# CS 255 Business Requirements Document Template

To decompose “Complete Project One”, the first step would be to review Project One Guidelines. From there we would Read the interview transcript, which would get broken down to reading followed by answering questions concerning the transcript. Next, we would Complete the business requirements document. We then would Schedule, using Lucidchart, to create Gantt chart outlining the schedule based on the questions answered from the transcript. Following this, we would complete a Model Application Short Paper, broken down further into a Model Application, describing how we would apply the process and object modeling approach, and then the Model Comparison, comparing the advantages and disadvantages to each of the models. We would then submit both the Business Requirements Document and Model Application Short Paper onto Brightspace. This would Complete Project One.

**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

*What is the purpose of this project? Who is the client and what do they want their system to be able to do?*

The project, DriverPass, is owned by our client Liam. Their purpose is to provide students with online and offline access to online practice exams, and on-the-road training to help prepare them for driving tests.

### System Background

*What does DriverPass want the system to do? What is the problem they want to fix? What are the different components needed for this system?*

DriverPass has tasked us with building a system that aims to prevent repeated DMV test failures, taking advantage of a void in driving test training systems on the market.

DriverPass should enable customers to access driving test data and practice exams both online and offline, and make reservations with drivers for driving lessons after providing a method of contact.

DriverPass, will consist of a cross-platform web-application built on React-native components in a Cloud-based container, connected to a database of users on the server.

Driving appointments will consist of packages that the user can select from, ranging from six-hours in a car to 12 hours + in-person lesson with online class content and material unlocked. The packages should be modular and easy to disable by an administrator.

### Objectives and Goals

*What should this system be able to do when it is completed? What measurable tasks need to be included in the system design to achieve this?*

The system (DriverPass web and mobile app), should provide the user with the following functionality:

* Access practice exams and user data online and offline (need to be online to modify any data).
* Choose from among different Driving packages:
  + Package One: Six hours in a car with a trainer.
  + Package Two: Eight hours in a car with a trainer, plus an in-person lesson where DMV rules and policies are explained.
  + Package Three: Twelve hours in a car with a trainer, an in-person lesson where DMV rules and policies are explained—plus access to DriverPass online class with all the content and material, including driving tests.
* Allow registration over a phone call or through online reservation system, using first name, last name, address, phone number, state, and their credit card number, expiration date, and security code.
* Allow choosing of picking location and drop-off location, and time of day from the app.
* Access to view online test progress if under package 3.
* Security, database, backups, redundancy, are all controlled in the cloud.

Administrators should have the following privileges:

* Track when a user makes any changes to a reservation, who cancelled or modified it last
* Print an activity report detailing time, user access info, and recent modifications
* Identify the driver a customer is scheduled to go with (track which user is matched with a particular vehicle, time, and driver)
* Access to view online test progress customers have taken

Drivers should be able to:

* Set time for lessons, leave notes, contact student/user

Application will adhere to DMV Compliance by being connected with DMV, for updates on new rules, policies, or sample questions. Allow a notification whenever DMV has a new update.

## Requirements

### Nonfunctional Requirements

*In this section, you will detail the different nonfunctional requirements for the DriverPass system. You will need to think about the different things that the system needs to function properly.*

#### Performance Requirements

*What environments (web-based, application, etc.) does this system need to run in? How fast should the system run? How often should the system be updated?*

* The environment should be **cross-platform**, meaning it should run on any web-browser, in any Operating System, and across any mobile device (with applications in their respective app stores).
* *Speed*: Strive for instantaneous access between pages, minimize load times. We could host on Amazon Web Service, making the server as fast and scalable as possible, with minimum server maintenance to allow our developers to focus purely on the business logic.
* *Scalability:* If we utilize Amazon Web Services, which offers “target utilization levels for multiple resources” in an intuitive interface, we can scale up to accommodate larger user base as needed.
* Updates to the system should be on a rolling release schedule from weekly to monthly depending on any important security patches needed. The project “packages” are modules which can be modified, removed, or updated as needed by an admin.
* *Redundancy and Recoverability*: Utilize “CloudEndure Disaster Recovery” from AWS to replicate and protect critical database stores driver and customer informations.

#### Platform Constraints

*What platforms (Windows, Unix, etc.) should the system run on? Does the back end require any tools, such as a database, to support this application?*

* REST API: Our DriverPass backend should be able to run on any Linux server, or within any virtual machine (for example VM on a windows machine), and may depend on additional softwares such as: Apache Web Server to host the server-side functions, Ruby on Rails server-side language, and PostgreSQL database server to store all user informations.
* Certain functionality will be serverless, hosted as an AWS Lambda for performance reasons. High latency background tasks such as compiling our PDF practice test results will be moved to a serverless function.

