Southern New Hampshire University

CS 255 System Design: DriverPass

UML Diagrams

DriverPass Driving Instruction

Diagram Key
- Use case
- Relationship
- Actor

Self-Regitser

Register User

Add User displays Sef-register for Students Only

Process Payment

Bank

Add User
Extension Points
Self-Register
Register User

<<extends>>

<<extends>>

<<includes>>

Select Instruction Package

<<includes>>

Log In

<<includes>>

View Landing Page
Extension Points
Student View
Instructor View
Admin View
Super Admin View

Login displays the appropriate Landing Page based on User Role

Web Services

Cloud Services

Database Services

Analytics Services

<<extends>>

Super Admin View

Instructor View

<<extends>>

Student View

<<extends>>

Admin View

<<extends>>

User

Student

Instructor

Admin

Super Admin

Reserve Instruction

<<includes>>

Update Instruction Calendar

<<includes>>

View Instructional Content

<<includes>>

Update Student Record

<<extends>>

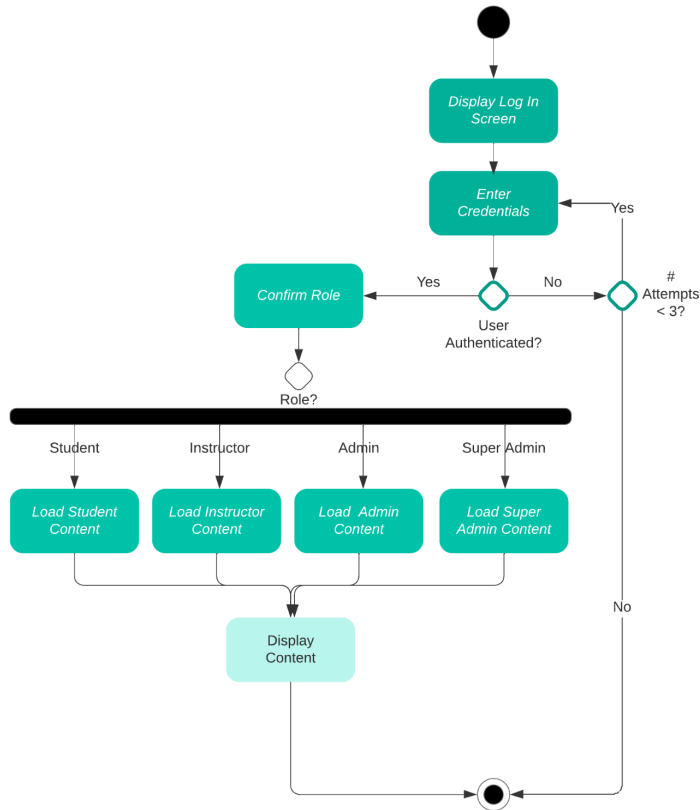Take Test

<<includes>>

Grade Test

<<includes>>

1

Southern New Hampshire University

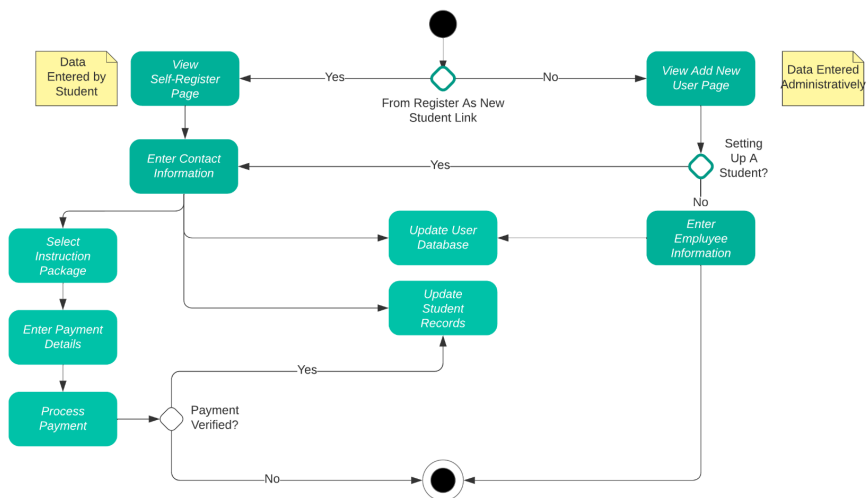## UML Activity Diagrams

Log In Activity diagram
Joshua Wozny | March 11, 2023

AddUser Activity diagram
Joshua Wozny | March 11, 2023

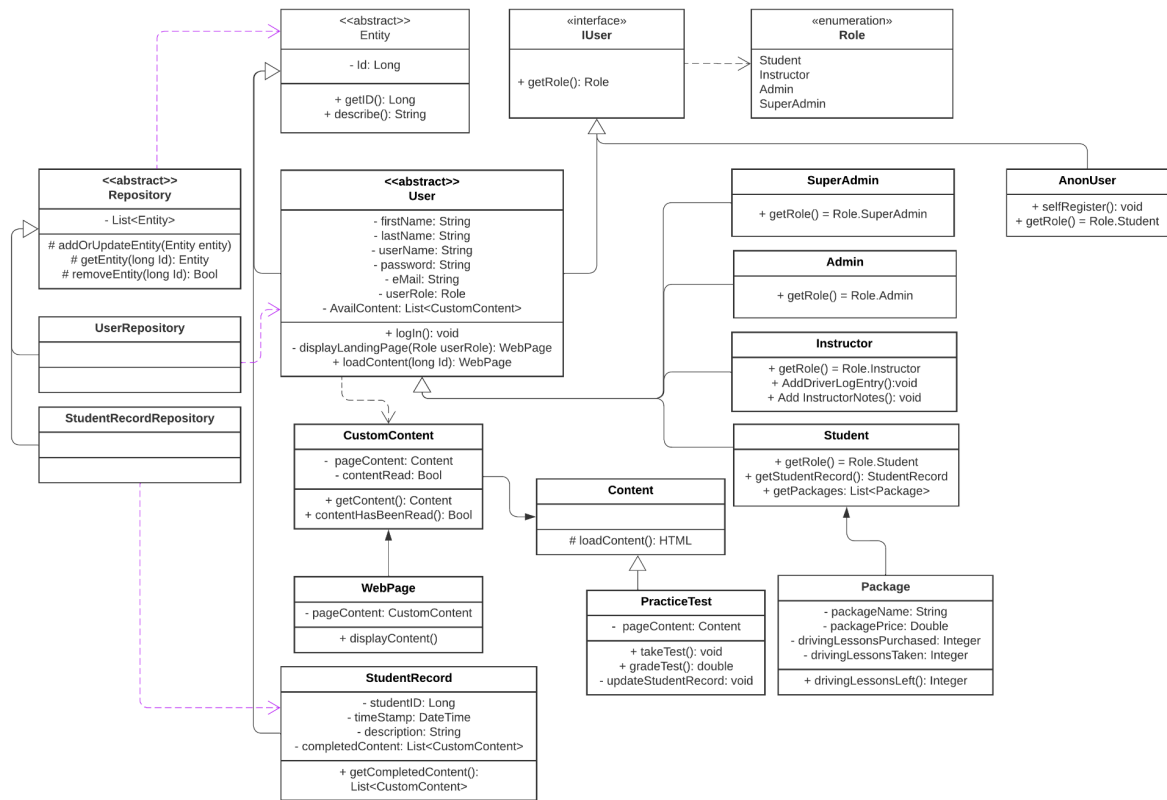**UML Sequence Diagram**

Log In Sequence Diagram
Joshua Wozny | March 11, 2023

User | Web Server | Database

Display Login Screen

