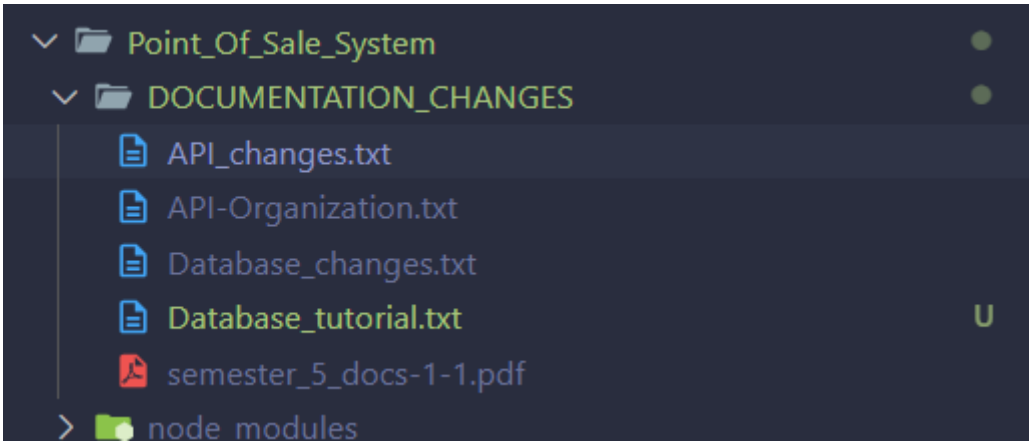
Dizaino įvertinimas

Bedarant darbą turėjome tikrai nemažai pokalbių bei diskusijų ar komandos “POS” dokumente pateikti sprendimai tikrai yra tinkami. Tad teko keletą dalykų pakeisti.

Nemažai pakeitimų taip pat yra dokumentuota mūsų repozitorijoje



Pagrindiniai “Frontend” pokyčiai

Dokumento siūlomi priekio sprendimai buvo tikrai įdomūs bei unikalūs, tačiau ne visai išdirbti ar netgi kai kas praleista. Štai keletas dalykų:

“Užsakovų” pateiktame dokumente menu navigavimas buvo padarytas puslapio kairėje. Tai nesijaučia intuityviai bei prideda sunkumų su turinio išdėstymu bei atvaizdavimu, tad mes pakeitėme į klasikinį sprendimą (Navbar’as svetainės viršuje)

Taip pat dokumente nebuvo užsiminta kaip tiksliai atrodo skirtingų rolių atliekami procesai, tad teko improvizuoti ir pridėti taip, kaip mes suprantame. keletą pvz.:

“Super admin” rolei pridėtas atskiras puslapis, kur galima pasirinkti bet kurią organizaciją ir, pasirinkus, žiūrėti tos organizacijos “owner” lygio puslapius

Nebuvo aprašyta kaip tiksliai reikėtų dirbti su rolėmis, tad sugalvojome jog darysime jas hierarchines: “Super admin” -> “Owner” -> “Manager” -> “Employee”

Pagrindiniai “Backend” pokyčiai

Duomenų bazė buvo stipriai pakeista nes nemažai sprendimų pasirodė nelogiški. Štai keletą mūsų įžvalgų:

All id's are strings.
Basically there are no foreign keys anywhere in the database diagram.
Emplee doesnt have any data only password and username.
There isn't a table which has flags held (Would need to hardcode it).
Emplee table doesnt have an owner id, so it would be impossible to know which emploees are the owners.
Emplees don't have a schedule id that would link them to a schedule.
Service doesn't have an end time just a start or even a duration.
Service doesn't have a price.
There is no point in having a service table because the appointment and MenuService table take care of everything.
There is no flag or something like that to show that a payment was split so it would be inpossible to trace back and refund normally.
Payment doesnt have a saved tax, so it would be only possible to try to count it out(but there are different taxes so it would be kinda impossible)
Discount isn't connected to order (they included a status for it but it's just a mistake)
Order and OrderItem doesn't have a saved item, only a foreign key to a menu item, so if a buisness deletes the item, the order would be empty.

(Ir aišku atitinkamai tai buvo išspręsta ir pakeista. pvz ID's pakeista kad būtų išsaugota skaičiaus pavidalu ir t.t.)

Kitos problemos

Taip pat iškilo problemų analizuojant API dokumentą bei šiaip logiški neatitikimai. Keletas įžvalgų:

API Contracts:
CRUDs don't have delete endpoints.
CRUDs don't have a way to get all of a specific businesses' data tables like Menu items, employees and so on.
No way to edit an order.

Main problems:

1. Employee cannot modify an open order. The order is considered non-modifyable after the "confirmation". So, an order even if it is not yet paid for, cannot be changed.
2. Similar problem, the Employee cannot cancel an order even though it is not yet paid for. This also transfers to appointments, as they are not cancelable after being confirmed.

(both problem 1 and 2 are visible in seqence diagrams figure 10 and figure 11.)

3. Tax management, even though tax is described in the database class diagram, it is not described in any way in workflows/wireframes.
4. Each service does NOT have an employee associated with it, an employee is not assigned, not during an appointment or order, this feature is described in neither the workflows/wireframes nor database class model.

Weird problems:

1. Discounts: when applied to an order or appointment is apparently supposed to be a special code that gives a discount, instead of literally just writing a discount (the application is not mentioned in any diagram, but is described in paragraph 2.4.1)
2. Split payments are possible and described properly in a sequence diagram, yet for some reason there exists an option to split it into a number of payments as shown in figure 12. Employee being able to choose the amount of splits is not described enough or is simply a non-needed function, because it is already possible to change the payment amount.

Visos šios problemos buvo išspręstos diskutuojant bei kritiškai klausinėjant “kas būtų geriau nei tai?”

Apibendrinimas

Nors ir buvo tikrai nemažai problemų su techniniu dokumentu, jis tikrai nebuvo tragiškas. Taip pat “užsakovai” buvo komunikabilūs, tad dėl keletos ne aiškumų tiesiogiai parašėme ir pasiklausėme