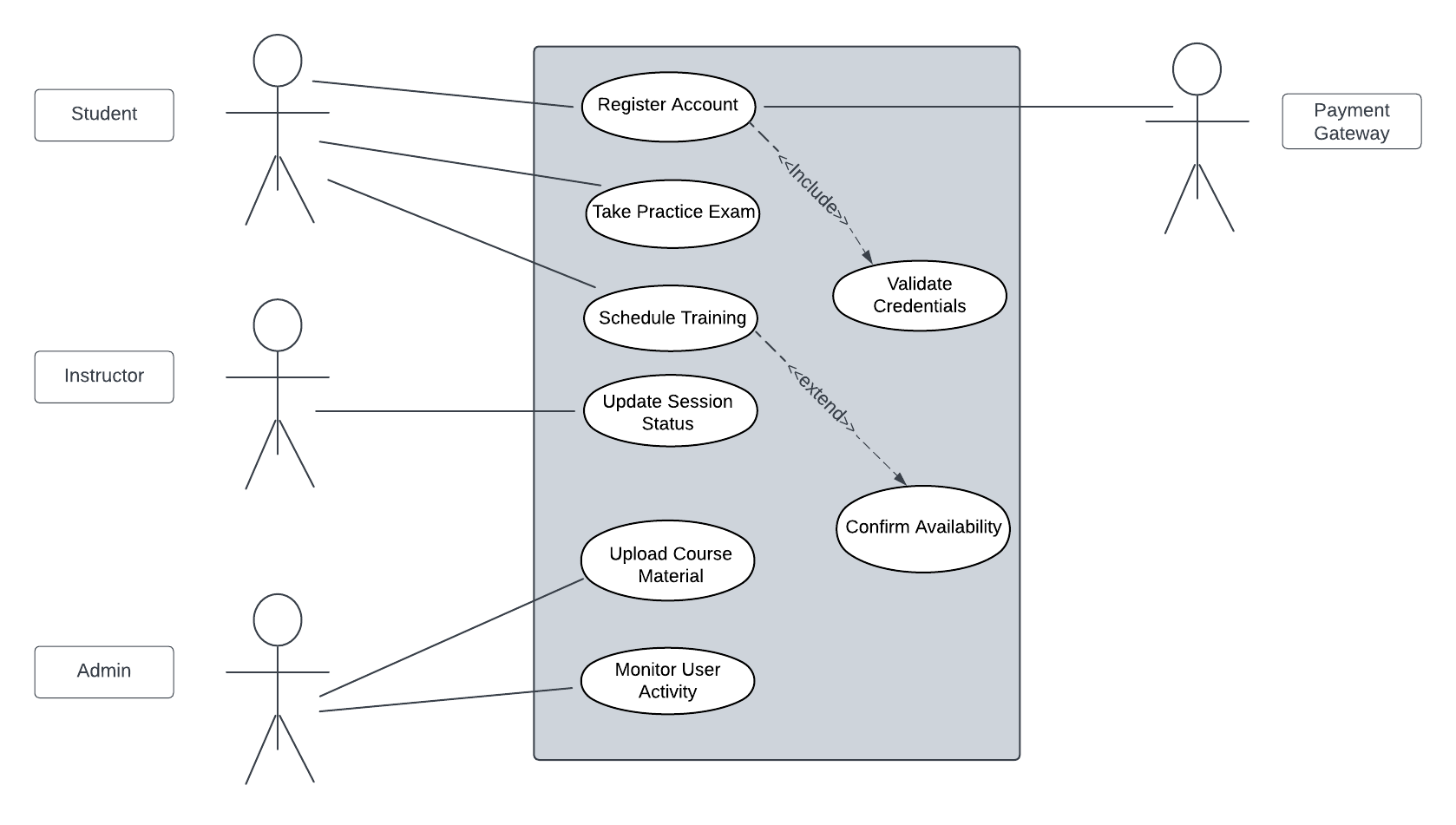
# CS 255 System Design Document Template