Enter Cedentials

Vdrify Password

Confirm Log In

Log In Confirmation

Get Role

Provide User Role

Display Landing Page

Southern New Hampshire University
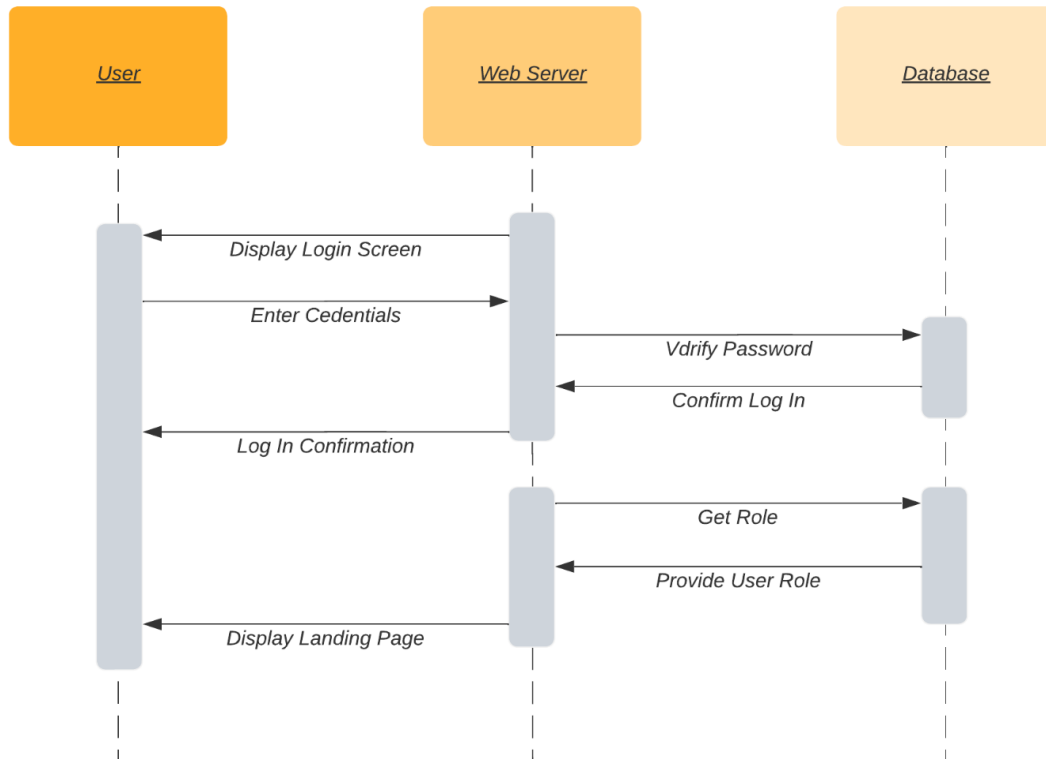
UML Class Diagram

DriverPass UML class

Joshua Wozny | March 12, 2023

<> Entity
- Id: Long

+ getID(): Long
+ describe(): String

«interface» IUser

+ getRole(): Role

«enumeration» Role

Student
Instructor
Admin
SuperAdmin

<> Repository
- List<Entity>

addOrUpdateEntity(Entity entity)
getEntity(long Id): Entity
removeEntity(long Id): Bool

UserRepository

StudentRecordRepository

<> User
- firstName: String
- lastName: String
- userName: String
- password: String
- eMail: String
- userRole: Role
- AvailContent: List<CustomContent>

+ logIn(): void
- displayLandingPage(Role userRole): WebPage
+ loadContent(long Id): WebPage

SuperAdmin

+ getRole() = Role.SuperAdmin

AnonUser

+ selfRegister(): void
+ getRole() = Role.Student

Admin

+ getRole() = Role.Admin

Instructor

+ getRole() = Role.Instructor
+ AddDriverLogEntry():void
+ Add InstructorNotes(): void

CustomContent
- pageContent: Content
- contentRead: Bool

+ getContent(): Content
+ contentHasBeenRead(): Bool

Student

+ getRole() = Role.Student
+ getStudentRecord(): StudentRecord
+ getPackages: List<Package>

Content

loadContent(): HTML

WebPage
- pageContent: CustomContent

+ displayContent()

PracticeTest
- pageContent: Content

+ takeTest(): void
