page

FAQ About Us

Create Account

Login

Contact

Dashboard

Class Page

Announcements

Materials

Live Chat

Discussion board
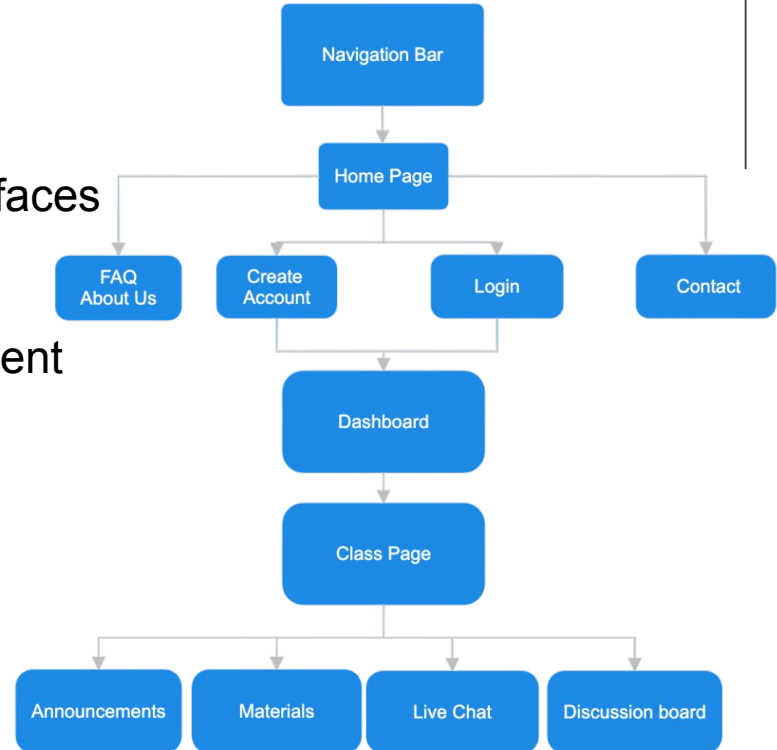
Backend

- Java & Spring Boot
  - Our backend code is written in Java.
  - Spring Boot for its web development features, security, authentication, and automated tests.
  - We utilize Maven for managing dependencies.

- MySQL
  - We use a MySQL server hosted on our own dedicated server.
  - MySQL allows us to appropriately design complex relationships and perform difficult queries.

CI/CD

- GitLab CI/CD
  - Autonomous deployment
  - Utilizes YAML file to run scripts
  - The deployment script follows these actions

```
Push to Main → Pipeline Starts → SSH to ISU VM
Pulls latest changes → Ends previous instances → Reboots Frontend/Backend
```
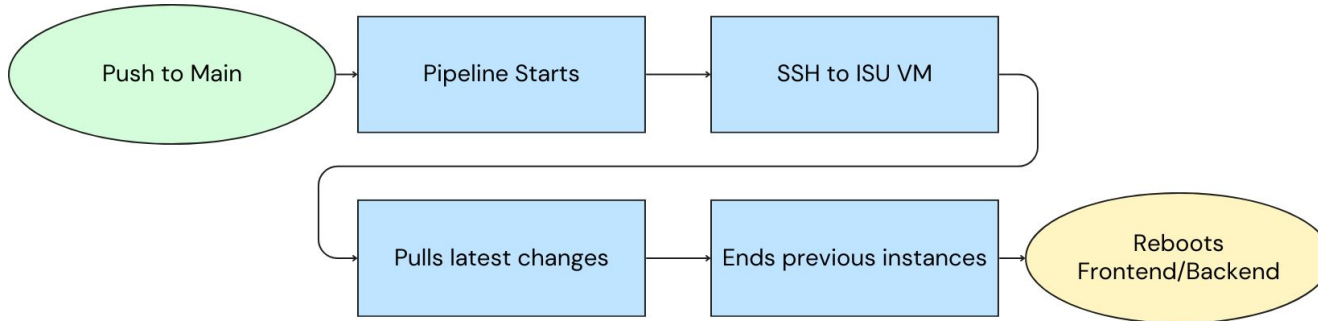
Design

