# CS 255 DriverPass Business Requirements Document

Table of Contents

[CS 255 DriverPass Business Requirements Document 1](#_Toc95077541)

[Terminology 2](#_Toc95077542)

[System Components and Design 2](#_Toc95077543)

[Purpose 2](#_Toc95077544)

[System Background 2](#_Toc95077545)

[Objectives and Goals 4](#_Toc95077546)

[Requirements 9](#_Toc95077547)

[Nonfunctional Requirements 9](#_Toc95077548)

[Functional Requirements 13](#_Toc95077549)

[User Interface 14](#_Toc95077550)

[Assumptions 18](#_Toc95077551)

[Limitations 19](#_Toc95077552)

[Gantt Chart 20](#_Toc95077553)

[References 21](#_Toc95077554)

## Terminology

For the purpose of this document, terms are defined as follows.

“Client” will refer to DriverPass.

“Student” will refer to a customer of DriverPass.

“User” will refer to a user of the system, either a student or a DriverPass employee.

## System Components and Design

### Purpose

* The purpose of this system design is to enable DriverPass to improve the driver training and minimize the likelihood of failing a driving test by enhancing existing the existing infrastructure by offering a web-based software solution for tracking individual progress, reserving lessons, and viewing current Department of Motor Vehicles (DMV) information.
* An additional purpose of this system design is to enable DriverPass to take advantage of a void in the market regarding training for the driving test by offering a web-based solution to augment the current DriverPass systems.

### System Background

* The system background of this system design is that DriverPass desires software that will allow students to take online classes and practice tests and will maintain all student records in addition to providing a way for students to make reservations for on-the-road driving training appointments.
* Package is the term for a purchasable set of products.
* Additional system background is that the driving packages that the solution will help support are: a package including six hours in a vehicle with an instructor, a package including eight hours in a vehicle with an instructor and an in-person lesson where a DriverPass instructor will explain the DMV rules and policies, and a package including twelve hours in a vehicle with an instructor, an in-person lesson where a DriverPass instructor will explain the DMV rules and policies, plus access to the DriverPass online class with all the content and material, which will also include practice tests.
* Also, the client would like to create, update, enable/disable, and delete packages. The client was told that “it can’t be built in a way that a non-developer can do easily”, but it can. This is much less complex than many of the other client requests, such as storing credit card information which was not even questioned.
* DriverPass would like to update data whether they are online or offline. Contrary to what the client was told, this can be done. It can be accomplished by saving the changes between the state of the data locally in a log, and then replaying the log when the system reconnects. When the system is synchronized, any data that was stored offline is attempted to be merged in with the online data, if the data was only modified in one place, the merge is simple; if the record was modified in two places, then that will be identified as a merge conflict. Merge conflicts can be displayed to the user to resolve in addition to providing the option to throw away local changes that were done offline or accept all changes that were done offline and throw away any changes on the remote system. According to Fowler (2017), this is Event-Sourcing because “we can confidently rebuild the system state by reprocessing the events at any time in the future”.
* In addition, DriverPass would like to download “the reports and some information” for remote processing potentially using Excel. No further information is available about this because rather than diving into more detail about these reports with the client, the interviewer decided it was time to talk about security. Based on the context, the reports and information may be something that can be logically stored in a spreadsheet.
* Other background about the system is that the administrator must be in control of resetting employees’ passwords when they forget them and be able to revoke access when they are fired. It has been requested that administrators also have access to maintain and modify the system.
* The system will also track changes of all records - another benefit of the event-sourcing technique of event driven architecture - so that the client can assign blame when anything is done incorrectly. The report will include the user that created, cancelled, or modified a reservation, and must be printable.
* Additional background is that the owner of DriverPass calls himself “The Big Boss”.
* Session is the term for a two-hour time span in the vehicle with a DriverPass instructor.
* The client has requested that students are not able to create their own account, but they must instead call, and an employee will create their account. The student will provide their first name, last name, address, phone number, state, credit card number, expiration date, security code, pickup, and drop-off location. The drop off location will default to the pickup location when it is filled out but may be modified. It is recommended that in addition to the above data, the system also accept the email address, as well as drawing a distinction between the mailing address and the billing address. If the below suggestion about utilizing a third-party service to handle billing, then the system should only have the ability to store an optional mailing address. In addition, if the email address is not used as the username, which if the email is not required to be the username, it does allow for better security, but there would also need to be the addition of a username field in that case. In addition to acknowledging this requirement, it is recommended that the system allow self-serve users to create their own student accounts. The addition of the interface required to do so would be trivial compared with other aspects of the system.
* The system will require an integration into the application program interface (API) made available by the DMV to meet the requirement that the system be “connected to the DMV … [for] new rules, policies, or sample questions”. According to the interview there is some form of notification when these updates happen, but the interviewers failed to ascertain any further details on how that might occur. Assuming that the DMV does not have a public API in order to service this requirement, the implementation of this requirement will be accomplished by using an extract, transform, and load (ETL) pipeline. The ETL pipeline is described in detail later in this document.
* The system should allow either the student to create a session reservation on their own account or allow an employee to create a reservation on their behalf to support the ability for students to make session reservations over the phone or by visiting the office.
* DriverPass currently only has 10 vehicles with each one assigned to an instructor.
* The system must identify the instructor that the student has a session scheduled with. It appears that DriverPass desires the assignment of an instructor to a student session to be done automatically when the session is reserved. After the session is reserved, the identity of the instructor can be viewed from the online user interface.
* Students must be able to create reservations through the online interface, as well as use the forgot password link. It is recommended that students also be able to create their own account to prevent them from always having to call to do that.
* The client does not want to deal with backup and security, so it is recommended that the system be hosted on a platform as a service (PaaS) environment.
* The system must display the status of tests to the student. Metadata of the test that will be displayed is the name of the test, the time it was taken, the score the student earned, and the status. The time and score must be nullable because tests that the student has not yet taken will be displayed. The statuses that each test could be in are not taken, in progress, passed, and failed.
* On the web interface, there must be a section for instructor feedback that includes a table with Lesson Time, Start Hour, End Hour, and Driver Comments. The instructor is the one that fills out comments and feedback for the student with regards to the session. Instructors will need to have an interface that allows easy entry of feedback on each session.
* The student profile must be available to view and edit for both students and employees.
* There must be a contact page on the website to allow students to contact DriverPass.
* In the event that a DriverPass employee needs to contact a student, there must be a user interface (UI) to allow them to look up the student’s contact information.
* In addition to the other things that we overlooked during the interview, the system must also store images of both the instructors and students to display as part of the UI.
* The User interface crayon sketch provided also has a special needs section. No information has been provided about this component; therefore, further information must be gathered from the client.