+ gradeTest(): double
- updateStudentRecord: void

Package
- packageName: String
- packagePrice: Double
- drivingLessonsPurchased: Integer
- drivingLessonsTaken: Integer

+ drivingLessonsLeft(): Integer

StudentRecord
- studentID: Long
- timeStamp: DateTime
- description: String
- completedContent: List<CustomContent>

+ getCompletedContent():
List<CustomContent>

4

Technical Requirements

Summary: The system will be hosted on the Cloud as a web-based application. Payments for services will use an API provided by DriverPass' bank. The application back-end will handle User Authentication, and data storage will use a Cloud-based SQL server. The Cloud Service will provide analytics data.

Development Environment
- IDE: Web Storm & IntelleJ
- Languages: Java, SQL, Javascript, HTML
- Frameworks: React, Hibernate, Java Authentication and Authorization Services (JAAS)
- Implement Bank's Payment Processing API

System Components:
- Cloud Services
  - Web Server
  - Analytics Service
  - Database Server with PostgreSQL
  - Virtual Machine to run the back-end application
- Windows, Android, Linux, and Apple software compatibility
- Pages Designed for Search Engine Optimization

Assumptions: Users have an active internet connection and a web browser.
Limitations: The system will not function if the Cloud Provider experiences an outage. The Cloud Provider can suspend services if suspicious activity occurs or for a Terms of Service Breach.