# CS 255 Business Requirements Document Template

Karl Konz

Complete this template by replacing the bracketed text with the relevant information.

This template lays out all the different sections that you need to complete for Project One. Each section has guiding questions to prompt your thinking. These questions are meant to guide your initial responses to each area. You are encouraged to go beyond these questions using what you have learned in your readings. 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 your client’s needs.

**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?*

* DriverPass has observed many students who attempt drivers exams fail. They feel that their business vision offers a competitive edge to close the success gap by offering educational services geared towards improving driving test success rates.
* The system that DriverPass has envisioned includes an application that has business integrations into a call service for scheduling and in person on-the-road driver assisted lessons.
* The improvement in material accessibility will help ensure that the students can get help when they need it.

### 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 is investing in a variety of services in their system to help students be more successful in their driver tests, including:
  + online training resources
  + in person lessons
  + in person on-the-road lessons with a driver
  + online practice tests
* Students will be empowered to study whenever they wish to do so by allowing students to access materials through the application whether they are on or off-line, from a mobile device or a pc
* Student users should be able to make reservations for driving lessons by filling out a form online, or by calling the company to make a reservation with the secretary. The secretary will use the same form in the application to fill out the reservation for the student.
* The reservation system will intake the following information:
  + First Name
  + Last Name
  + Address
  + Phone Number
  + State
  + Credit Card Number
  + Expiration Date
  + Security Code
  + Pick up location
  + Drop off location
* Reservations will be for 2 hour long driving lessons
* User picks time and day
* DriverPass employees (admin and super-admin; Liam, and Ian) will need to be able to track which drivers are assigned to which students for lessons, as well as which car a student will be using, and what time and day a student will have their lesson.
* The company will have 10 cars available for lessons
* Users need to be able to reset their passwords
* Systems needs to be updated as notifications of changes for DMV policies are received
* Preference for cloud hosting

### 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 end users, driving students, of the system need to be able to:
  + Access scheduling form
    - Modify existing reservations
    - Cancel existing reservations
    - Create reservation
  + Include a contact page
  + Call to schedule reservation by phone
  + Access information about their profile
  + Online testing progress
    - Testing name
    - Test time
    - Test score
    - Test status
  + Take online practice test
  + Drivers notes for driving on-the-road lesson that includes the notes and time of the lesson
  + A picture of user
  + A picture of the driver
  + List any special needs for the student
* DriverPass secretary
  + Answer phone calls
  + Access scheduling form on end user student’s behalf
    - Modify existing reservations
    - Cancel existing reservations
    - Create reservation
* DriverPass admin (Liam CEO and Ian from IT)
  + Reset passwords
  + Remove/block user access
  + Disable packages from site
  + View and download user reports in tabular format (desire to analyze in Excel)
    - Who made a reservation
    - Who canceled a reservation
    - Who modified a reservation
    - Which student are assigned to which drivers
    - Which cars are used by which students
    - What time and day are users scheduled
* User Packages
  + Package 1 – 6 hours in a car with trainer (3 sessions of 2 hours in a car lessons)
  + Package 2 – 8 hours in a car with trainer and in-person lesson where we explain the DMV rules and policies (4 sessions of 2 hours in a car lessons)
  + Package 3 – 12 hours in a car with trainer, an in-person lesson where we explain the DBC rules and policies – plus access to our online class with all the content and material. The online class also includes practice tests (6 sessions of 2 hours in a car lessons)
* Desire for future ability to be able to create custom packages (not in scope for now)

## 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 DriverPass application should be mobile first to strengthen the goal of ease of access in course materials to best prepare students to pass their drivers exams.
* The DriverPass application should be set up in such a way that the backend is scalable. For ease of implementation and to reduce overhead this is recommended to be accomplished using AWS elastic load balancing. To ensure that everyone’s scheduling requests be completed, a queuing system will be implemented in case there are many students attempting to schedule services at the same time. Again, for ease of implementation and maintenance this is recommended to be accomplished using AWS SNS service. If a schedule request in a queue is no longer available by the time the transaction message is processed it should display a modal to the customer that the requested time slot is no longer available.
* The system should be updated regularly as updates become available. Only stable updates to components within the system will be implemented. To ensure continuous availability of the system, updates will be preformed with a build release pattern using something like Jenkins or AWS Cloud Formation. Additionally, the application and system should follow a blue/green deployment. In this, the updated system will be built and deployed in a staged manner while the existing system remains untouched. If the system needs to be reverted the older version will still be deployed and can easily be rolled back without needing to rebuild/deploy it.

#### 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?*

* The system should be mobile first and be platform agnostic. It is recommended that the application use React for the front and NodeJS for the backend. Additionally, historic records, account information, person data, and driver records should be stored in a database. For cost considerations it is recommended to use AWS DynamoDB.

#### 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?*

* Users will be identified by a unique identifier in the database. Additionally, they will have a user name that is also stored in the database. A user account should be primarily tied to an email address and a phone number. There will also be a user facing user name and a system user name. If there are duplicates of the same user facing name then the system name will add a suffix of an incrementing number, like what happens in Facebook for example.
* The input does not need to be case sensitive; this is to make it as easy as possible for users to access their accounts and study materials. This decision is in line with the company goal of having their customers pass their drivers exams at a higher rate than those who do not. The assumption is that accessing the materials more regularly, and not being blocked from accessing their materials, will increase passing rates.