Course Creation and Management

Live Polling and Questions

Anonymous Question Submission

Class Material Distribution

Student Participation Tracking

In-Class Discussion Forum

Current Features

- Create Account (Teacher and Student)
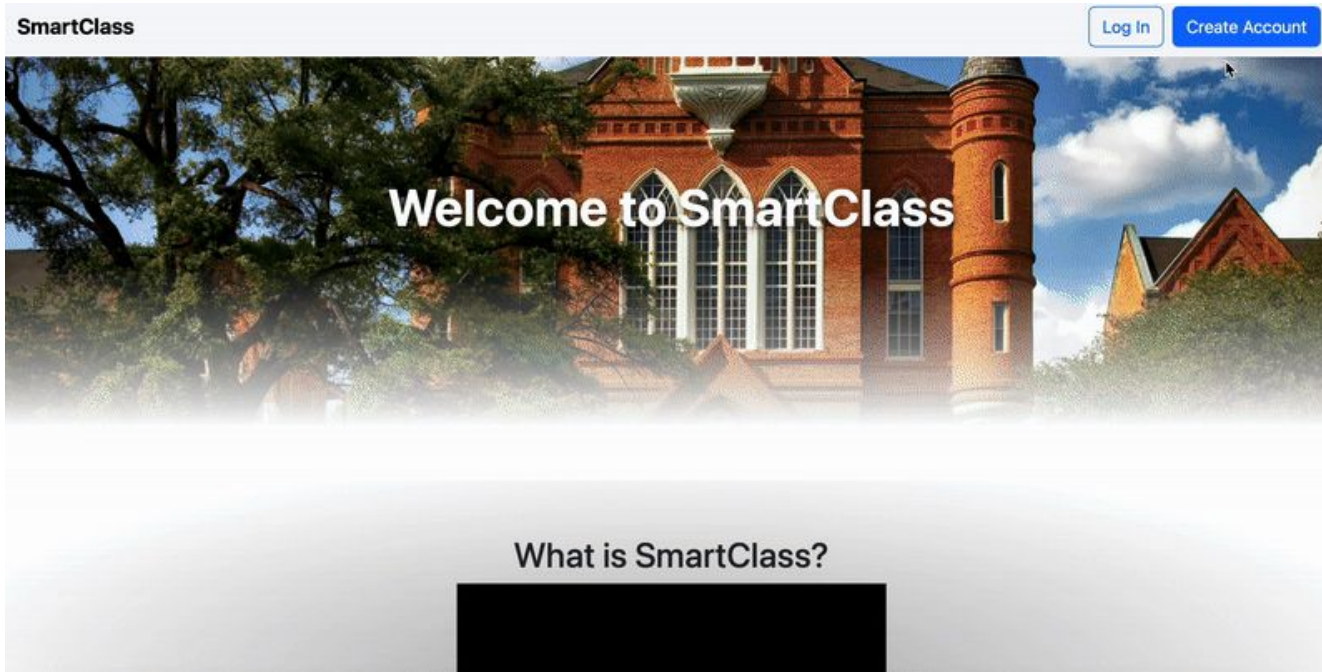
Current Features

