# 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

A diagram of a driver pass system

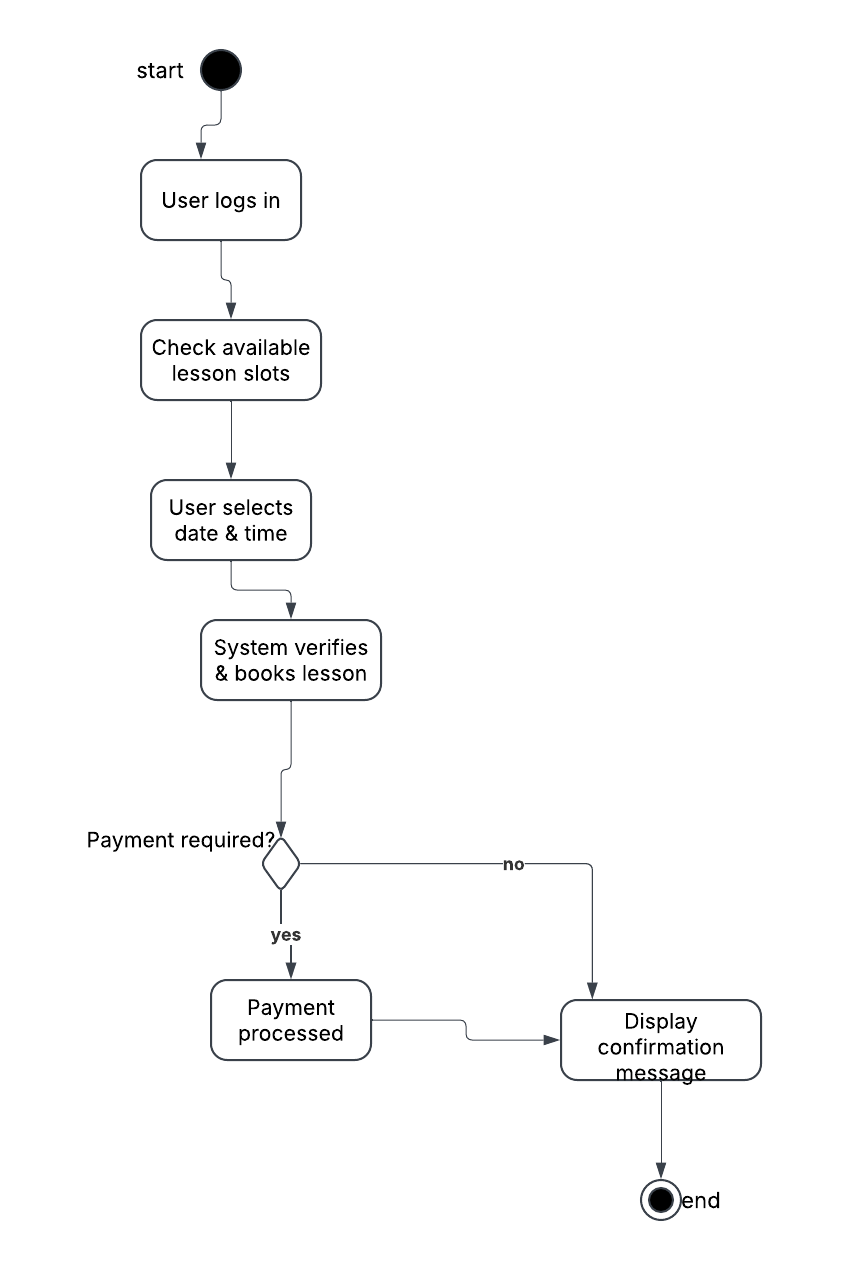
AI-generated content may be incorrect.

