CSC 431

Taxation Calculator

System Architecture Specification (SAS)

<Team number>

Shusen Han	Lead Designer
Ziheng Tang	Planning Lead
Jinrui Dong	Programmer

Version History

Version	Date	Author(s)	Change Comments
1	2022/4/20	Shusen Han, Jinrui Dong, Ziheng Tang	All Requirement

Table of Contents

Table of Tables

None

Table of Figures

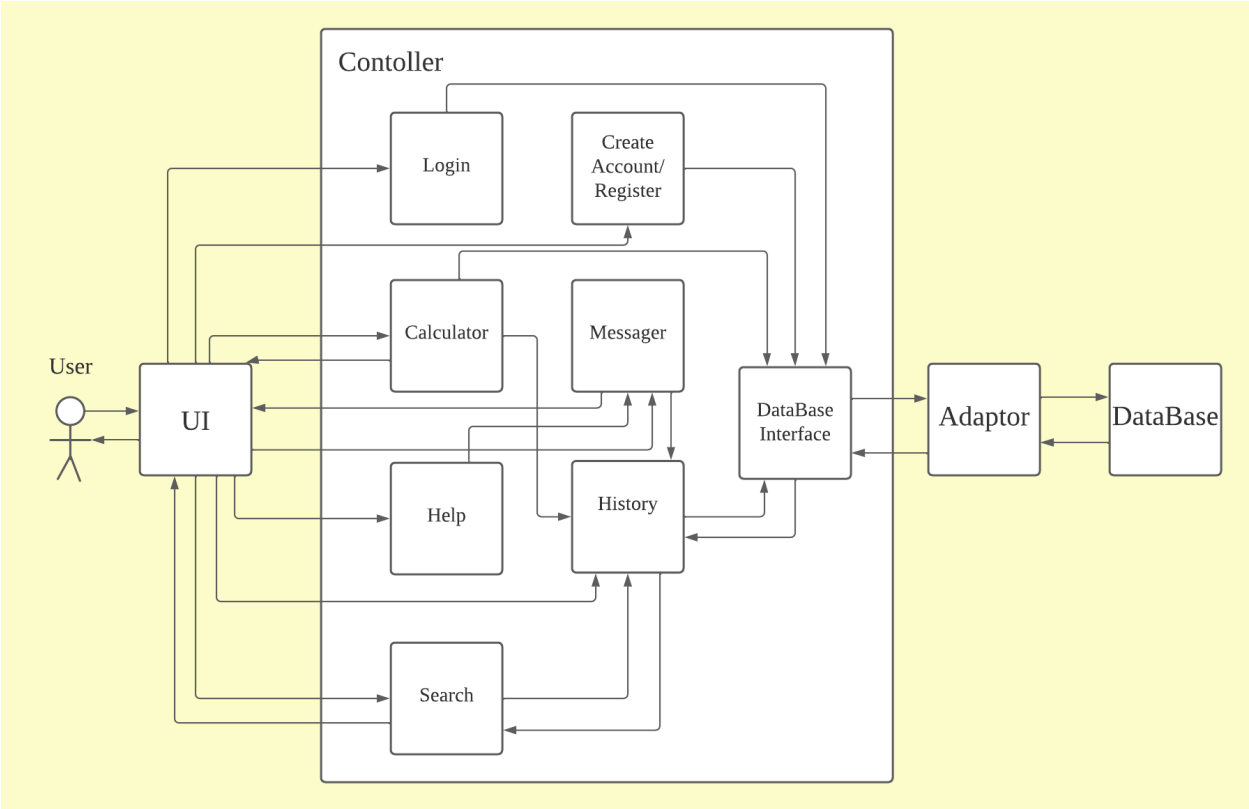
1. System Analysis

1.1 System Overview

This document describes the architecture design and specifications of the Tax Calculator Application. This system is Pipes-and-filters architecture, and it includes three main parts: the UI, the Controller, and the Database. The UI makes it easy for the user to operate the application, which is the medium between the user and code. The Controller is designed for invoking various functions and methods. The Database can store user operations' information and data, such as login history, account information and so on.

When the user uses the application, the UI works first, which allows the user to do some operations and show the information users need, such as searching. Then these operations will create inputs to the Controller. Then the Controller invokes different functions or methods, such as matching the login information to the account information stored in the DataBase. The last part is the Database, all information is stored in it, and most operations need to invoke the data in the Database.