### Objectives and Goals

#### Objectives

* User management
  + The system must include authentication for users via username and password authentication. It is recommended that the system also support multi-factor authentication (MFA), such as authenticator apps, text messages, or push notifications. The system must restrict user permissions into the following groups:
  + Administrator – Administrator users must be able to revoke the access of any other user, including administrators, send password reset links to any other user, as well as have all the permissions that an employee has. Administrators must also be the only ones that can create, update, delete, and enable or disable packages.
  + Employee – Employee users must be able to create student accounts, create reservations, view sessions across all students, and view individual student account details. Viewing sessions should be in the context of a calendar that shows all the sessions and allows employees to click on specific sessions for more information.
  + Student – Student users must be able to create and manage their own student account and create and manage reservations. Students must be able to reset their password using the forgot password link.
* Securely store student records
  + Personally Identifiable Information (PII) – According to Thomson Reuters (n.d.), PII “can be defined as data relating directly or indirectly to an individual from which the identity of the individual can be ascertained. Examples of PII include clients' names, addresses, phone numbers, social security numbers, and financial account numbers”. It is recommended that the amount of PII collected about each student be reduced to the minimum level possible. The persistence layer must be secured to prevent unauthorized access, and it is recommended that high risk data be encrypted at rest. It is also recommended that, in addition to the student data described in the interview, the student also be required to provide an email that can be used for reminders and notifications, and of course gratuitous advertising emails. In addition to the billing
  + Credit card information – Given that among the requirements in order to be payment card industry data security standards (PCI DSS) compliant, commonly known as PCI compliant are, according to Dahn, an annual PCI DSS Self-Assessment Questionnaire (SAQ), a quarterly network scan by an Approved Scan Vendor (ASV), and an Attestation of Compliance (AOC), it can be rapidly determined that based on the client requirement that “We do not want to deal with backup and security; we need that to be taken care of”, that the system will not be transmitting or storing credit card information. Instead, a third-party platform such as Stripe, Square, or PayPal will be implemented, and their UI will be embedded in places where the entry of credit card information is required. It is also recommended that rather than requiring students to call in to register, that they be allowed to create their own account to eliminate the handling of credit card information by employees.
* Service Availability
  + Online or offline - The service must be available to users with or without an internet connection. When users are online, the changes made are synchronized immediately using event sourcing, but when they are offline, the logs of any changes are stored locally and synchronized all at once when the device is reconnected. Merge conflicts may occur if data is edited online and offline at the same time, so an easy-to-use interface for resolving merge conflicts must be implemented.
  + Desktop or mobile – The website must be easy to use on both a computer and a mobile device. It is recommended that this requirement be fulfilled by using reactive design in the web page so that forms automatically adjust to best fit the window size available.
* Logging and reporting
  + The recommended use of event sourcing for data persistence has the ability to solve the request for logging and reporting of the creation, modification, and cancellation of reservations in addition to every other change that is saved to the data persistence layer by creating a log containing all data required to rebuild the application state from scratch using just the logs. Those logs can then be output into any desired format including, but not limited to Excel. To provide the information that is most beneficial to the client in the reports, the client must provide further information than their desire to assign blame to a specific user when data is updated in error. Some examples of the beneficial ways the logged information can be aggregated is to identify for example, the views on each individual package to determine what is most popular and derive ways to tweak packages that will result in an increase in profitability. Another way that reporting might be useful is to identify sections of the coursework where students are having an unusually difficult time on a particular test question for example. If the distribution of incorrect answers is uneven, then more focus can be put on educating students on the principles behind the answer to that question, or perhaps just raw memorization in some cases.
* Packages
  + The initial packages that will be provided with the system are, package 1: 6 hours of driving sessions, package 2: 8 hours of driving sessions and an in-person classroom lesson on DMV rules and policies, and package 3: 12 hours of driving sessions in addition to the classroom lesson, online class, and access to practice tests.