### UML Activity Diagrams

A diagram of a test

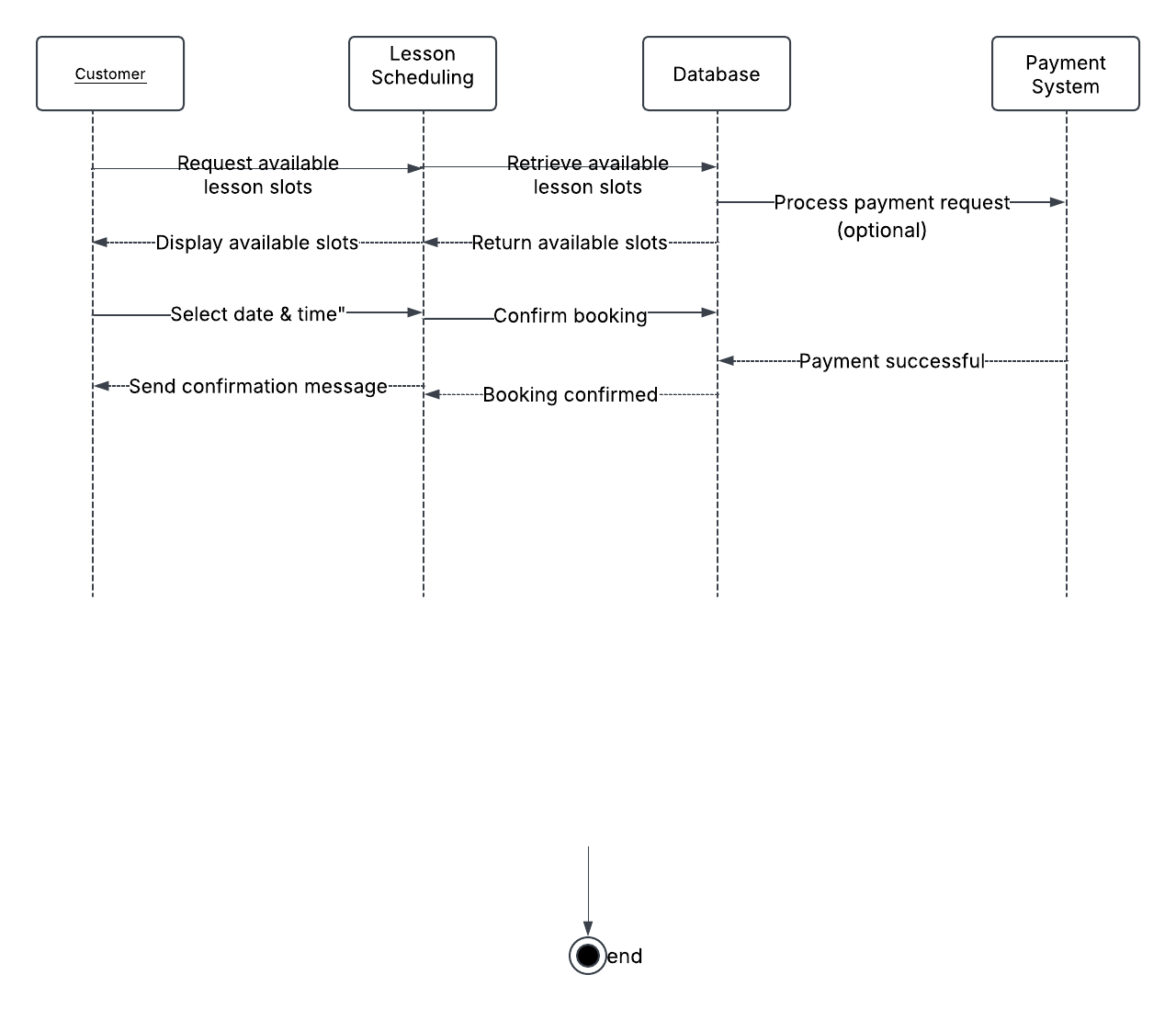
AI-generated content may be incorrect.

