Bradley Jackson

03/07/2021

CS255

CS 255 Business Requirements Document Template

Contents

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 is the client.
- They want to provide students with access to online practice exams and on-the-road training to better prepare them for driving tests.
- DriverPass wants help in designing a system that can handle these needs.

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 wants the system to record user data, provide logging and RBAC for user data, be able to update user information as well as package information, and provide an interface via a web application. The problem that DriverPass wants to solve is: many people fail their driving tests at the DMV. We will need the following components:

- Web UI
  - stateless web service
  - serves frontend of application
  - admin portal
    - should provide a means to pull reports from DB services and export as CSV/JSON
- Database Service(s)

        - stateless web service
        - RESTful
        - updates database with predetermined queries based on user credentials and action
    - Database
        - user schema
            - userid, first name, last name, password, address, phone number, state, credit card number, cc expiration date, cc security code, role (user, admin)
        - package schema
            - userid, package, current date
        - appointment schema
            - userid, date, address, drop-off location, pickup location
        - test schema
            - userid, test name, time taken, score, status (not taken, in progress, failed, or passed)
        - car schema
            - id, checked out, driver, userid
        - driver schema
            - id, userid, first name, last name

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?

- DriverPass needs to be able to download the reports and some information in CSV/JSON format.
- Role-based access controls for administrators.
- Users must be able to make appointments, cancel, and modify appointments online.
- Logging to see when users make reservations, cancellations, and modifications.
- Users need to be able to make reservations for driving lessons which are two hours long.
- Users should be able to make this reservation online using their account, via phone, or a visit to the DriverPass office to schedule an appointment with our secretary.
- Need to be able to track which driver the customer is scheduled to go out with and record the time and car used as well.
- There are a maximum of ten cars and drivers.
- Three packages available for driving appointments
    - Package One: Six hours in a car with a trainer
    - Package Two: Eight hours in a car with a trainer and an in-person lesson where we explain the DMV rules and policies.
    - Package Three: Twelve hours in a car with a trainer, an in-person lesson where we explain the DMV rules and policies—plus access to our online class with all the content and material.
- The online class should also include practice tests.
- Each driving session is two hours long and are broken down into separate meetings.

- DriverPass should be able to disable a package.
- Enhancement: DriverPass needs to be able to add, remove, and modify packages (may need this for compliance).
- Data fields gathered upon registration: first name, last name, address, phone number, state, and their credit card number, expiration date, and security code.
    - It should also ask them for a drop-off location, which should be the same as the pickup location.
- Customers should be able to reset their password.
- Cloud architecture with web UI/UX
    - The online test progress should show the tests the customer took. It should show what's in progress and the ones that the customer completed. So, it would say something like test name, time taken, score, and status. The status could be not taken, in progress, failed, or passed.
    - In the driver notes, I need to show any comments the driver left as well as the times for the lessons.
    - There should be an input form where the student (or secretary) fills in the student information, such as first name, last name, address, et cetera.
    - There also should be a page for contacting us, and a way to contact the student.

Requirements

Nonfunctional Requirements

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 system needs to be hosted within a cloud environment and serve browser clients via HTTP. We should aim for less than 100ms response time when clients' browsers are making requests/loading pages. With this flavor of architecture, we should be able to make updates without any outage so they could be planned around any schedule.

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 most cost effective method of hosting the web services would be using a lightweight flavor of linux such as Alpine; however, Ubuntu would also work. We would also require a database which could be a cloud-native service or we could provision a VM to host a Windows SQL server (this may be more expensive).

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?

Any time that we have a free form text field, we will need to sanitize the users' inputs. For instance, if the user tries to provide a userid which does not conform to our userid standards, we should throw a visible error message within their browser so they may correct it. Because we will be forming the service calls between the UI service and REST services, inputs should already be limited enough to prevent many errors. Despite this fact, we should take advantage of console logging and log alerts for our cloud platform. In this way, we could send an email to admins when a critical error occurred.

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?

We can easily create a function to add/remove/modify users without requiring an update to the entire system. Administrators would be able to access an admin-only section of the website's UI where they could make changes to user data via a form. The form would use REST calls to interact with the REST service which, in turn, would perform inserts, updates, and deletions on the database.

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?

In order for the user to log in, they would be required to register. During this registration process, we could have them validate their email address and add their current browser as a trusted user. Using cookies, we could verify if the user is accessing our application from a new browser and send a verification code to their email address. After they have successfully verified their identity, they could again trust their browser. Due to the two-factor authentication (2FA) method described above a brute force attack from a browser outside of the user's trusted systems would be ineffective. We can also consider other 2FA methods; however, the same browser trust system will be implemented so it is a matter of preference. If the user forgets their password, we can send a reset link to their verified email address. Apart from that, we could use recovery questions to verify the user's identity before allowing them to change their password.