  + Packages should be architected in such a way that they are easily modified, added, and removed. At a minimum, the client has requested that they be able to manually enable and disable the packages. Based on the simplicity of the request to allow them to be modified by the client, it is recommended that the interface include the ability for administrators to create, modify, and delete packages in addition to enabling and disabling them. If the recommendation to enable the client to update packages is not followed, then apparently, we are just going to hard-code them and require a developer or a system analyst to make changes. Another severe disadvantage of following this method to not allow the client to update them is that they will also lose the logging record that seemed to be so critical to them regarding reservations for sessions.
  + Packages must also be purchasable by students. Students will be able to select from the available packages after they have signed in with their student account. They will be able to explore the different options and when they have selected, they can purchase it online, ideally using the third-party credit card processor that is recommended to handle all transactions requiring PCI compliance.
* Reservation of Sessions
  + Based on the packages that are enabled and which one is selected by a student, they will need to reserve times for when the driving sessions will take place. The client has 10 vehicles and 10 instructors, one for each vehicle. Students will be able to reserve session times by either using the website to create a reservation, calling the client, and having an employee enter the reservation, or by visiting the DriverPass office in person to have an employee enter the reservation. Regardless of the method of entry, the reservations can be modified or cancelled in any one of the three ways as well.
  + During the reservation process, an available instructor and vehicle will be assigned to the reservation. The availability of these vehicles and instructors is a constraint that will require the system to have knowledge of availability prior to allowing reservations. It is recommended that there be a calendar view of the available times along with a count of the instructors that are still available for any given time slot. This system knowledge of instructor availability and time slot popularity can be another example of a constructive use of analytics.
  + The client must provide the available time slots. The client has indicated that they would be in two-hour blocks, but it is recommended that the instructors be allowed time to eat and allow time to travel from one student drop off location to the next student pick up location since the sessions are unlikely to begin from the DriverPass office location if the student is given the choice.
* Tests and practice questions
  + The client has requested that the online platform contain current information regarding the DMV’s rules, policies, subject materials, and test sample questions. Because the New York (NY) DMV does not appear to have a publicly available API, the information will most likely need to be retrieved using an ETL pipeline. The ETL will extract the data from the content from the New York State Driver's Manual & practice tests provided by the New York DMV (2021). The data can be extracted using a simple web scraping application. Note: web scraping is simple, but it is also brittle, so it is likely that there will be many things done over time to the NY DMV website that are breaking changes to this ETL that will require many additional hours of development that should be included in cost analysis. The transform part of the ETL takes the data retrieved from the extract part of the operation and transforms the data models into the format that is needed for the DriverPass application. The final part of the ETL pipeline is to reliably load that data into the client’s system.
  + The client indicated that they would receive a notification whenever the DMV has an update, but none of the critical details about that notification were requested. If the notification is by mail, it is irrelevant to the platform, but some means of electronic notification may allow for the ETL to be automatically triggered.
* Contact
  + As part of the online interface, the students will have access to the DriverPass contact page that will provide them with the phone number, email address, and physical address of DriverPass. Contact pages are a standard feature on most business websites and will be implemented in a similar manner to the generally accepted format.
  + Any contact information that is provided by the student will be accessible to authenticated DriverPass employees or administrators. The student may provide their phone number and an address as part of the basic information suggested in the interview, but it is also recommended that the email address be captured.
* Backup and security
  + The client has indicated that they are not interested in managing backup and security and that must all be taken care of for them. Given that they desire the management of those aspects be transparent to them, it is recommended once again that the software be hosted on a PaaS provider such as Microsoft Azure or DigitalOcean.
* Web interface
  + The client has requested that the web interface appear according to the rough drawing provided. With the addition of other screens that will be needed to make the interface usable, the provided interface will be used as the base for the look and feel of the interface overall. The objective of the interface is to allow users to accomplish all other objectives form an easy to user interface that requires less then 30 minutes of training for administrator users, less than 15 minutes of training for the average employee user, and zero training for student users, who can sign up online and immediately begin using the platform. Further details regarding the web interface are included in the User Interface section.