1.2 System Diagram



1.3 Actor Identification

- Create a new account
- Login
- Input personal information when use calculator
- Search state or city tax rate
- History of calculator and messager
- Ask for help
- Send Message

1.4 Design Rationale

1.4.1 Architectural Style

The application will utilize Pipe-and-filters architecture: UI - Controller - Database

- UI:We will use Qt to create the UI.
- Controller: This part includes most logic and functional parts.
- DataBase: The database is used for storing all information, and we will use third-party services, such as AWS.

1.4.2 Design Pattern(s)

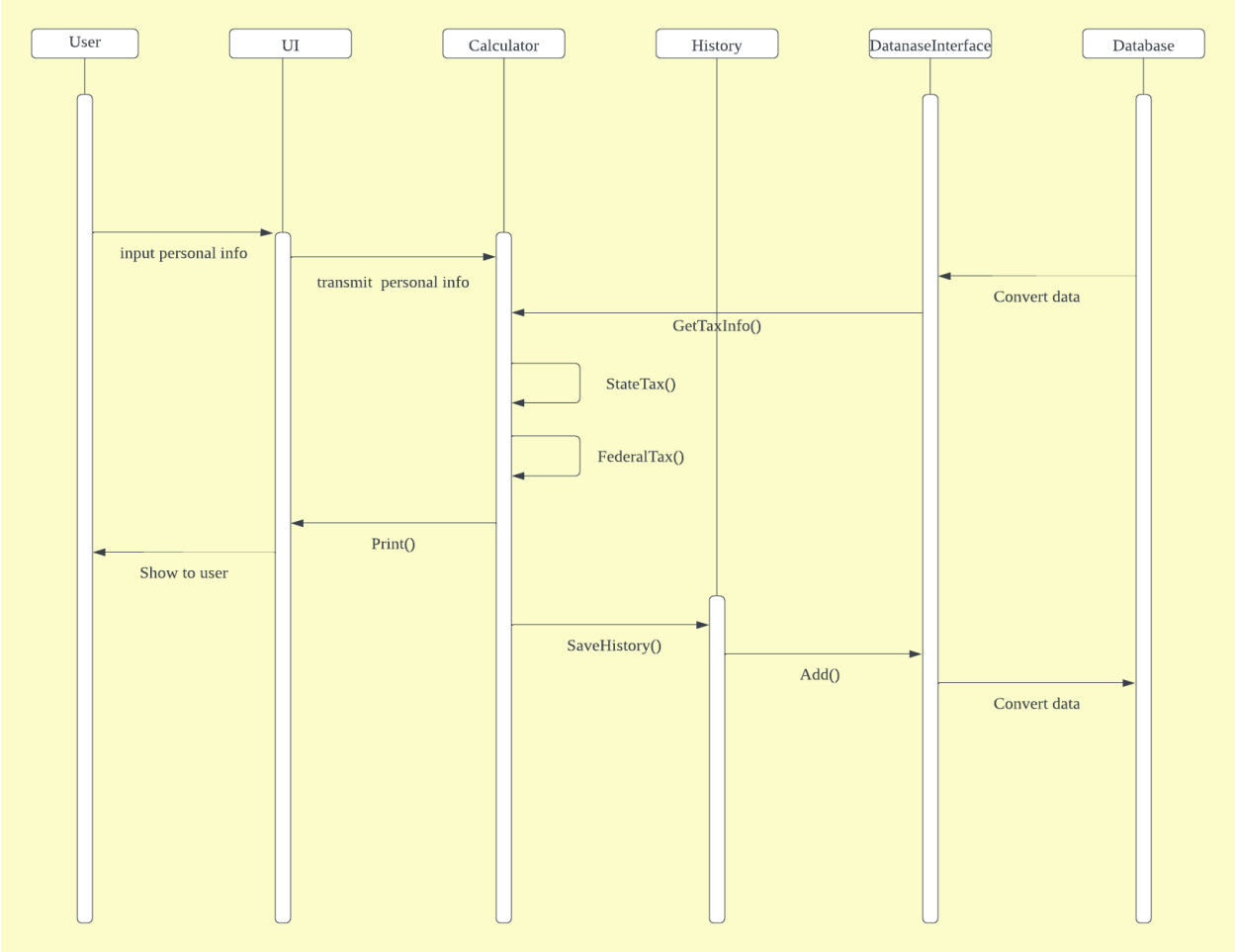
We think that the Adaptor Design pattern will be used in our application. Adaptor classes will be used to communicate between the Controller and the Database. Data created by the user, such as message history, will be converted to the suitable format by the adapter class. Correspondly, the data created by the user will be converted to suitable format and stored in the database.