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 collect required data if the user is registering.
- The system shall require a newly registered user to set up some form of two-factor authentication.
- The system shall validate user credentials and 2FA/cookies when logging in.
- The system shall route the home web UI.
- The system shall allow the user to make appointments, cancel, modify appointments, and take practice exams online.
- The system shall be able to keep track of drivers and their schedules.
- The system shall allow drivers to view their scheduled appointments.
- The system shall allow administrators to view and edit user data.
- The system shall allow administrators to edit packages available to users.
- The system shall allow administrators to edit drivers and their schedules.

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 service for the frontend will need to be dynamic, detecting and updating views based upon updates to the database. The different users for this interface will be customers, drivers, and administrators. Each user will be able to interact with the frontend from any platform or browser. It is expected that the service will use a framework which conforms to mobile standards for viewing websites.

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?

- By making our application design harness universal protocols, the only limitation for our users is the requirement of a browser. It is important to note that, with the architecture described above, the specifications of the browser are also irrelevant. The user should not be limited by their technology and if they happen to be using an EarthLink browser from 1997, they will most likely be calling us directly (and our support team will address their needs).

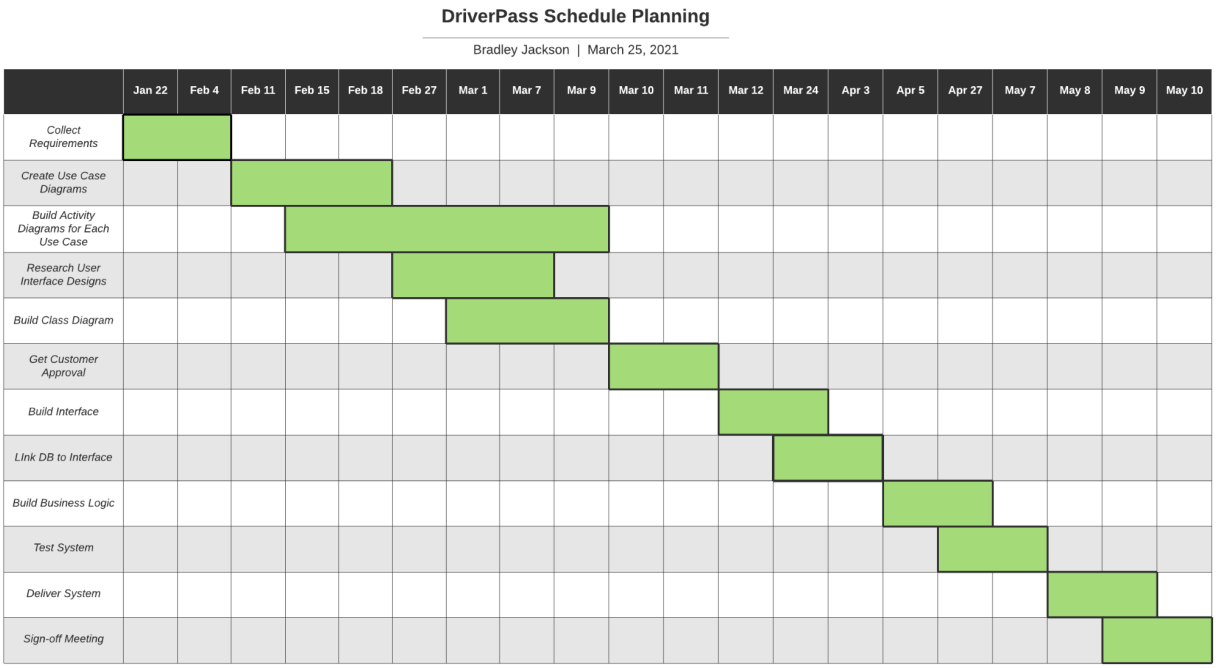
Limitations

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?

- Realistically, our main limitations will be time and budget. Harnessing frameworks (Spring Boot Java for the REST backend, Node JS, React Native, and bootstrap for the frontend) it is easy to overcome our time constraints; however, in order to hire engineers who know how to use this technology to complete our goals efficiently and in a timely manner will be costly. The database will also be the largest part of the application cost and further research will be required to determine the cost of this project.

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.



References

Dennis, A., Tegarden, D. P., & Wixom, B. H. (2012). Systems Analysis and Design with UML, 4th Edition. John Wiley & Sons. Retrieved April 1, 2021, from https://learning.oreilly.com/library/view/systems-analysis-and/9781118037423/