#### Goals

* User management – The goal of user management is to provide access to authenticated users to the minimum number of resources required for them to accomplish their required tasks.
* Securely store student records – The goal of securely storing student records is to minimize the risk and mitigate the effects of a breach.
* Service availability – The goal of being available both online and offline is to allow updates to the system to occur without heavy reliance on an internet connection. Updates can be made, and they will synchronize when a connection is available. The goal of being accessible from both a desktop and mobile platform is to provide access to as broad a range of users as possible.
* Reporting – The stated goal of the reporting feature was to “figure out who is responsible”, but a more healthy and constructive goal might be to identify if employee training is necessary or if students were having trouble using the website.
* Packages – The goal of packages is to enable the client to create and update compelling packages that can be purchased by students online.
* Reservations – The goal of the reservations system is to make it easy for students to select the times that they would like online, and also make it easy for DriverPass employees to view the availability of instructors.
* Tests and Practice questions – The goal of having tests and practice questions be the latest information from the DMV is that it will give students the best possible chance to pass the written test and minimize the risk that old data will result in a missed question.
* Backup and Security – The goal of the system with regards to backup and security is that the client never has to deal with backup and security. That must all be transparent to the client.
* User Interface – The goals of the user interface are to present the data in an easy to use format and allow all user types to interact with the data to meet all use cases in the simplest and easiest to use method.

## Requirements

### Nonfunctional Requirements

* The system must be integrated with a third-party Credit Card Processor.
  + Rationale: To fulfill the client’s request that they not deal with backup and security, offloading the tremendous security burden of becoming PCI compliant to a third party is imperative.
* The system should make use of a third-party Authorization Server with custom user roles.
  + Rationale: to fulfill the client’s request that they not deal with backup and security, offloading the proper handling of user credentials to a third-party is recommended to increase the speed of application development and reduce the amount of code that must be maintained.
* The system should make use of a Persistence Layer (use browser local storage to persist data while offline)
  + Rationale: The application state must be persisted over time to keep track of packages, sessions, and students. A persistence later of one sort or another will make this possible.
* The system will be Integrated with the DMV System by an ETL pipeline
  + Rationale: The New York State DMV does not appear to have a publicly available API, so it is likely that the only way to load the data into the system is through an ETL pipeline.
* The system must track Instructor availability or have access to an API that provides that information.
  + Rationale: Student users must be able to reserve sessions either directly in the system, or through a DriverPass employee. Given the limited number of vehicles and instructors, there
* The interface must honor the browser back button to return to the previous screen, including any rule that will prompt a user for if they would like to save changes etc.
  + Rationale: Users expect that the browser back button will take them back to the previous screen, and they also expect to be warned before data loss occurs.

#### Performance Requirements

* The front end of the application should run very rapidly once all required information has been sent to it, however there may be a delay when the application is first loaded because the application state will all be synchronized to the browser. One way of doing this to minimize impact on the user is to prioritize sections that will be used first, and allow navigation while data is being loaded in the background. Once this synchronization has been completed, provided that the browser is connected to the internet, further communication should be very rapid. Developers should strive for under 500 millisecond (ms) response times on all HTTP requests after the application state is already synchronized. Requests over this 500 ms limit are to be reviewed and the justification for the response time documented and reviewed periodically, with the period for reviews to be determined.
* The UI should be performant, meaning that when a user interacts with a component, the component immediately reacts in some way to let the user know that their interaction has had some effect. Examples of this include keyboard and mouse events such as click, hover, drag, etc.
* When buttons are clicked that should initiate an action, there should be some indication that the action is in progress and the page should ignore duplicate clicks on buttons that have already been pressed. For example, if a user clicks a submit button, the button should be animated to indicate the tap, should ignore additional taps, and should display some progress indication animation to the user that the submission is in progress.
* Loading animations should be displayed while waiting on any HTTP request, regardless of how brief the loading period is. If the loading periods are expected to be very short, the UI must be designed in such a way that a rapidly appearing and disappearing loading animation is not jarring to the user and does not move the content on the screen.
* The system’s software should only be updated as-needed, not according to a schedule.