#### 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?*

* The account class will have mutators and accessors to perform CRUD operations (create/read/update/delete) on objects tied to database in the account table. This means that changes to users will only happen in the database and do not require changing code.
* It is recommended that the application be hosted in a docker container in a serverless manner, for ease in implementation and maintenance it is recommended that the system be setup in a combination of EKS, AWS managed Kubernetes, and AWS lambdas for serverless execution of smaller operations. In this manner platform updates will largely be handled by AWS. By running the application through docker containers means that the application becomes more modularized and decoupled across the system.
* There will be a variety of IT admin access needs. DevOps will need to be able to access the Pod that the EKS is running on and have access to push changes to the application. This is recommended to be preformed through either puppet or terraform. Additionally, there will need to be access to the AWS account where the services are running and be able to view cloud watch logs to trouble shoot issues that arise. Finally, there will need to be application administrators who track user reported bugs, set permissions for users, and make changes to users accounts.

#### 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?*

* A user login will be required to have a username and password. Additionally, the user can opt in for multi-factor authentication. Admins can sign into the application with Single Sign On. The authentication of the application will be handled through OKTA and not need to be a completely customized solution for the DriverPass system.
* The connection of the front and backend will occur through several mediums. Firstly, there will be routing policies setup within the application as a way for the front end to access the services from the back end. Additionally, the repos for the front and back end will have the appropriate Access Control List settings as pertains to AWS with permissions from the application based on roles and policies defined within the code repositories to grant access to specific resources of the application, including the DynamoDB database.

### 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 perform CRUD operations of users
* The system shall perform authentication of users
* The system shall perform CRUD operations of user photos
* The system shall display user photos
* The system shall perform CRUD operations of user details; first name, last name, address, phone, and email.
* The system shall perform CRUD operation for scheduling in-person drivers lessons
* The system shall differentiate what types of package(s) a user has
* The system shall perform CRUD operations on the types of packages offered through the app
* The system shall perform CRUD operations on payments from students
* The system shall make learning materials available to users
* The system shall report on historic test data for users
* The system shall report on in-person driving lesson data for users
* The system shall report on driver and vehicle availability
* The system shall preform CRUD operations on learning resources
* The system shall perform CRUD operations on driver notes
* The system shall report on driver notes

### 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.)?*

* The interface should include the following:
  + The logo should be displayed at the top of the page.
    - This will be static to students and drivers and can be updated by admins
  + Online test progress
    - Students will see what tests they need to take as well as what test they have already taken and how they performed on each test and what time they took the test.
    - Drivers will be able to gauge the student’s ability by seeing their test progress and will know what areas the students will need to focus on in their lessons.
  + Drivers’ notes
    - Students will see what notes were left by the driver after their lessons, this will include the driver’s name, lesson time, start hour, end hour, and comments.
    - Drivers will be able to enter the information pertaining to the students’ lessons.
  + User information
    - Students will see their first name, last name, address, phone, and email. Students should also be able to update this information as needed.
    - Drivers will be able to see the information for a student. This will be important so the driver knows where to meet the student and can contact them in case there are any issues.
  + Special needs
    - Students will be able to add any special conditions and needs for the drivers. They should be able to update this information as needed. For example, a student might have a cast on their leg due to a broken bone at one point in time and should have the ability to remove the note about their cast once it has been removed.
    - Drivers will be able to read the special needs for a student and be able to proactively prepare for the specific needs of the individual student prior to their lesson.
  + Photos
    - Both students and drivers will be able to see photos of each other in the application.

Additional interface pages:

* Scheduling page for creating/viewing/updating and deleting in person lessons. This can be used by the secretary or students to schedule their lessons.
* Payment page to buy any of the three packages. Admins and developers can update the packages and students can submit payments
* Admin page that shows users, permissions can be set for individual admins in this page.

### 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?*

* Users of the system are assumed to have internet access to download the application and to receive and view updates to their account. It has been requested that they be able to access course materials whether on or off line. However, in order to see information updated as they take exams and additional driver and lesson information gets updated, they will need to be connected to the internet.
* The volume of users is not known, due to this reason the application it is assumed that the application will need to be setup in a scalable manner.
* The data retention is not known in regards to how long will a user’s data be persisted in the databases. Due to this not being defined it is assumed that user data will not be deleted.
* The Service Level Agreements have not been set therefor it is assumed that there are no thresholds expected in the time needed to resolve system issues.

### 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?*

* Since users will be put in a queue for scheduling appointments there is a possibility that if a student requests a certain time, by the time their schedule creation transaction message gets processed that time will no longer be available.
* While the budget was not articulated for this project it appears to be a rather small project and therefore will likely only have a couple dedicated developers while building out the application. One of the bigger concerns will be post release and how quickly we will be able to address issues or feature requests that come through while balancing other projects and customers.

### 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.*