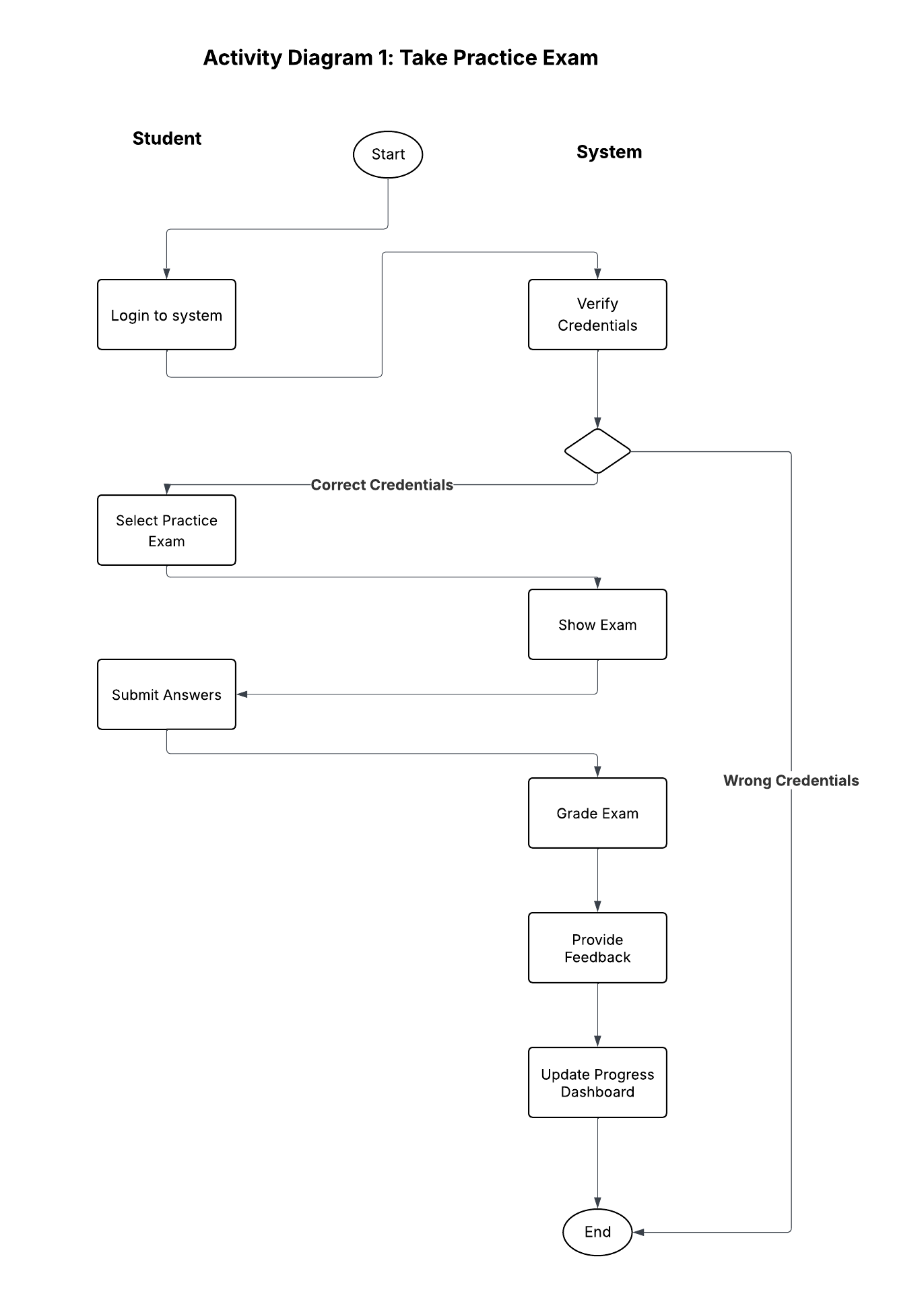
This template lays out all the different sections that you need to complete for Project Two. Each section has guidance to prompt your thinking. You will need to continually reference the interview transcript as you work to make sure that you are addressing your client’s needs. There is no required length for the final document. Instead the goal is to complete each section based on what your client’s needs are. Remove this note when you are finished, and replace all bracketed text with the relevant information.