#### Platform Constraints

* Backend Platform constraints: Internet accessible hosting service capable of running Windows, Linux, or Macintosh. It is recommended that the backend of the platform be developed in the .Net 6 framework, and as a result must be hosted on a platform compatible with .Net 6. It must also be internet connected in order to allow users to access it remotely.
* Data persistence layer constraints: A central relational database is not strictly required as a tool to persist data for the application. The system will make extensive use of caching and will store records in browser storage to enable offline use. In some cases, browser local storage is referred to as a database, and for good reason. Browser local storage is a database. In practice, the requirement of having a “database” is often used by individuals other than the software architect to dictate an implementation detail that they should not be dictating. The server-side data persistence layer must meet the minimum requirements that it can store the log records of all changes as they are entered, store a snapshot of application state at an interval to be determined, and return log records based on requests and user permissions. In the event that during development it is discovered that a SQL database is the best way of accomplishing those requirements, then a SQL database will be used, and the same is true of any other storage mechanism, such as elastic search, Redis, and flat files. Until the determination has been made about which is the most appropriate, the data persistence layer will default to a flat file. This should not be deemed a constraint however, engineers working on the system should do whatever makes the most sense in each individual case.
* Front End constraints: To use the platform, users must use a browser that follows web standards. This will allow maximum compatibility with other modern browsers. According to Mozilla (2022), “Web standards are the technologies we use to build web sites. These standards exist as long technical documents called specifications, which detail exactly how the technology should work”. This constraint will limit users to modern browsers which with, chances are, they are already familiar. It is recommended that the front end is developed in a common JavaScript based framework such as Vue.js or Angular to simplify development and accelerate time to market and make it easy to find developers that are familiar with the framework.

#### Accuracy and Precision

* The system will distinguish between users based on roles configured on the authorization server. The default role for a new user is a student if not explicitly set to a different role upon creation. Only administrators have permission to create non-student accounts. Users are identified by a unique email.
  + Rationale: This strategy attempts to minimize the number of permissions that any given user has to only the permissions they absolutely need.
* User emails are not case sensitive.
  + Rationale: Users are not likely to remember capitalization of an email address unless they type it the same way every time. Emails are unique, so there will not be a conflict between two email addresses with different capitalization.
* The system will log all errors in a cloud accessible error logging platform such as Application Insights or DataDog.
  + Rationale: the utility of web-based logging is that it makes the timely reporting and reaction to server errors more efficient. Error rules can be set up that notify specific users about rule violations, so they can respond quickly. Rules can be set up to notify the team in the event of a certain number of errors per minute, the standard for this should be set low initially and then tuned once the application is in production. A rule can be set to notify the team of any 500 Server Error. In addition, the web-based logs make the troubleshooting and resolution of issues easier in a time where many software engineers work remotely.

#### Adaptability

* Users of any role can be modified without changing code. Administrator users have permissions to edit other users’ roles. All other user types have permissions to edit their own user profile, but not their user role. Employee users can create student users on behalf of a student who calls in or visits the office, and students can create a student user when they sign up.
  + Rationale: there is no good reason to have any user specific data hard coded. Every user and every role will be managed by the authorization server.
* Reservation Packages can be modified by Administrator users.
  + Rationale: Currently there is only one role that has permissions to add, modify, delete, and enable/disable packages, but additional roles could be easily added to the authorization server if this requirement changes in the future.
* Session Reservations can be added, and modified by any user type, with the limitation that Student users can only add and modify session reservations for their own user, while employees and administrators can add session reservations for any student user.
  + Rationale: Student users should not be able to modify any user’s data but their own, which DriverPass employees may need to modify a session reservation for a user who calls in or visits the office.
* The system will adapt to platform updates by continuing to run without interruption due to fully testing the application with the platform updates applied in a non-production environment before rolling the updates out to production.
  + Rationale: platform updates have the potential to cause a service interruption or outage and must be undertaken with care. Sometimes platform updates include security related patches, so they must be applied expeditiously. In those cases, the updates will be treated with the appropriate urgency, which will be based on the security issue that is being resolved.
* The IT Admin will require the Administrator role.
  + Rationale: the highest access possible for a user is the administrator role, and at least one user must have the administrator role to grant permissions to other users and take care of administrative functions. The IT Admin does not require access to the code.

