# CS 255 Business Requirements Document

**Tip:** You should respond in a bulleted list for each section. This will make your thoughts easier to reference when you move into the design phase for Project Two. One starter bullet has been provided for you in each section, but you will need to add more.

## System Components and Design

### Purpose

* The purpose of this project is for DriverPass to better train students for their driving test at their local DMV. This system will allow students to book appointments online with driving teachers, purchase these appointments, take online practice tests, and keep track of student progress.

### System Background

* Ability to access DriverPass data from anywhere, both online and offline (read-only in offline)
* Adequate security and handling of data/changes
* Users can register for packages
* Packages may be disabled by admin if desired, possibly automatically if a certain criteria is met.
* Curriculum will be always up to date with DMV policies.
* Keeps track of tests taken by users with relevant data
* Comprehensive database containing all necessary information

### Objectives and Goals

* Implement a cloud-based database which will take care of backup and security and store all the necessary data.
* Have adequate security so that it prevents all users from making inappropriate/unauthorized changes to the system.
* “Reservation” component is required to appropriately create reservations made by users
* “Driver” and “User” components that will share data to track which user is matched up with a specific driver, time, and car. This may possibly fall under a “Reservation” or “Appointment” parent component.
* “Appointment” component: includes 10 “car” objects, each with a unique “driver” object
* “Packages” component: has 3 “package” objects. Each package builds upon the last in complexity. Each package also has a boolean “canRegister” value to enable/disable registering.
* “Registration” component: has “Customer” object which has *firstName*, *lastName*, *address, phoneNum, state, ccNum, expDate, securityCode, location* properties. (*location* will be the pickup and dropoff location for the driver/student)
* Connect to DMV database 24/7, watch for changes.
* “Test” component: has *name*, *time*, *score*, and *status* properties under the “User” parent component
* “Notes” component: has *lessonTime*, *startHour*, *endHour*, *comments* properties under the “Driver” user parent component.
* Multi-page interface:
  + Input form for getting student information
  + Contact page for communications between DriverPass and the student

## Requirements

### Nonfunctional Requirements

#### Performance Requirements

* The system needs to run off the web, preferably over the cloud. It can be accessed via Windows, macOS, Linux, Android, and iOS.
* The system needs to be connected to the DMV for expedient updates to regulations
* Must be able to support many users, capable of maintaining all user data which includes appointment, payment, and other account information.
* Data must be continuously updated to reflect all changes made by users within seconds. If an appointment is made, this change should happen almost instantly.

#### Platform Constraints

* The system should run on all major browsers, enabling compatibility for most operating systems.
* The back end will be a cloud-based database.

#### Accuracy and Precision

* User and admin accounts are distinguished by a unique username and a case-sensitive password.
* Admins will receive notifications of problems in the system. These notifications should be automatically labeled according to their severity/priority.
* The admin should be immediately notified of a problem

#### Adaptability

* There will be a singular implementation to add/remove/modify all and any data
* Developers will be able to adapt to any browser updates that may affect the system. Moreover, they are able to modify the source code, if absolutely necessary.
* The IT admin will need access to user data with the ability to add/remove/modify it.

#### Security

* All users will have a unique username and case-sensitive password that will require at least one capital letter and a special character.
* Client authentication through the browser will secure the connection between the client and server.
* If there is a brute force hacking attempt, the account should require a password reset to login sent to the user’s email on file.
* If a user forgets their password, they should answer challenge questions to be sent an email to reset their password.

### Functional Requirements

* The system shall be secure and handle the backup of data over the cloud
* The system shall allow each user to schedule appointments, take practice tests, register accounts, upload photos, and review driver notes.
* The system shall be accessible through a web-based browser on all major operating systems, including Android and iOS.
* The system shall validate user credentials when logging in

### User Interface

* The user interface shall look like the sketch provided by the owner
* Student users need to be able to fill in personal/payment information, schedule appointments, contact the company, and take tests
* IT users need to be able to modify data and have access to Student data. They should be given admin privileges.
* Administrator users need to be able to know who made a reservation, who canceled it, and who modified it last.
* The UI shall have multiple pages, including ones for entering student information, and for contacting the company
* Users will interact with the interface through a web based browser application

### Assumptions

* All users will need internet access, and no data will be able to be modified offline
* All users must run the application on a compatible web browser
* The DMV must be compliant with our program and be capable of sharing data effectively

### Limitations

* The system is limited to a web browser, and there are no plans for a mobile app as of yet
* Time and budget constraints.
* Will have any limitations of having the back end living on the cloud as opposed to dedicated servers.

### Gantt Chart