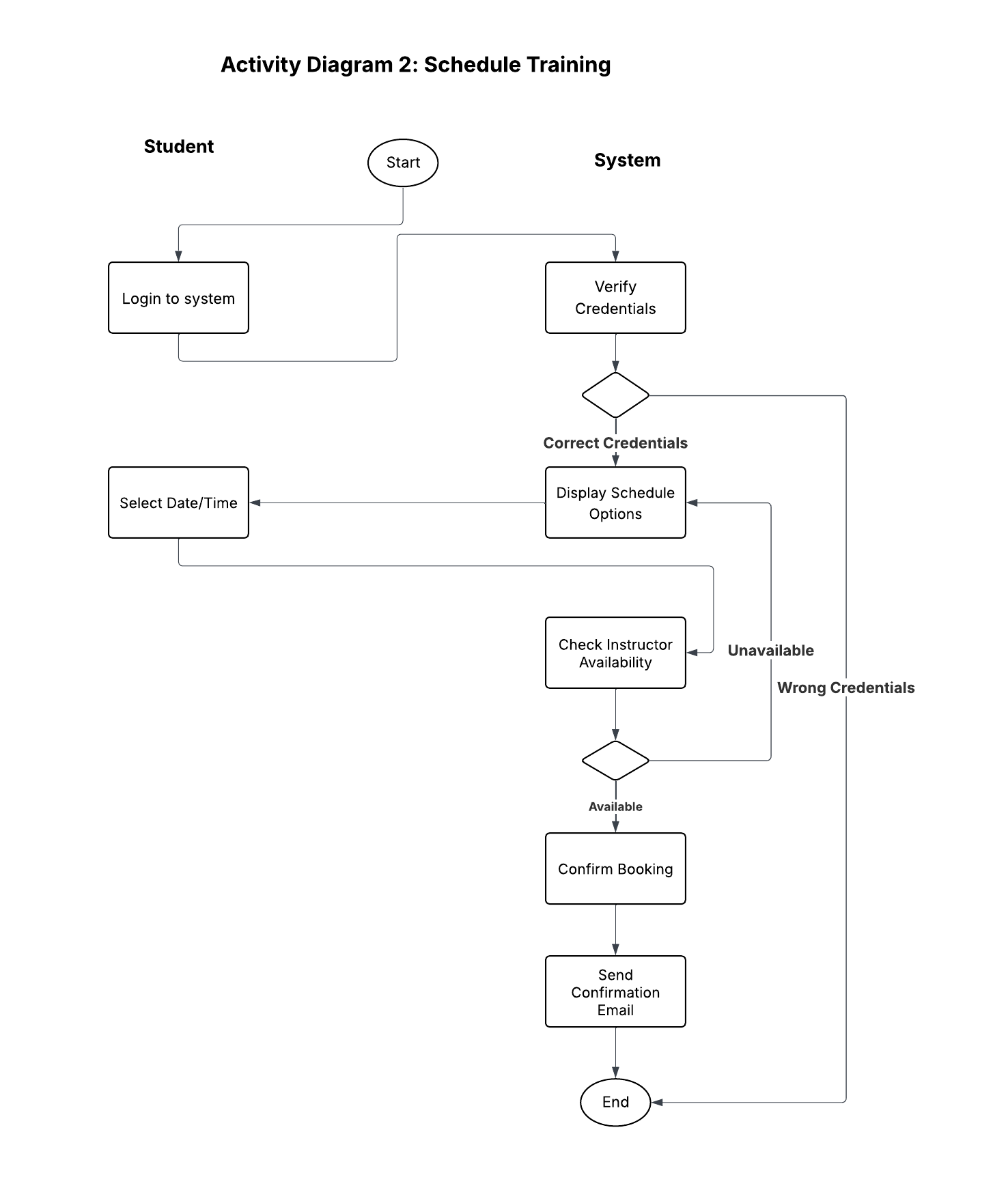
## UML Diagrams

### UML Use Case Diagram

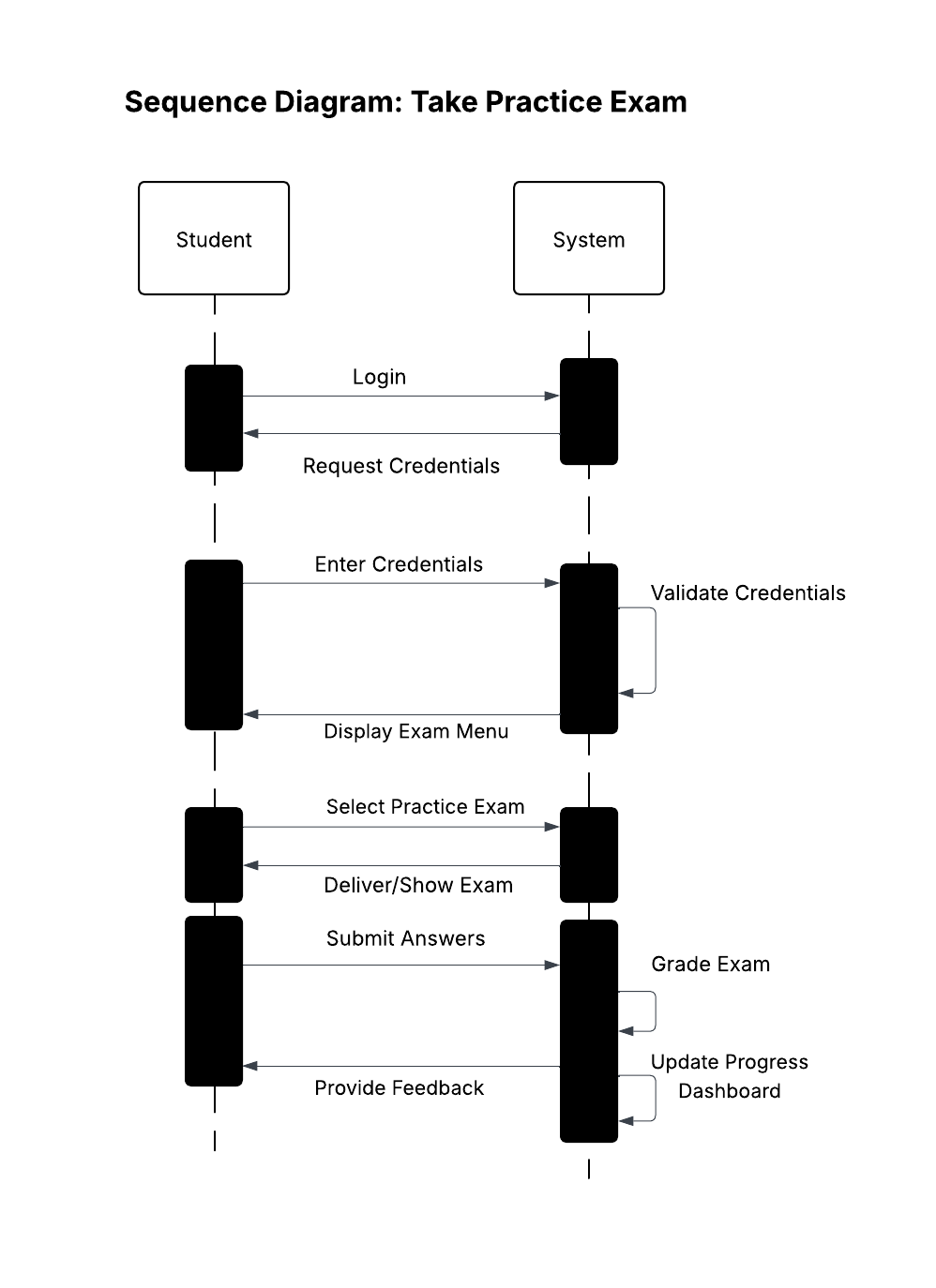


### UML Activity Diagrams

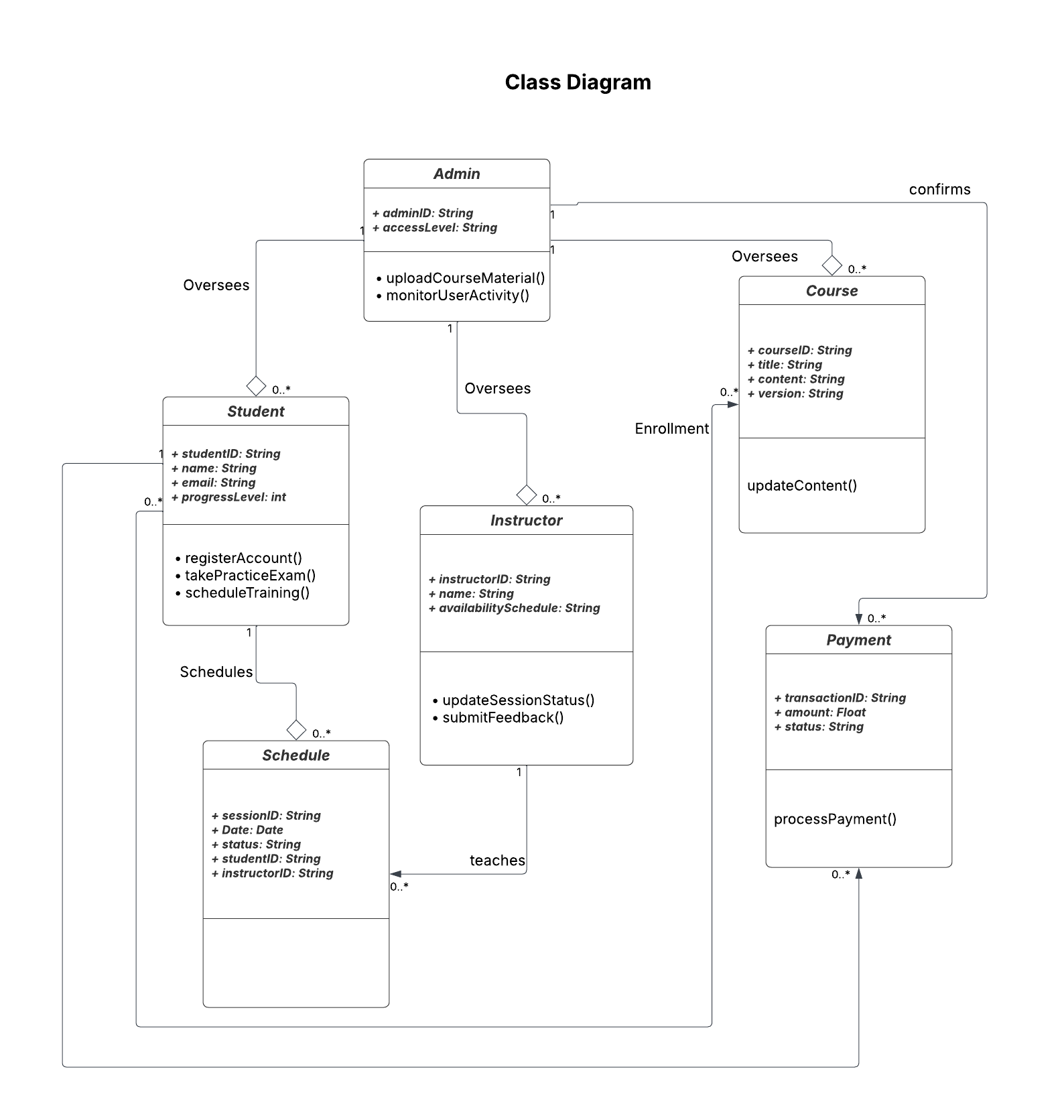




### UML Sequence Diagram



### UML Class Diagram



## Technical Requirements

The DriverPass system requires a robust, scalable, and secure technical infrastructure to meet the business objectives outlined in the CS 255 Business Requirements Document. These technical requirements address the hardware, software, tools, and infrastructure necessary to support a web-based platform that integrates online practice exams, on-the-road training scheduling, content management, and financial transactions, while ensuring accessibility, security, and future adaptability.

## **Hardware Requirements**

* **Servers**:
  + Minimum specification: Multi-core processors (e.g., 8-core CPU at 3.0 GHz or higher), 16GB RAM, and 500GB SSD storage to handle concurrent user requests and database operations efficiently.
  + Scalability: Provision for vertical scaling (e.g., additional CPU/RAM) and horizontal scaling (e.g., load-balanced server clusters) to accommodate growth as user base expands beyond initial projections.
* **Client Devices**:
  + Support for desktops, laptops, tablets, and smartphones with modern web browsers (e.g., Chrome, Firefox, Safari) and screen sizes ranging from 320px to 2560px width, ensuring responsive design compatibility.
  + Minimum device specs: 2GB RAM, 1.5GHz processor, and stable internet connectivity (e.g., 5 Mbps download speed).
* **Networking Hardware**:
  + High-speed routers and switches with QoS (Quality of Service) settings to prioritize real-time data (e.g., exam feedback, scheduling updates) over less critical traffic.
  + Redundant internet connections (e.g., dual ISPs) to ensure 99.9% uptime for the web application.

