Group 8 Project ITR2

11/03/2024

Jiaye Chen, Joshua Zuker, Tejae Whitley, Kyle Williamson, Cyrus Yang, and Julianna Silva

LE/EECS 2311 X | Software Development Project
York University

Table of Contents

Our Ideas

Time Consuming but Unique:

- ~~Game~~
    - ~~2D top down game (Zelda, Metal Gear)~~

Simple, but Effective:

- ~~Music player~~
    - ~~Play music~~
- ~~Recipe Management System~~
    - ~~Input recipes, recipe info, categorise recipes, …~~
- ~~Soundboard~~

Balanced:

- Claims Database
    - Input claim number, get information on claim, and edit claim information
- ~~Dental Patient Record~~
    - ~~Contains different patients, the condition of their teeth, their insurance status~~
- ~~Movie Review (Database?)~~
    - ~~Search Movies, Leave Reviews,~~
- ~~Personal Finance App~~
    - ~~Takes real financial info, UI interface, Shows visual data~~

Todo List

- [ ] Interview video 2
- [x] ~~Make up 2 or 3 more client stories + 2 or 3 more insurer stories (small)~~
- [x] ~~Rewrite the Big User Stories~~
- [x] ~~Assign stories~~
- [ ] Finish 10 or 12 small stories implement by 3.11

- [x] ~~Update this Pdf~~
- [x] ~~Update the log file~~
- [x] ~~Put all files and codes on github by 3.11~~
- [x] ~~Peer evaluation for it2~~
- [ ] Prepare slides with itr3 for the final demo

Database List:

CustomerDB - Jaye

AdvisorDB - Josh, Cyrus

VehicleDB - Julianna (For Now)

ClaimDB - Kyle

AdminDB - Cyrus (It-3)

PolicyDB - Tejae

Vision Statement

The claims database is used to hold the personal information of an insured individual, as well as the information regarding their claim. This can include whether the vehicle has been totaled, who was at fault for the accident, and the payment information. The database will also have a toolbar which contains a variety of utilities an advisor can use. This project will be done from scratch.

This database also contains a 'notes' section of the toolbar that will allow the claims advisor to add or remove information pertaining to the claim and the client. In order to access a particular claim, the advisor must use a corresponding claim number.

If a client requires a claim be opened, the advisor can open a new claim by going to the 'open claim' section of the toolbar. This will allow them to input information regarding the claim that they have at the moment, such as the make and model of the client's vehicle, the time of the accident, etc. As time goes on this information can be updated.

When the client has been satisfied, and has been paid, the advisor may close the claim. This can be done by going to the 'close claim' section of the toolbar. The advisor must note the conditions in which the claim closed, such as how the client felt, and will be able to close the claim afterwards.

This database will be an improvement to the current system, as it will be more simplified compared to the one in use. It will also include a more modern UI. This system will be successful if users of the older system prefer it to the current version.

Our customer will be a claims advisor who specialises in bodily injury claims.

User Stories

- [ ] ## Big User Stories
- [ ] **User 1(Customer): As a customer, I want a comprehensive and user-friendly online claims processing system that allows me to quickly and conveniently file a claim at any time without the need to visit an office or speak with an agent. This system should support the upload of pictures or video footage of my damaged vehicle, providing me with the option to submit additional evidence for my claim.**

  Additionally, I would like the ability to track the progress of my claim, allowing me to monitor each step of the process in real-time through a clear and modern dashboard upon logging in. The login experience and account creation process should be modern, easy, and straightforward, ensuring that I can sign up for an account without unnecessary complications or excessive information requirements.

  Furthermore, the system should enable me to easily edit my account information, such as my phone number and email, to ensure accuracy. In case I decide not to use the service anymore, I would like the option to delete my account hassle-free. Overall, the goal is to provide a seamless, efficient, and user-centric online claims experience.

- [ ] **User 2(Insurer): As an insurer, I need an intuitively designed dashboard within autoclaims that consolidates all submitted claims, allowing me to**

efficiently organize my workload. This dashboard should provide access to related photos, videos, and documents, offering additional context for accurate claim assessment. Furthermore, I want the ability to update claim statuses, facilitating timely notifications to customers.