#### Security

* Industry best practices for Login security will be implemented by using a third-party authentication system such as Auth0 or FusionAuth.
  + Rationale: In the event that a malicious entity attempts to attack a company that offers driver license training and has ten vehicles by using a “brute force” attack, they will quickly (after 3 attempts with the incorrect password) find themselves locked out of the system. In addition, third-party services that offer authorization have the sole focus to provide state-of-the-art security and monitoring. It is likely that a small company such as DriverPass, only having 10 vehicles, is not a target of interest to any bad actors, so the implementation of industry leading security practices is an excellent deterrent. The authentication service will be configured to allow multi-factor authentication (MFA), password complexity and length requirements, failed login attempt account locks, and a password reset link.
* Sensitive user data will be encrypted at rest or omitted from browser local storage.
  + Rationale: This application as proposed makes use of browser local storage to enable offline access to data. As a result, administrator users will have access to sensitive user data. The client will determine which data is too sensitive to store in browser local storage, and that will only be available when connected to the internet, while any data that is user data but is not sensitive enough to warrant omission as determined by the client, will be stored encrypted in browser local storage. The repository containing system logs and application state on the server side will store sensitive data encrypted and use access control to verify that only authorized users, as determined by the client, have access to that data.

### Functional Requirements

* The system shall validate user credentials when logging in.
* The system shall only provide access to resources to users authorized to access that specific resource.
* The system shall allow administrator users to reset the password of any user account, including other administrators.
* The system shall allow administrator users to block the access of any user, including other administrators.
* The system shall allow administrator users to modify packages, including adding, removing, modifying, and enabling/disabling packages.
* The system shall allow employee and student users to view instructor availability.
* The system shall allow employee users and student users to add, update, and remove session reservations based on instructor availability.
* The system shall display instructor availability in a user-friendly manner during the session reservation process.
* The system shall allow employee users to view all current reservations and all student data.
* The system shall allow student users to view and modify only their own student data.
* The system shall allow students to purchase packages.
* The system shall allow administrator users in the offline state to create, modify, delete, and enable/disable packages.
* The system shall synchronize data from offline sessions when it is reconnected to the network.
* The system shall clearly indicate changes that have not been synchronized to the online system.
* The system shall enable administrator users to resolve conflicts between data that has been modified offline, and the current application state when the data is synchronized.
* The system shall be available on both desktop and mobile platforms.
* The system shall make activity reports available for download of creation, modification, and deletion of any record including the user that took the action.
* The system shall make DMV learning materials available for students to study.
* The system shall make DMV practice tests available for students to take.
* The system shall keep DMV provided materials up to date.
* The system shall record the time that a test was taken and the score.
* The system shall display all available tests and their status (not taken, in progress, failed, passed).
* The system shall allow instructors to enter feedback for the student about each session.
* The system shall display all completed sessions in a table with start and end time and the instructor’s comments about the session.
* The system shall display DriverPass contact information.
* The system shall display student contact information to employee and administrator users.

### User Interface

* The user interface for the login screen provides input areas for username and password. There will be a button for “Forgot Password” and a button for “Login”. The login button will be disabled until the at least one character has been typed into both the username and password fields. The UI will display any error messages returned from the API to the user. In a future phase, the client has the option for federated logins from third-party providers such as social media platforms.
* The user interface for the forgot password screen provides an input area for the user’s email and a button to “Send Password Reset Link”. The send button will be disabled until there is a valid email typed into the email field.
* When an administrator logs in, they will be presented a screen with basic analytics, which are to be determined by the client. They will have options for Packages, Students, Sessions, and Users. In addition they will see the profile button on the interface that is common to all user types. Clicking on one of the options will take them to the interface described below for the administrator view of each of those pages.
* The administrator view for the packages page allows the administrator user to view a list of all reservations or filter them by if they are enabled or disabled. Further facets can be added to the filter in a later phase at the client’s request. At the top of the interface, there will be an element that indicates if the system is in online or offline mode. Clicking on a single package will take the administrator user to the edit package interface. Each package in the list will also have a “kebab” style menu (the three dots) that will allow toggling of the enabled/disabled state, deleting the package, or navigating to the edit package screen. In the event a user clicks to delete the package, there will be a confirmation dialog that confirms the user would like to delete the package. The data persistence layer will not truly delete these records, but simply add an IsDeleted flag to them and omit them from any future queries.
* The administrator view for the edit package screen allows the administrator user to modify the fields in the Packages model. At the top of the interface, there will be an element that indicates if the system is in online or offline mode. The screen will have a Save button that is disabled when there are no changes between what the user has entered and the application state. When the user makes a change, the save button will be enabled. Saving the record will create a message in the log with the differences between the application state and the updated model and send that to the service responsible for recording changes and synchronizing those changes when connected to the internet. The edit package screen will have a back button that takes the user back to the packages list page. If there are unsaved changes in the form when the back button is pressed
* When the system synchronizes, if there are data conflicts, the interface will present a pop-up message offering to let the user resolve the conflict now. If the user agrees, they will be taken to the conflict resolution screen. If the user refuses, then an icon indicating that the system is out of synchronization will be displayed in the interface according to UI designs to be determined. The icon can then be pressed at any future time to get back to the conflict resolution screen.
* The conflict resolution view will show the changes that the user made, and the remote changes that are available on the server on an individual record basis if there are more than one record in conflict. The user will have buttons to accept theirs changes or the remote changes as a whole or select specific changes from each. They can also go to the next record and previous one throughout the entire list of records in conflict. Once satisfied they can accept each record’s changes individually which will then be synchronized to the server. The conflict resolution screen has an ‘x’ button that closes the conflict resolution screen. If there are any conflicts remaining when the conflict resolution screen is closed, the icon indicating that there are still conflicts will still be displayed on the UI.
* The administrator view for the user screen will display a searchable list that can be filtered by user type. On each user there will be the ability to add and remove roles, reset password and delete the user. This screen will be developed last because the authorization server will already have a user management interface that can be linked to if necessary.
* When an employee logs in, they will be presented a screen with basic analytics to be determined by the client. They will have options for Students and Sessions. In addition, they will also see the profile button on the interface that is common to all user types. Clicking one of the buttons will take them to the employee view for that page type.
* The administrator and employee view for the Student List page is a searchable list of all students. Certain facets of the student record will be displayed in the list such as name, last login time, etc. based on UI designs to be determined. When an employee or administrator click on a specific user, they are taken to the user details page.
* Graphical user interface, application