## **Software Requirements**

* **Frontend**:
  + Built using React.js with Tailwind CSS for a responsive, user-friendly interface across all device types, supporting the student dashboard, instructor portal, and administrative dashboard as specified.
  + JavaScript libraries for dynamic features like progress tracking and real-time updates.
* **Backend**:
  + Node.js with Express framework to handle API requests, ensuring scalability and efficient server-side logic for authentication, scheduling, and content management.
  + Integration with a PCI-compliant payment gateway (e.g., Stripe or PayPal) for secure transaction processing, adhering to the assumption of an existing payment processor.
* **Database**:
  + Relational Database Management System (RDBMS) such as MySQL or PostgreSQL to store user profiles (Student, Instructor, Admin), course materials, schedules, and payment data, supporting complex queries for reporting and monitoring.
  + Encrypted storage for sensitive data (e.g., email, payment tokens) using AES-256 encryption.
* **Security Software**:
  + SSL/TLS certificates for HTTPS encryption across all communications.
  + Two-factor authentication (2FA) via SMS or email for user logins, enhancing security as per the system’s secure authentication layer requirement.
  + Web Application Firewall (WAF) to protect against common vulnerabilities (e.g., SQL injection, XSS).

## **Tools Requirements**

* **Design and Development Tools**:
  + Lucidchart for creating and maintaining UML diagrams (use case, activity, sequence, class) during the design and documentation phase.
  + Git and GitHub for version control, enabling collaborative development and tracking changes to code and content (e.g., version control for course materials).
* **Testing Tools**:
  + Selenium for automated browser testing to ensure cross-device compatibility and responsiveness.
  + Postman for API testing to validate backend endpoints (e.g., scheduling API, payment API).
* **Deployment Tools**:
  + Docker for containerization to ensure consistent deployment across development, testing, and production environments.
  + Jenkins or GitHub Actions for continuous integration and deployment (CI/CD) to automate updates and reduce downtime.

## **Infrastructure Requirements**

* **Hosting**:
  + Cloud-based platform (e.g., Amazon Web Services AWS) with elastic compute instances (e.g., EC2) and auto-scaling groups to handle varying loads, supporting the system’s scalability goal.
  + Backup and disaster recovery: Daily incremental backups stored in a separate region (e.g., AWS S3 with versioning) and a full backup weekly, ensuring data integrity as emphasized by Ian in the interview.
* **Network**:
  + Content Delivery Network (CDN) (e.g., Cloudflare or AWS CloudFront) to cache static content (e.g., course materials, images) and reduce latency for users across different geographic locations.
  + Load balancers to distribute traffic evenly across servers, preventing bottlenecks during peak usage (e.g., exam periods).
* **Real-Time Features**:
  + WebSocket or Server-Sent Events (SSE) for real-time synchronization of training schedules and progress updates, requiring low-latency network infrastructure (e.g., <100ms round-trip time).
  + Message queue system (e.g., RabbitMQ) for asynchronous tasks like sending confirmation emails, ensuring non-blocking performance.
* **Security Infrastructure**:
  + Virtual Private Cloud (VPC) with segmented subnets for web servers, databases, and administrative access, isolating sensitive data.
  + Intrusion Detection System (IDS) and logging to monitor security events, fulfilling the IT monitoring requirement for system performance metrics and security logs.

## **Additional Considerations**

* **Performance**: The system must process user actions (e.g., exam submissions, schedule bookings) within 5 seconds, as inferred from the need for an efficient user experience, with server response times under 2 seconds under normal load.
* **Accessibility**: Compliance with WCAG 2.1 Level AA for screen reader support and keyboard navigation, ensuring accessibility for all users as part of the responsive design goal.
* **Future Adaptability**: Modular architecture to allow integration of advanced features (e.g., AI-driven analytics, multilingual support) once budget and resource constraints are alleviated, aligning with the client’s vision for evolution.

These technical requirements ensure the DriverPass system is secure, scalable, and user-friendly, meeting the client’s needs for improved driver training success rates and operational efficiency.