To begin claims adjustments, I seek a straightforward account signup process within autoclaims. Additionally, I desire access to past claims history and records for future reference, ensuring a comprehensive overview of the insurer's interaction with claims over time.

- [ ] **User 3(Administrator): As an executive of an insurance company, in order to track customer interactions, preferences, and feedback across each interaction and improve customer loyalty and retention, I want a customer relationship rating system that is integrated with our claims database.**
  As an executive of an insurance company, in order for customers to understand our policy price calculation I want the system to list some factors that can affect the price that the customer is paying.

Highlighted = Done but need database

User Stories

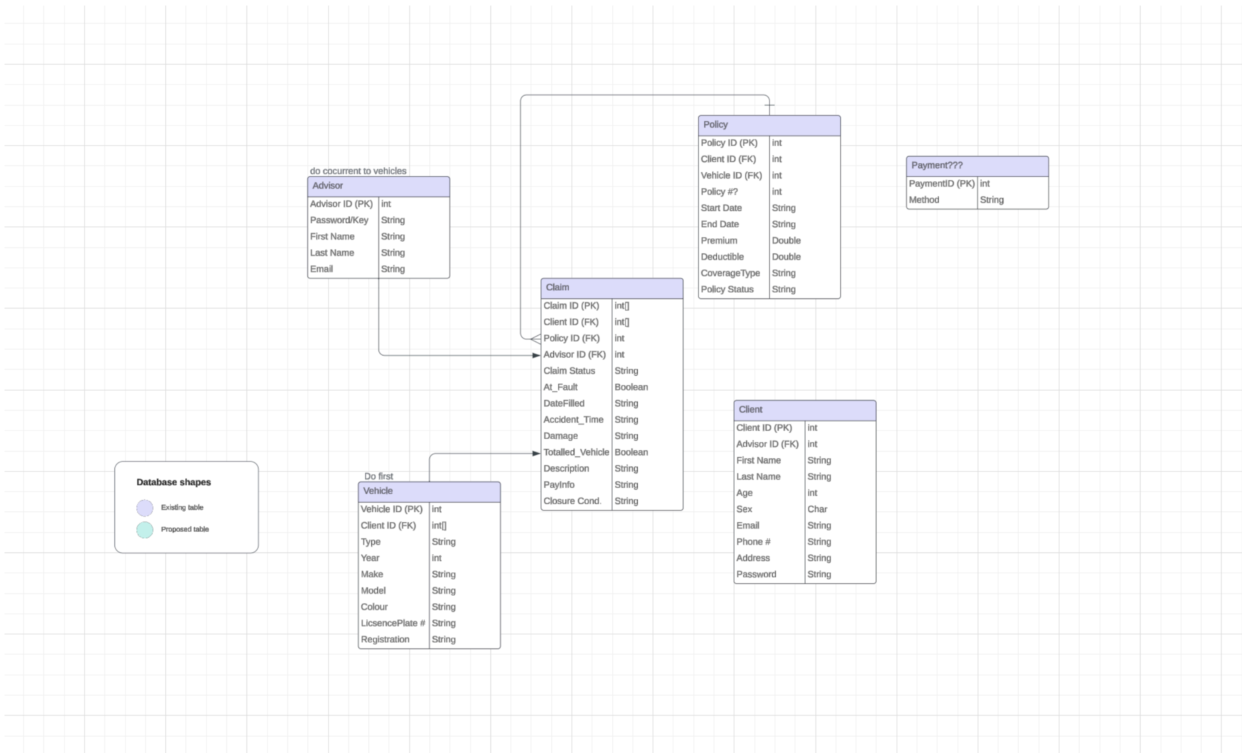
- [x] ~~As a customer, in order to quickly file a claim at any time without having to visit an office or speak with an agent, I would like to do the whole process online. (**Cyrus)**~~
- [ ] As a customer, I would like to be able to track the progress of my claim so I can see what step of the process my claim is at. **(Julianna)**
- [x] ~~As a customer, I would like to have a modern and easy login experience. **(Klye)**~~
- [x] ~~As a customer, I would like to sign up for an account easily, to avoid too much complicated and useless information. (**Tejae**)~~
- [x] ~~As a customer, I expect a clear and modern dashboard that provides a real-time overview of my claim when I'm logged in. (**Tejae**)~~
- [x] ~~As a customer, I would like to edit my account information, like phone number, and email. **(Jaye)**~~