  Description automatically generatedWhen a student logs in, they will be presented with the student details page. A rough mockup of the student details page is below:
  + The DriverPass logo will be displayed at the top, centered on the page.
  + There will be a menu option for sessions that takes the user to the session reservation page and the package purchase page.
  + There will be a table displaying the status of all DMV tests which shows the name of the test, the time it was taken, the score received, and the status of the test (Not taken, In progress, Failed, and Passed).
  + There will be a table displaying previous sessions results including Start and End Times, and Instructor feedback for the student.
  + The page will display a picture of the instructor for any upcoming sessions, or if there are not any upcoming sessions, the previous session, or if there are no past or future sessions, the instructor photo will be blank and display a message encouraging the student to reserve a session instead.
  + The page will display a photo of the student and allow the student to upload a new photo by clicking on the image or empty frame where the image would be. The upload image dialog will be a simple image upload and crop dialog from whatever front end framework is used to develop the application.
  + The page will display a special needs section. More information about the purpose of this section is required to be provided by the client.
  + The page will display the user details and an edit button. If the edit button is clicked, a save button appears and the data can be modified in place. When the save button is clicked, if there were any changes to the user profile, they will be updated in the persistence layer.
  + The page will display a blank rectangle above the user details. More detail is required on the purpose of this frame to be provided by the client.
* The employee session reservation page is accessible from the menu bar for Administrator and Employee users. This page is similar to the below student session reservation page with the exception that employees and administrators can select the student for which the reservation is being made from a list on the page. If a student is not selected, the upcoming session list displays past and future sessions for all students. The session list can be filtered by date, instructor, student, and other facets as determined by the client. Clicking on a specific past session will take the user to the session feedback screen. Clicking on a future session reservation will take the user to the edit session reservation screen. The instructor availability is still displayed and clicking session slots with availability still displays the available instructors, but it does not allow the reservation of a session without the context of a student. Once the student is selected, a reservation can then be made.
* The student session reservation page consists of a list of existing session reservations for that student, and a calendar that shows the count of available instructors during each session time slot. Selecting a time slot that has at least one available instructor pops up a list of instructors to choose from, in addition to a “Surprise Me” option. With the session time and instructor selected, a reserve button becomes enabled. When clicked, the screen for users to reserve a session. If the student has already used all the sessions, they are allotted with the package they purchased, the calendar is disabled with a link to purchase another package.
* The purchase packages page displays a list of the available packages and what they offer. When a student is ready to buy, they click the buy now button that will take them to the third-party credit card processing service’s component and they will be able to make their payment. When the payment is completed, the student will be redirected back to the main student page.
* The edit reservation screen allows the administrator, employee, or student user change the date, time, and instructor of a future session. Modification of future sessions will be restricted from students at a certain time before the session begins and will display a message that if a student needs to make a change, they must call or visit the office. When a change is made to a session reservation, a save button will be enabled.
* The session feedback screen allows employees to enter feedback for a student with regards to a specific session. The entry form will be a text box that supports text formatting (only bold, italic, and underline to start with), and a save button. If the client wants to get fancy with the text formatting or add emoticons or whatever, that will be a scope change.