1.4.3 Framework

The calculator will implement UI by Qt which is suitable for User Interface and Development. The controller is written in Java for its complex function and logic. Database service is provided by third-party, but it will be implemented via Python due to its advantages in data processing.

2. System Analysis

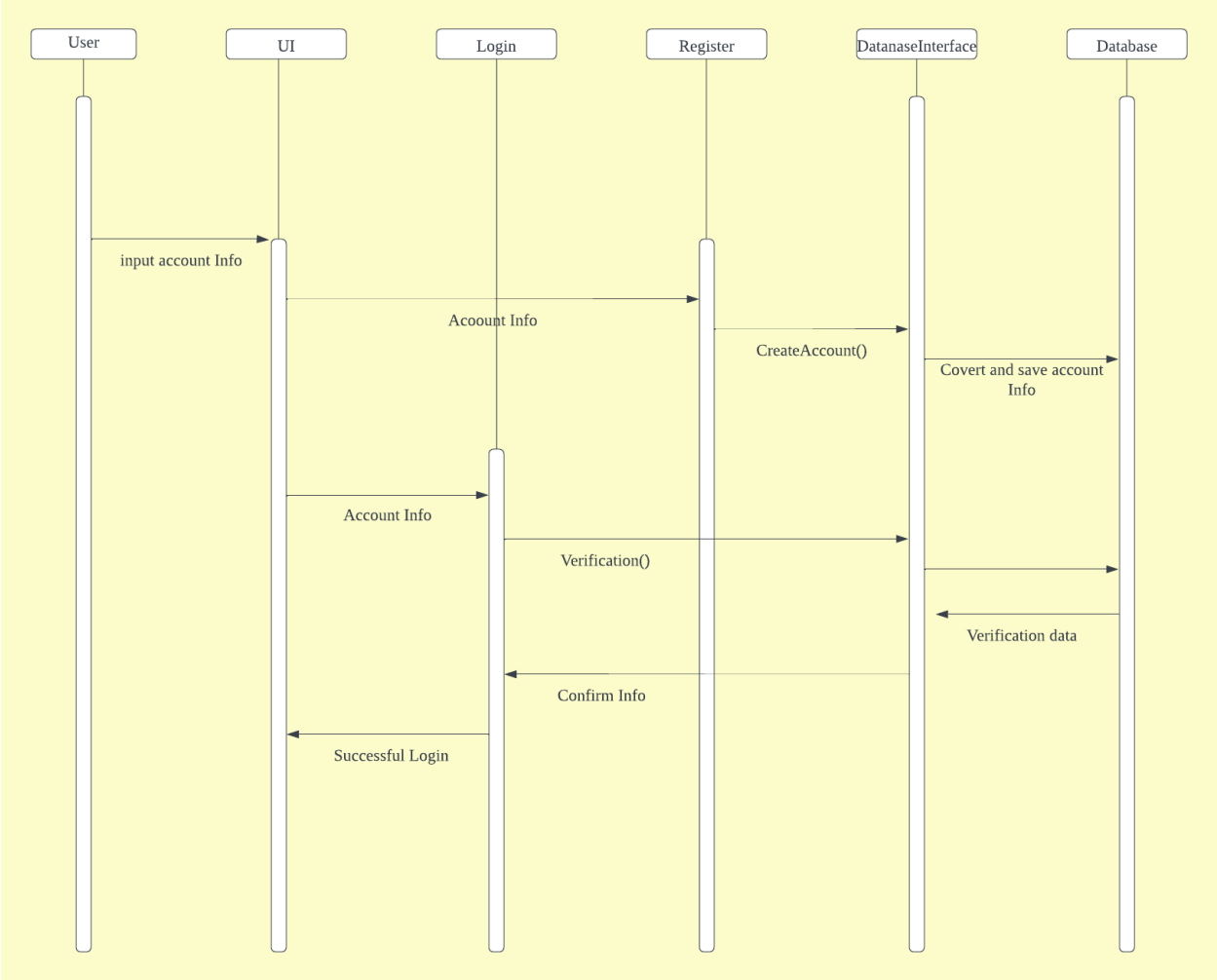
2.1 Calculator



- When the user uses the calculator, the UI receives the input from the user and sends it to the Calculator.
- The Calculator will use the GetTaxInfo() function to invoke the data about the tax form the Database through the Database Interface.
- Then, the Calculator uses the StateTax() function and FederalTax() Function to count how much the user needs to pay.
- Finally, the Calculator will use the Print() function to show the result to the user.

- After calculating, the calculating history will be stored in the Database through Database Interface.

