Design and Alternate Designs

A) Design

The final design has three distinct Java parts as well as an independent database. These three Java parts are the GUI (includes a login screen, menu screen, and a transfer screen), the Session Manager, and the Interest Calculator. Together, these work to create the final project. The final part of the project is a MySQL database that is hosted on Amazon Web Services. The GUI handles all of the front end work, and the Session Manager and Interest Calculator work together to form the backend of the project.

The GUI was designed to be as light and flexible as possible. There is no functionality in the GUI that can be better served by hosting it on the backend. In doing so, this creates a light, fast, and efficient user experience. The first screen that the user will see is the login screen. This contains the login portal as well as the user acceptance agreement. The user must accept the agreement and input valid login credentials to pass to the next screen, the menu. From the menu the user can use buttons to access the functionality of the project: view their balance, view their accumulated interest for the month at their current rates, transfer money, and view their transaction history. If the user clicks to transfer money, they are brought to the transfer GUI where they are allowed to transfer money between their accounts. The GUI includes a custom icon for the project so that the user knows what application they are in as well.

The Session Manager is the direct link between the application and the MySQL database. In order to properly run, it requires a MySQL driver that is included in the download of the zip file. The driver allows the Session Manager to talk to and get data from the database while staying within the Java environment. It allows the user to talk to the database via telling the GUI what to do, and the GUI then responds by activating an action listener which then tells the Session Manager what it needs. The Session Manager then gathers the relevant data from the current session (username and account information are the most likely ones) and creates a query based on this information for the database. The database is then pinged and returns the resultSet containing the results of the query. This resultSet is then parsed in the Session Manager, and the relevant information is sent back to the GUI. This partitioning of the Session Manager from other parts of the project means that it is much easier to sanitize data and prevent any sort of bad data getting through, but it also means that the program as a whole is much lighter and easier to control. The Session Manager holds code that would otherwise be highly repeated.

***Insert Interest Calculator design here***

B) Alternative Designs

The most basic alternative design choices boil down to two alternate paths: what language to write the application in and how are we going to store the data. Of course, what the group settled on was Java and a MySQL server hosted on Amazon Web Services, but there were other options. The first thing that was floated was to create a website that would host the program. That had some clear benefits. First off, most people are familiar with a website and use it for their online banking needs. Secondly, we would be able to access the project from anywhere if it was hosted successfully. But there were downsides too. None of the group had ever worked extensively in HTML, CSS, or PHP though some had a brief exposure to it. The other problem was that while we could simply hold everything in offline html files, that had the sane downsides as a Java file in that we had to ensure we were working off the same one at many times and we would have to be extremely strict in our version control. That, combined with the inexperience of the group made the decision to go with a fully Java version easy. We had also recognized that many features we wanted to do were going to be tough in the sense that we would be combining CSS, HTML, and PHP/Javascript. Java offered the ability to keep everything in the same language. This turned out to be valuable as the group was able to jump in and help anywhere that was necessary with no time to learn a new language.

The other design choice was to go with a MySQL server hosted on Amazon Web Services. This offered just as many choices as picking the language, as our needs were fairly basic: we needed everyone to access some data and we just needed to store small amounts of data. Because of this and the way the group organized their tables, it allowed us a huge amount of choice. The group was able to use either a relational database or a non-relational database. This made Cassandra, MongoDB, MySQL, and Microsoft SQL the top choices. The person in charge of the database, Conor, was most familiar with MySQL and thus it was chosen.

The next problem with the database was how it could be hosted so that everyone could access it. It was decided that it was not terrible if we did not all have access to the same database, so long as the data was similar enough. This meant that if the group could copy the database over, each instance of the program could come with their own database that was copied over from a master list. Of course, this was not ideal but certainly passable. The issue with this lied in the installation. The most important part of the project is that it works when it is being graded, and as such a quick and easy installation was important. After some research, the best design choice was hosting the database on Amazon Web Services. It met the requirements as it was small, free, and accessible from anywhere.

The last big alternate design decisions regarded the interest calculator. The interest calculator was something that was necessary as the group needed to show that the program was capable of getting interest. This was tough because interest is generally a very slow process. The group deliberated between calculating the interest over the next year, from the last year, over the next month, or from the past month. What the group eventually decided on was previewing how much interest they were going to earn based on the average of their last month’s balance. This struck a good compromise because each of the other decisions were going to mean assumptions or working backwards into creating test data that mirrored the data at the start.