### Assumptions

* It is assumed that the client will support the use of a third-party credit card processing service such as Stripe or PayPal.
  + Rationale: The rationale behind the use of a third-party credit card processing service is to eliminate the requirement to make the system PCI compliant. Implementing a PCI compliant credit card processing system was clearly not accounted for in the timeline provided during the schedule planning meeting.
* It is assumed that the client will support the use of a third-party authentication provider such as Auth0, FusionAuth, or another equivalent service.
  + Rationale: The rationale behind using a third-party authorization server is to minimize the number of lines of code that must be maintained for the life of the product. Using a third-party is effectively paying them to maintain and update that component.
* It is assumed that native mobile applications are not being developed, but the system will be developed as a responsive design single page web application.
  + Rationale: Responsive web design is a good place to start for getting the software out quickly especially for a first iteration. Based on user feedback and analytics, determining if a native mobile all is required can be done while the client is already making money from the first phase of the system.
* It is assumed that the back-end API for the platform is hosted on a cloud provider’s PaaS environment such as Microsoft Azure or DigitalOcean.
  + Rationale: The client has stated that they do not want to deal with system backup or security, and online providers take care of all that for them for a reasonable fee.
* It is assumed that student users will be allowed to sign up online without having to call and have an employee set them up in the system.
  + Rationale: The Student user model is so simple that designing an interface for self-service to set up a new account is almost trivial. Given all that other complexity that has been adamantly requested, this can be easily included and will enhance the experience of do-it-yourself (DIY) student users.
* It is assumed that the client will find the event driven architecture acceptable or at a minimum leave that architectural decision to the software engineering team.
  + Rationale: The event sourcing architecture solves two of the major issues that the client requested (which were unnecessarily refused by the former system analyst for our company) and reduces the amount of data that must be transmitted for update operations.

### Limitations

* The system is not voice activated
  + Rationale: The diverse array of hardware that the system may be running on, the complexity of the operations that users must execute, and the level of effort required to implement a voice activation system are all factors in the decision to limit the system from being voice activated.
* The number of session reservations is limited to the number of instructors available during that time slot.
  + Rationale: The client only has 10 vehicles, so creating more session reservations than the instructors can handle will only cause the student to be dissatisfied.
* The system will only synchronize data when it is connected to the internet, therefore there is the capacity for data in an offline state to become stale. It is possible that there will be some merge conflicts that appear too complex for the user to sort out. This applies only to administrator users and employees, as student users will be restricted to making changes in the online status to prevent a conflict from happening with a change they made.
  + Rationale: Offline data becoming state is a fact of life. The best way to mitigate the issue is to make sure the device with the offline data is synchronized periodically by connecting to the internet.
* The timeline provided in schedule planning is utterly unrealistic for the work in the scope.
  + Rationale: The timeline was generated in a vacuum, and they clearly did not think through all the components that would need to be created to make the system work. The timeline was so bad, that the addition of the offline access and student self-serve account creation don’t even make a material difference. Promising the below timeline to a client is an awful idea that will only end in frustration and disappointment. In addition, the high-level tasks indicate that all the actual coding is to be done in six weeks! That leaves zero room for error on the estimates, and it’s clear that there is a great deal of uncertainty in the requirements.
* The budget for this project is too low.
  + Rationale: The client has not been informed about the myriad third-party systems they must subscribe to, not the least of which is Microsoft Azure or equivalent, so if they are expecting to pay the standard rate for six weeks of development and get the product they asked for, they will be disappointed. This project will cost us an order of magnitude more than we are presumably charging the client for it based on the schedule planning, so it will be a losing proposition.

### Gantt Chart

A picture containing timeline

Description automatically generated

Full Gantt chart available in the included CS-255 DriverPass Gantt Chart - Holden.xlsx

## References

Dahn, M. (n.d.). A guide to PCI compliance. <https://stripe.com/guides/pci-compliance>

Fowler, M. (2017, February 7). What do you mean by "event-driven"? martinfowler.com. <https://martinfowler.com/articles/201701-event-driven.html>

Mozilla. (2022, January 21). The web and web standards - learn web development: MDN. Learn web development | MDN. https://developer.mozilla.org/en-US/docs/Learn/Getting\_started\_with\_the\_web/The\_web\_and\_web\_standards

Thomson Reuters. (n.d.). Data Privacy principles all legal providers should adopt. Data Privacy Principles All Legal Providers Should Adopt | Thomson Reuters. <https://legal.thomsonreuters.com/en/insights/articles/data-privacy-principles>

New York DMV. (2021, June 18). New York State driver's manual & practice tests. New York DMV. https://dmv.ny.gov/driver-license/drivers-manual-practice-tests