- Sign in
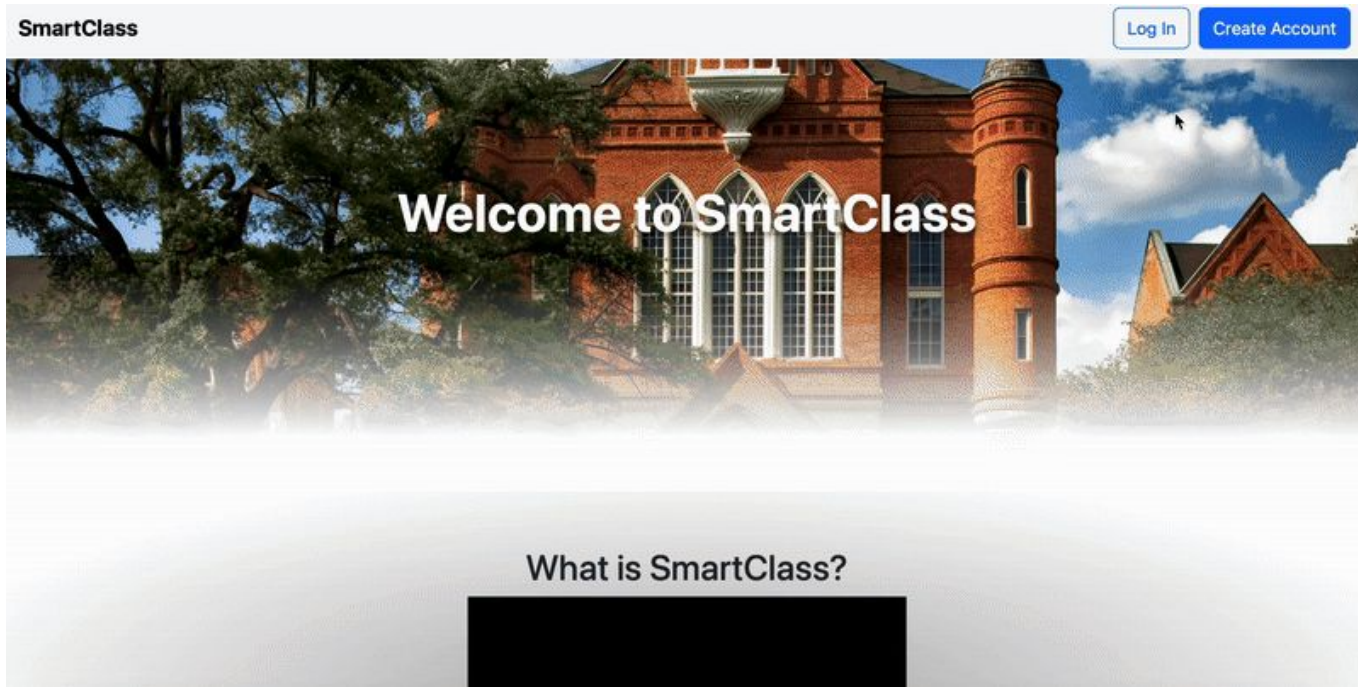
Current Features

- Create Class

Current Features

- Join Class

Current Features

- Live chat (including anonymous mode)

Current Features

- Discussion Forum
  - GIF to be added

Future Features

- Add Quizzes to live-chat
- Check lecture quiz grades/materials
- Live Lecture Recording
- Uploading lecture notes
- Whiteboard

Non-Functional Requirements

1. Performance
2. Reliability
3. Scalability in terms of number of users
4. Compatibility

Testing/Security

Unit Testing

Frontend

- React Testing Library
- Jest

Backend

- Database Queries
- Spring Boot
- jUnit

```
import {render, screen} from '@testing-library/react'
import userEvent from '@testing-library/user-event'
import '@testing-library/jest-dom'
import Fetch from './fetch'

test('loads and displays greeting', async () => {
  // ARRANGE
  render(<Fetch url="/greeting" />)

  // ACT
  await userEvent.click(screen.getByText('Load Greeting'))
  await screen.findByRole('heading')

  // ASSERT
  expect(screen.getByRole('heading')).toHaveTextContent('hello there')
  expect(screen.getByRole('button')).toBeDisabled()
})
```
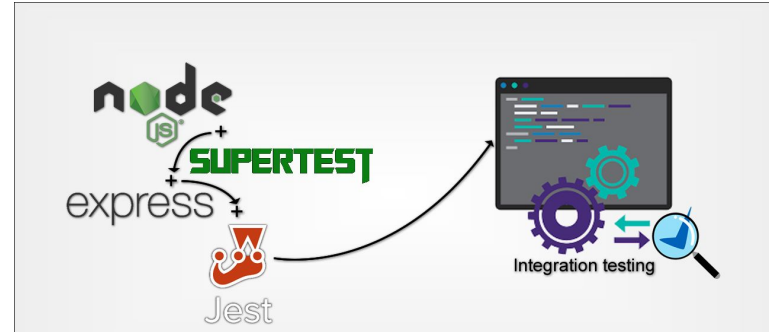
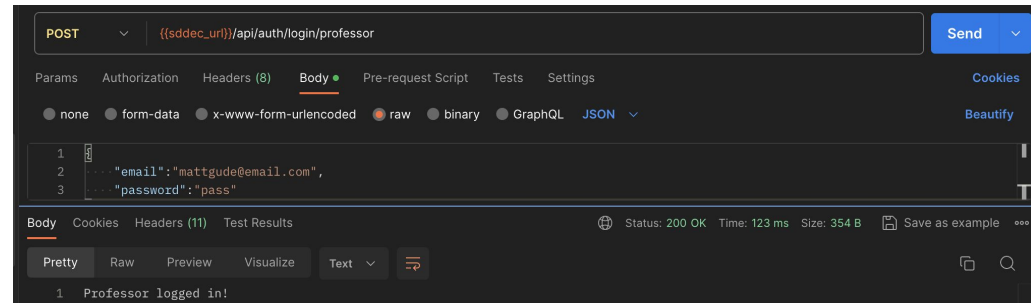# Integration Testing

Frontend:

- Supertest

Backend:

- Postman to Test APIs
- CI/CD - automated tests on gitlab pipeline

Performance Testing

- Load Testing - Simulate expected user load
- Stress Testing - Simulate beyond expected user load to see limits
- UI Load Testing - See if website is fast & responsive
- Browser-Based Load Testing - Ensuring website works on different browsers
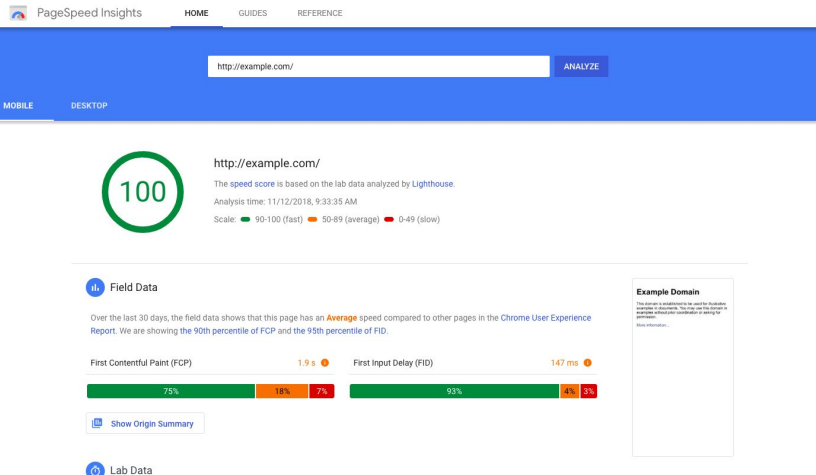
Usability Testing

- Focus on two main user groups:
  - Students
  - Teachers
- How will it be tracked?
  - Post-Test Survey: Quick feedback on usability (Google forms)
- Get feedback/suggestions

Student Feedback

We're gathering feedback on the usability of the SmartClass learning platform. Your input will help us improve the student experience.

Name (Optional)

Short answer text

What device did you use for testing?

○ Laptop

○ Tablet

Security Measures

Authentication & Access Control

- Passwords hashed with BCrypt
- JWT used for login and session validation
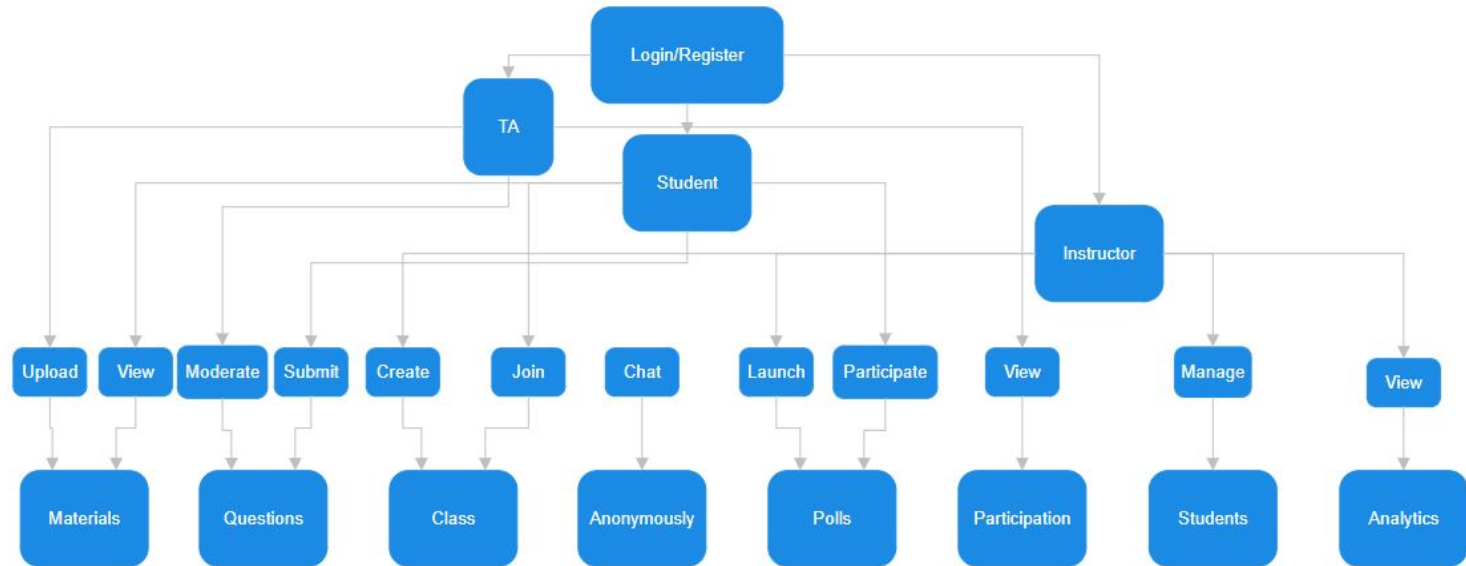- Role-based access controls (Student, TA, Instructor

Data Protection

- CORS configured to block unauthorized origins
- Input validated via DTOs
- Only required data exposed in responses

Backend SafeGuards

- Secured with Spring Security filters
- API endpoints require valid token
- Permission checks on class/forum access

Use-Case Diagram

Conclusion

SmartClass
Interactive Learning Tool

Design
React Application using
Spring Boot/MySQL

Future
Implement Live-Lecture
functionalities

GIFs (copy/paste if needed)