# DriverPass System Design Document

## UML Diagrams

### UML Use Case Diagram

A screenshot of a diagram

AI-generated content may be incorrect.

Figure . DriverPass Use Case Diagram

### UML Activity Diagrams

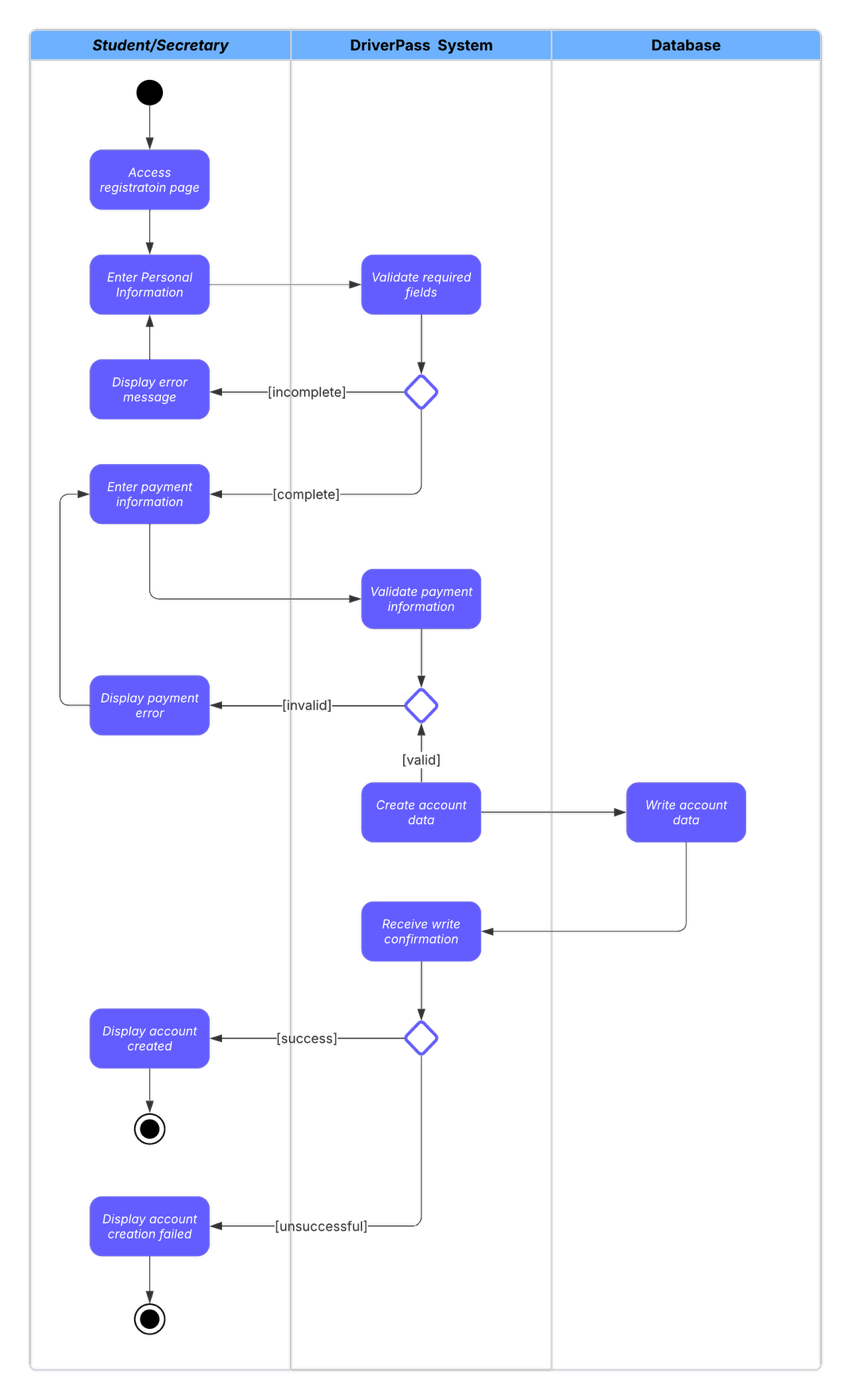


Figure . DriverPass Register Account Activity Diagram

### A screenshot of a computer screen AI-generated content may be incorrect.

Figure . DriverPass Make Reservation Activity Diagram

### UML Sequence Diagram

*A screenshot of a computer screen

AI-generated content may be incorrect.*

Figure . DriverPass Make Reservation Sequence Diagrm

### UML Class Diagram

## A screenshot of a computer screen AI-generated content may be incorrect.

Figure . DriverPass Class Diagram

## Technical Requirements

## Cloud Infrastructure Requirements

### Cloud Platform Services

* **Cloud Provider**: Amazon Web Services (AWS), Microsoft Azure, or Google Cloud Platform providing fully managed infrastructure to eliminate DriverPass's need for hardware management, backup administration, and security maintenance
* **Application Hosting**: Cloud-native application services (AWS Elastic Beanstalk, Azure App Service, or Google App Engine) with auto-scaling capabilities supporting concurrent user sessions
* **Managed Database**: Cloud database services (AWS RDS, Azure SQL Database, or Google Cloud SQL) with automated backups, security patches, and high availability configurations
* **Load Balancing**: Cloud-managed load balancers (AWS Application Load Balancer, Azure Load Balancer) automatically distributing traffic without hardware management
* **Content Delivery**: Managed CDN services (AWS CloudFront, Azure CDN) for global distribution of course materials and practice test content