- [x] ~~As a customer, I'd like to delete the account when I'm not using it. **(Jaye)**~~
- [x] ~~As an insurer, I want to be able to access all the submitted claims through an intuitively designed dashboard, so I am able to organize my workload well. (**Joshua**)~~
- [ ] As an insurer, I would like to be able to view all the related photos, videos, and documents, so that I can have more context to assess the claim.**(Kyle)**
- [x] ~~As the insurer, I want to be able to update the status of the claim, so that I can notify the customer. (**Joshua)**~~
- [x] ~~As an insurer, I would like to sign up for an account in autoclaims to start doing claims adjustments. (**Cyrus)**~~
- [ ] As an insurer, I'd like to access past claims history and records for future references. **(Julianna)**

(keep admin stories for itr3)

- [ ] As an administrator, I want to control user accounts and permissions to make sure staff members have access to the right resources and information for their jobs.

The Planning Map



☐ Database schema

Clients	
client_ID (primary key)	
advisor_ID (foreign key)	// does every client have an assigned advisor?
vehicle_ID (foreign key)	// not sure about this one bc maybe clients can have multiple vehicles // Perhaps we can allow clients to add more vehicles
first_name	
last_name	

phone_number	
email	
address	

Vehicles	
vehicle_ID (primary key)	
client_ID (foreign key)	
vehicle_type	// what type of vehicles? // Refers to shape of car e.g sedan
vehicle_make	// Car manufacturers e.g Toyota
vehicle_model	// Specific car model from manufacturer e.g Corolla
vehicle_colour	

Advisors	
advisor_ID (primary key)	
first_name	
last_name	

Claims	
claim_ID (primary key)	
client_ID (foreign key)	

advisor_ID (foreign key)	
claim_status	// What kind of status conditions?
at_fault	//true, false, 5050
accident_time	// date and time
totaled_vehicle	// true or false
payment_info	// should clients have payment info?
closure_feedback	

Notes	
note_ID (primary key)	
claim_ID (foreign key)	
advisor_ID (foreign key)	
note_content	
note_timestamp	

☐ Java Classes

Claim (who files the claim? Is it the advisor after the client requests? Or is it the client alone?)

-

ClaimDAO

- Handles connection between Claim and the database
- Claims can be created, edited or closed through this

Who registers a new client? Is it the advisor?

Client

- Add a counter for when a client is registered the number is incremented and assigned as clientID (if it is random then there is a chance that the IDs will be duplicate)
- firstName, lastName, phoneNumber, email…
- Getters and setters for each one
- One advisor object from Advisor