1: Take a Test

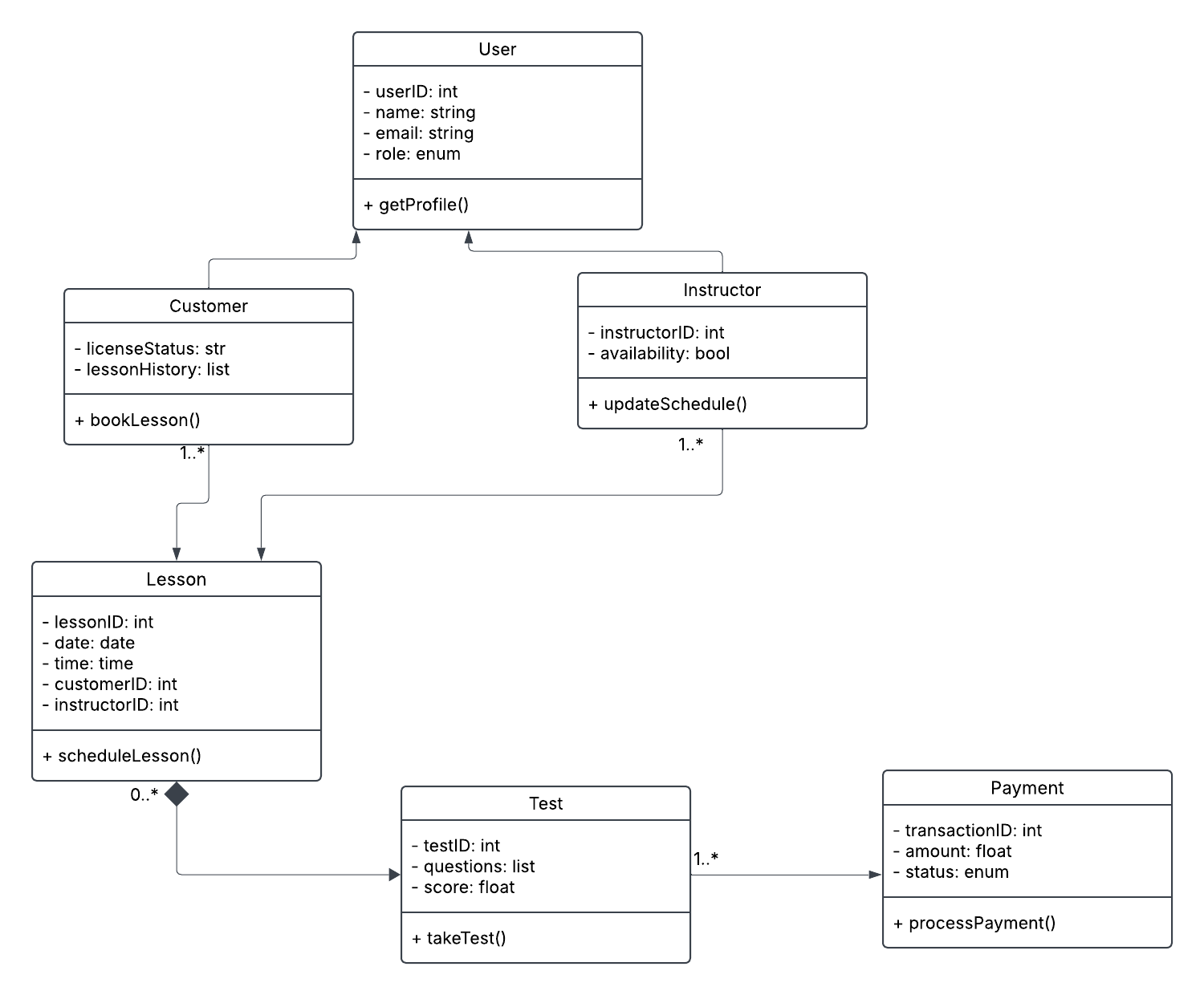


2: Schedule a lesson

### UML Sequence Diagram



### UML Class Diagram



## Technical Requirements

To ensure seamless operation, the system requires a combination of robust hardware, scalable software, and a secure infrastructure.

**Hardware Requirements**

The system will require both server-side and client-side hardware to function efficiently. On the server side, a dedicated web server with an Intel Xeon E5 or AMD EPYC processor, at least 32GB of RAM, and a 1TB SSD storage capacity will be necessary to handle user requests, process transactions, and store lesson records. Additionally, a separate database server with a minimum of 64GB RAM and a 2TB SSD storage (configured with RAID-10 for redundancy) is recommended to ensure reliable data storage and fast query execution.

On the client side, users will need a standard desktop or laptop with a minimum of an Intel i3 processor, 4GB of RAM, and a stable internet connection. Since the system is designed to be mobile-friendly, users can also access it through smartphones or tablets running Android 8.0+ or iOS 12+, ensuring accessibility across different devices.

**Software Requirements**

The backend of the DriverPass system will be built using a Linux-based operating system, such as Ubuntu 22.04 LTS, running on an Apache or NGINX web server. The application will be developed using a combination of Django (Python) or Node.js (Express.js) for backend functionality and PostgreSQL or MySQL as the database management system. Given the system’s requirement for frequent updates, the database should support replication to prevent downtime.

On the client side, users will interact with the system through a modern web browser, including Google Chrome, Mozilla Firefox, Microsoft Edge, or Apple Safari. The frontend will be developed using HTML5, CSS3, and JavaScript, with frameworks such as React.js, Vue.js, or Angular to ensure a responsive and interactive user experience. Bootstrap will be incorporated to enhance the system’s mobile compatibility, allowing students to schedule lessons and take tests from any device.

**Development Tools and Infrastructure**

To support development, version control will be managed using GitHub, GitLab, or Bitbucket, allowing developers to track changes and collaborate efficiently. The backend will be developed in Visual Studio Code or PyCharm, while database management can be handled using tools such as DBeaver or MySQL Workbench.

Several third-party integrations will enhance system functionality. A payment gateway such as Stripe or PayPal will facilitate secure online transactions for lesson bookings. OAuth 2.0 authentication will allow users to log in via Google or Facebook, improving security and convenience. Additionally, SMTP services like SendGrid or Mailgun will handle email notifications for booking confirmations and reminders, while the Google Maps API will assist with selecting lesson pickup locations.

**Network and Security Infrastructure**

The system will be deployed on AWS EC2, Microsoft Azure, or Google Cloud, utilizing load balancing with NGINX or AWS Elastic Load Balancer to handle traffic surges. A content delivery network (CDN), such as Cloudflare or AWS CloudFront, will improve website performance by caching static assets closer to users.

Security is a top priority, with SSL/TLS encryption implemented to protect all communications between the client and the server. User passwords will be securely hashed using Argon2 or Bcrypt, and two-factor authentication (2FA) will be required for account security. The system will comply with GDPR and CCPA regulations, ensuring that user data is handled responsibly.

To safeguard against data loss, daily database backups will be stored in AWS S3 or Google Cloud Storage, with automated recovery options in case of system failure. A firewall, such as UFW (Linux Firewall) or Cloudflare Web Application Firewall (WAF), will be deployed to protect against cyber threats.