### Managed Security Services

* **Automated Backup**: Cloud-native backup services with configurable retention policies, eliminating manual backup management for DriverPass staff
* **Security Management**: Cloud provider managed security including firewalls, DDoS protection, and automated security patches as requested by Ian
* **SSL/TLS Certificates**: Managed certificate services providing automatic renewal and implementation of HTTPS encryption
* **Identity Management**: Cloud-based authentication services (AWS Cognito, Azure Active Directory) handling user authentication and session management

### Client Device Requirements

* **Desktop/Laptop**: Modern web browsers (Chrome 90+, Firefox 88+, Safari 14+, Edge 90+) with JavaScript enabled for full cloud application access
* **Mobile Devices**: iOS 12+ or Android 8+ with responsive web browser support for mobile-optimized interfaces
* **Internet Connectivity**: Stable broadband connection (minimum 5 Mbps) for optimal cloud application performance
* **Device Performance**: 4GB RAM recommended for smooth operation of cloud-based courses and practice tests

## Software Requirements

### Cloud Application Platform

* **Managed Application Platform**: Cloud-native application services (AWS Elastic Beanstalk, Azure App Service, Google App Engine) where the cloud provider manages all underlying operating system requirements, eliminating DriverPass's need for OS administration
* **Client Platforms**: Platform-independent browser-based access supporting Windows 10+, macOS 10.15+, iOS 12+, Android 8+ - no local software installation required

### Cloud Database Services

* **Managed Database**: Cloud-hosted PostgreSQL (AWS RDS PostgreSQL, Azure Database for PostgreSQL) or MySQL (AWS RDS MySQL, Azure Database for MySQL) with automatic backup, patching, and high availability
* **Database Features**: Full ACID transaction support, foreign key constraints, automated backup with point-in-time recovery, and automatic security updates managed by cloud provider
* **Disaster Recovery**: Multi-region database replication and automated failover capabilities provided by cloud infrastructure without DriverPass management overhead

### Web Application Framework

* **Backend Framework**: Node.js with Express.js or Java Spring Boot for robust API development and business logic implementation
* **Frontend Framework**: React.js or Angular for responsive user interface supporting role-based access control
* **Authentication System**: JWT (JSON Web Tokens) or OAuth 2.0 for secure user session management
* **Real-Time Conflict Prevention**: WebSocket or Server-Sent Events (SSE) implementation for real-time appointment scheduling conflict detection and prevention across driver, vehicle, and time slot resources
* **Input Validation Framework**: Comprehensive client-side and server-side validation library (joi, express-validator, or similar) with configurable validation rules for all user inputs including registration data, appointment details, and payment information
* **Business Rule Engine**: Event-driven notification system for automated alerts when students fail multiple practice tests, appointment conflicts arise, or critical system errors occur with configurable business logic rules

### Third-Party Integration Software

* **Payment Processing**: Integration with Stripe, PayPal, or Square APIs for secure credit card transaction processing
* **DMV Integration**: RESTful API clients for connecting with state DMV systems to receive updates on rules, policies, and sample questions
* **Email Services**: SendGrid or Amazon SES for automated notifications and password reset functionality
* **File Storage**: Amazon S3 or Azure Blob Storage for storing student photos, course materials, and exported reports
* **Content Delivery System**: Learning Management System (LMS) components for online course delivery including video streaming capabilities, progress tracking, and interactive content presentation with current DMV materials
* **Administrative Management Platform**: Dedicated admin interface framework supporting user account management, package configuration controls, system monitoring dashboards, and comprehensive reporting tools for the IT Officer and Owner roles

## Tools and Infrastructure

### Development Tools

* **Version Control**: Git with GitHub or GitLab for source code management and collaborative development
* **IDE/Editors**: Visual Studio Code, IntelliJ IDEA, or similar modern development environments
* **API Testing**: Postman or Insomnia for testing RESTful API endpoints during development
* **Database Tools**: pgAdmin (PostgreSQL) or MySQL Workbench for database administration and query optimization

