Final Report

# Use Case Design

* When open the index webpage, a user has two options, one is login and the other is register.
* If the user clicks the register link, then he/she would be directed to the register page, where the user needs to enter some personal information, including username, password, first name, last name, date of birth, gender, email, ssn, number of children, smoke or not, region and bmi. Some of the information is for account and some of the information is for prediction of their annual health expenditure. When the user submits, he/she would be redirected to the login page. (Usually there should be a confirm email step but in this case, I ignore that.)
* If the user clicks the login link, then he/she would be directed to the login page, where there should be a form to enter username and password. If the user clicks the login link by mistake, there is a link directed to the register page. If the username and password don’t correspond to the information in the database, then this page will refresh and show the alert. If the username and password are correct, then the user will be redirected to the home page.
* Home page should have the following links:
  + Check the predicted annual expenditure
  + Check and add billing account(s)
  + Check and add contract(s)
  + Logout
* Predicted annual expenditure page will utilize the trained model to predict the expenditure based on the information the user provided and show the number.
* Check and add billing account(s) page will show the billing account(s) already added by the user and a form that the user can submit to add another billing account. When a new billing account is added, the page will refresh and show the newly added account as well as previous account(s).
* Check and add contract(s) page will show the contracts that the user already added and a form that the user can submit to add a new contract. When a new contract is added, the page will refresh and show the newly added contract as well as the previous contract(s) if any.
* Logout link will log the user out.

Due to limited time and knowledge of how to settle the contract and determine the premium amount, I skip many steps. For example, instead of building up a contract for a customer and then add it to the system, I assume that the company has already come up with a contract for this customer and the contract number and other information already exist, and thus directly add the contract information to the system. And for the customer, when he/she comes to the company, there should be many different types of insurance product. I assume there exists only one type, which is the health insurance and based on their information, I use the “simple” machine learning model which was trained in the part 3 to predict their annual health expenditure and here I simple mix all things up.

# Use Case Notation

Diagram

Description automatically generated

# Application Document

This Webapp is based on Python / Flask / HTML / Sklearn / ONNX / Microsoft Azure / MySQL Server frameworks. Flask is used to develop the backend architecture. HTML is used to develop the frontend architecture. Sklearn is used to train the model on dataset. ONNX is used to save the model and do the inference. Microsoft Azure and MySQL server serves as the database system platform for data storage and organize.

Database design is from part 2 but new tables are added to meet the use cases. Such as a table relation of account member and billing account. In the previous part, billing accounts are related to accounts, but accounts are belonging to companies. Personal account members should also have billing accounts. And some other slight changes are made to simplify queries such as change SSN to username in some tables.

Queries are not complicated due to simple use cases and modified physical database structure.

GitHub link:

# Summary

Database structure is well organized for these use cases and most tables are reduced to 2NF for optimization and faster query operation. Users’ information is well protected, and password is processed with hash operation to prevent leakage. This application forms a understanding of most requirements and end-users business use cases for small start-up companies. Due to lack of knowledge of real insurance company business cases for company administrators and associates, this project doesn’t support administrator end and associates end.