#### Accuracy and Precision

*How will you distinguish between different users?* *Is the input case-sensitive? When should the system inform the admin of a problem?*

* Admins will have a special [Admin] flag with corresponding additional access privileges.
* Case-sensitive input: Username / password information shall be case-sensitive for Admins. For users, only password information shall be case-sensitive.
* Admin: Passwords must be at least 10 characters in length, containing at least 1 alphabetical, 1 numerical, and 1 special character from amongst ($, @, ^, %, $, \*, &, !, ?).
* The system shall inform the admin of the following exceptions:
  + Log multiple incorrect username / password attempts (for admin and users)
  + DMV current laws and regulations are out of date, update practice tests accordingly
* Inform the admin if a database call took more than the predefined time
* Inform admin if memory cache or CPU are near capacity.
* Inform the admin of any transaction errors from a customers Bank or Debit card information
* Inform the admin of any function failure with a clear log message.

#### Adaptability

*Can you make changes to the user (add/remove/modify) without changing code? How will the system adapt to platform updates? What type of access does the IT admin need?*

* Roles should be tracked separately, theoretically any user could be easily added, removed, or promoted to admin. Keep all users (admin, and regular) in a single table; keep all roles (Admin, user, owner) in another table; have a key between the two databases to match roles to users.
* The server shall be taken offline for all regular users during a system update or patch, only Admins or Owner will be able to implement and schedule patches and updates.

#### Security

*What is required for the user to log in? How can you secure the connection or the data exchange between the client and the server? What should happen to the account if there is a “brute force” hacking attempt? What happens if the user forgets their password?*

* [Insert text]

### Functional Requirements

*Using the information from the scenario, think about the different functions the system needs to provide. Each of your bullets should start with “The system shall . . .” For example, one functional requirement might be, “The system shall validate user credentials when logging in.”*

* The system shall allow offline access to view practice tests, which shall teach basic driving informations and laws. While offline, edits to the test cannot be made.
* Driver: The system shall have a GPS navigation to show directions on where to instruct the student.
* The system shall allow searching in the database for past exams, driver information, and questions pertaining to driving laws and regulations.
* The system shall automatically backup all student information to the server periodically in case of system crashes or errors.
* The system shall comply with all DMV regulations, and provide a page linking to the newest applicable DMV rules and regulations.
* The system shall allow the student to rate / review the driver view a “5 star” + message, and vice-versa for the driver being able to review the student.
* The system shall allow user authentication and login via Gmail, Facebook accounts or a cell phone number.

### User Interface

*What are the needs of the interface? Who are the different users for this interface? What will each user need to be able to do through the interface? How will the user interact with the interface (mobile, browser, etc.)?*

* Beneath the driver information, on the application, the system shall have the phone number of their available driving instructor which can be clicked to call/message their driver.
* Student: Upon connecting with a driver, the interface shall change to a map of the current route, with GPS indicating current student location.
* The user will input username and password at the top of the screen (center for mobile), and press or click enter. The user will be brought to their homepage, which lists current packages, and suggestions of other possible packages based on their past purchases.
* The interface should show the current test progress, tests, and completed. It should read “test” “name”, “time taken”, “score”, “status”.
  + Status: not taken, in progress, failed
* For Drivers: User should be able to make comments, and give time availability for lessons, listing : Lesson Time, Start Hour, End Hour, Driver Comments to the administration log.

### Assumptions

*What things were not specifically addressed in your design above? What assumptions are you making in your design about the users or the technology they have?*

* Assuming the client is running Firefox, Google Chrome with Javascript enabled. Internet Explorer is **not** supported at this moment.
* It should run as an application in any Android Device. IOS not currently supported but planned for implementation.

### Limitations

*Any system you build will naturally have limitations. What limitations do you see in your system design? What limitations do you have as far as resources, time, budget, or technology?*

* Currently no plans to support Internet Explorer browser.
* REST API is implemented server side with Ruby on rails: currently relying on a more expensive Amazon AWS solution for certain serverless functions requiring intensive operations, for less throttling.
* Our REACT developers have inexperience working with IOS devices, porting could take some time.
* Currently browsers on Android redirect to the App Store; Android browser not planned.
* Full “OFFLINE MODE” as per user request story not implemented, however we have a function to handle lost connections gracefully. Offline mode planned for future release.
* More aggressive asset caching needed for large PDF memory operations to improve user experience.

### Gantt Chart

*Please include a screenshot of the GANTT chart that you created with Lucidchart. Be sure to check that it meets the plan described by the characters in the interview.*

[See next page for GANTT chart]