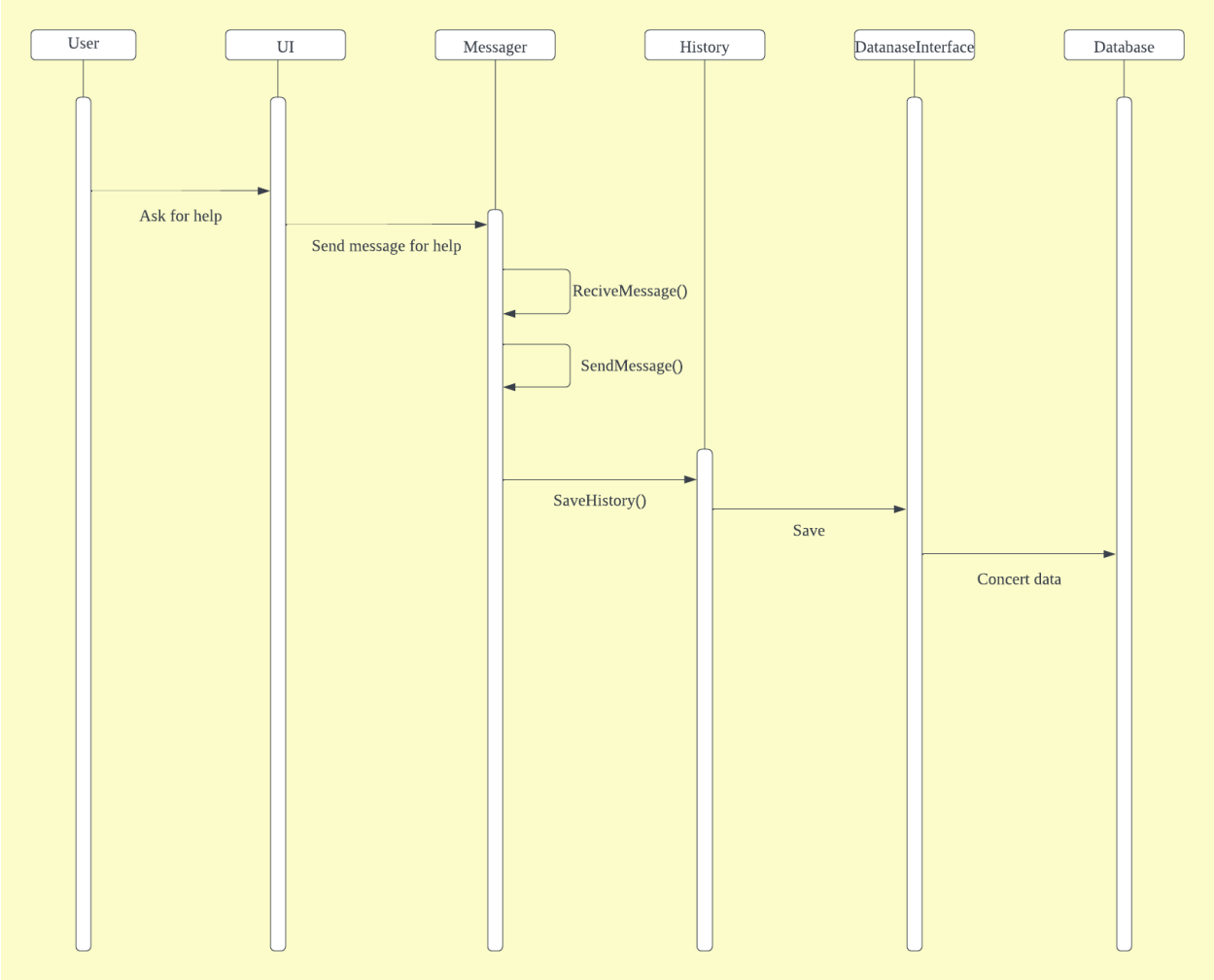
2.2 Login and Register



- When the user first uses the application, he needs to create an account. There will be a register window, and the user input the account information.
- The CreateAccount() function stores the account information in the Database through the Database Interface.
- When the user logs in, the input in the login window will be transmitted to the Login part, and the Verification() function will match the input to the data in the Database.

- If the input matches an account information, the user logs in successfully. Otherwise, failed.

2.3 Customer Help



- When the user needs help, such as technical problems, he can use the messenger to communicate with the staff. The main function of the messenger is ReceiveMessage() and SendMessage().
- The message history will be stored in the Database by the DatabaseInterface.

Structural Design