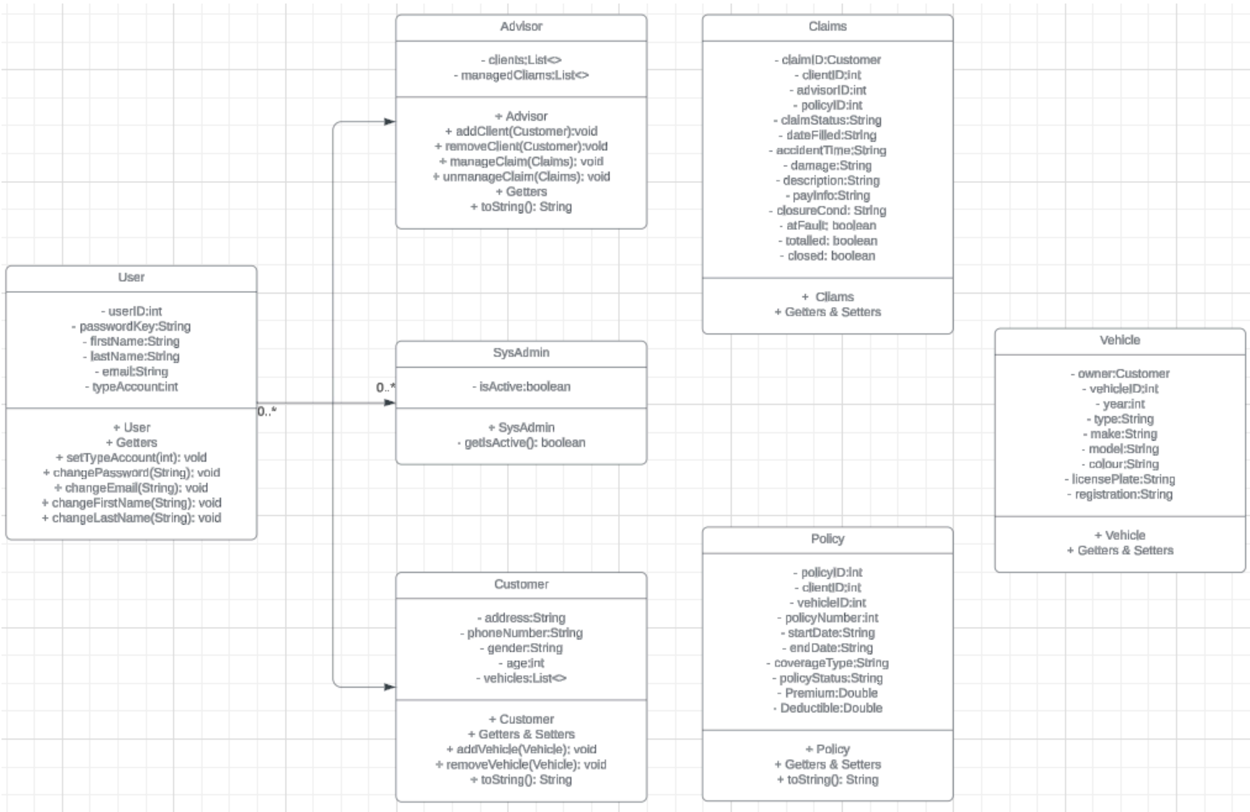
ClientDAO
- Handles connection between Client and the database
- Registers clients, returns specific client by ID, etc

Vehicle
- Type, make, model, year etc
- Has an owner of type Client

VehicleDAO
- Handles connection between Vehicle and database
- It should be able to search an specific vehicle through its owner ID (clientID)

Advisor
-

Note (only an advisor can create a note)
  - Note is attached to an object of type Claim


Advisor
- clients:List<>
- managedCliams:List<>
+ Advisor
+ addClient(Customer):void
+ removeClient(Customer):void
+ manageClaim(Claims): void
+ unmanageClaim(Claims): void
+ Getters
+ toString(): String
Claims
- claimID:Customer
- clientID:int
- advisorID:int
- policyID:int
- claimStatus:String
- dateFilled:String
- accidentTime:String
- damage:String
- description:String
- payInfo:String
- closureCond: String
- atFault: boolean
- totalled: boolean
- closed: boolean
+ Cliams
+ Getters & Setters
User
- userID:int
- passwordKey:String
- firstName:String
- lastName:String
- email:String
- typeAccount:int
+ User
+ Getters
+ setTypeAccount(int): void
+ changePassword(String): void
+ changeEmail(String): void
+ changeFirstName(String): void
+ changeLastName(String): void
0..*
0..*
SysAdmin
- isActive:boolean
+ SysAdmin
- getIsActive(): boolean
Vehicle
- owner:Customer
- vehicleID:int
- year:int
- type:String
- make:String
- model:String
- colour:String
- licensePlate:String
- registration:String
+ Vehicle
+ Getters & Setters
Customer
- address:String
- phoneNumber:String
- gender:String
- age:int
- vehicles:List<>
+ Customer
+ Getters & Setters
+ addVehicle(Vehicle): void
+ removeVehicle(Vehicle): void
+ toString(): String
Policy
- policyID:int
- clientID:int
- vehicleID:int
- policyNumber:int
- startDate:String
- endDate:String
- coverageType:String
- policyStatus:String
- Premium:Double
- Deductible:Double
+ Policy
+ Getters & Setters
+ toString(): String