### Cloud Infrastructure

* **Cloud Platform**: Amazon Web Services (AWS), Microsoft Azure, or Google Cloud Platform for scalable hosting
* **Container Technology**: Docker for application containerization and consistent deployment environments
* **Orchestration**: Kubernetes or Docker Swarm for managing containerized applications and auto-scaling
* **CDN Services**: CloudFlare or AWS CloudFront for global content distribution and improved performance

### Monitoring and Maintenance Tools

* **Application Monitoring**: New Relic, DataDog, or Application Insights for real-time performance monitoring
* **Log Management**: ELK Stack (Elasticsearch, Logstash, Kibana) or Splunk for centralized logging and analysis
* **Uptime Monitoring**: Pingdom or UptimeRobot for continuous availability monitoring
* **Backup Solutions**: Automated backup services integrated with cloud storage providers
* **Error Notification System**: Automated alerting system for critical system errors, appointment scheduling conflicts, and business rule violations with immediate notification delivery to appropriate administrators via email and dashboard alerts
* **Business Intelligence Dashboard**: Real-time monitoring interface for system administrators showing user activity patterns, appointment booking trends, practice test performance analytics, and system health metrics

### Testing Environment

* **Automated Testing**: Jest (JavaScript) or JUnit (Java) for unit testing of business logic
* **Integration Testing**: Selenium WebDriver for automated browser testing of user workflows
* **Load Testing**: Apache JMeter or LoadRunner for performance testing under peak user loads
* **Staging Environment**: Production-mirror environment for pre-deployment testing and validation

## Security and Compliance

### Cloud-Managed Security and Compliance

#### Automated Security Management

* **Managed Security**: Cloud provider handles all infrastructure security including firewall management, intrusion detection, and security patch deployment as specifically requested by Ian
* **Automated Backup**: Cloud-native backup services (AWS Backup, Azure Backup) with configurable retention policies and automated disaster recovery, eliminating DriverPass's need for backup administration
* **Compliance Frameworks**: Cloud providers' built-in compliance certifications (SOC 2, ISO 27001, PCI DSS) supporting DriverPass's regulatory requirements without internal compliance management

#### Application-Level Security

* **Data Encryption**: Managed encryption services for data in transit (TLS 1.3) and at rest (AES-256) provided automatically by cloud infrastructure
* **Identity and Access Management**: Cloud-based authentication services with role-based access control for the five user types (Owner, IT Officer, Secretary, Driver, Student)
* **Automated Monitoring**: Cloud security monitoring services with automatic threat detection and response, reducing IT Officer workload
* **Account Security Controls**: Multi-factor authentication (MFA) for administrative users, automatic account lockout after configurable failed login attempts (default 5 attempts), and immediate notification to IT Officer when security thresholds are exceeded
* **Comprehensive Audit Logging**: System-wide activity tracking and logging framework capturing all user actions, data modifications, appointment changes, payment transactions, and administrative activities with tamper-proof timestamps and user identification for complete accountability trails
* **Case-Insensitive Authentication**: Login system supporting case-insensitive username/email input while maintaining password case sensitivity for enhanced user experience and reduced login failures

## Performance and Scalability

### Performance Requirements

* **Response Time**: Maximum 3 seconds for page loads, 1 second for API responses under normal load
* **Concurrent Users**: Support for minimum 500 simultaneous users with ability to scale to 2000+ users
* **Database Performance**: Query optimization with indexing strategies for fast data retrieval
* **Caching Strategy**: Redis or Memcached for session storage and frequently accessed data

### Scalability Architecture

* **Horizontal Scaling**: Auto-scaling groups that automatically add/remove server instances based on demand
* **Microservices Architecture**: Separate services for user management, scheduling, payment processing, and content delivery
* **Database Scaling**: Read replicas for improved query performance and load distribution
* **Content Optimization**: Image compression and minification of CSS/JavaScript files for faster loading

### Integration Capabilities

* **API Design**: RESTful APIs with comprehensive documentation for future third-party integrations
* **Webhook Support**: Real-time notifications for appointment changes, payment confirmations, and DMV updates
* **Export Functionality**: APIs for exporting data to Excel, CSV, and PDF formats
* **Mobile App Ready**: API-first design to support future native mobile application development

## Quality Assurance

### Testing Requirements

* **Browser Compatibility**: Testing across all major browsers and devices specified in client requirements
* **Accessibility Testing**: WCAG 2.1 AA compliance for users with disabilities
* **Security Testing**: Regular penetration testing and vulnerability assessments
* **Performance Testing**: Load testing to validate system performance under expected user volumes

### Documentation

* **Technical Documentation**: Comprehensive API documentation using tools like Swagger/OpenAPI
* **User Manuals**: Role-specific user guides for each type of system user
* **Deployment Guides**: Step-by-step deployment and configuration documentation
* **Maintenance Procedures**: Documentation for routine maintenance, updates, and troubleshooting