Atamjeet Singh 100842821, Arzaan Irani 100826631
Satinder Singh 100742575, Corey Cougle 100745291 **tutorTonight by dDevelopers** COMP3004 Deliverable 2

Functionality Demo

User Case 1 (Student) – A Student with a name generated by the audience will create an account and log in to tutorTonight. The Student will search for a course that they would like to be tutored in. They will submit a session request and await a response. Once a response has been received, the list of upcoming sessions should appear to be updated with the newly confirmed session.

User Case 2 (Tutor) – A Tutor will receive an update to available sessions and see that a requested session has been populated. The Tutor will read the available session to decide whether the time and subject meet their preferences. The Tutor will confirm the session which sends a response to the Student indirectly. The newly confirmed session should be present in the updated list of upcoming sessions.

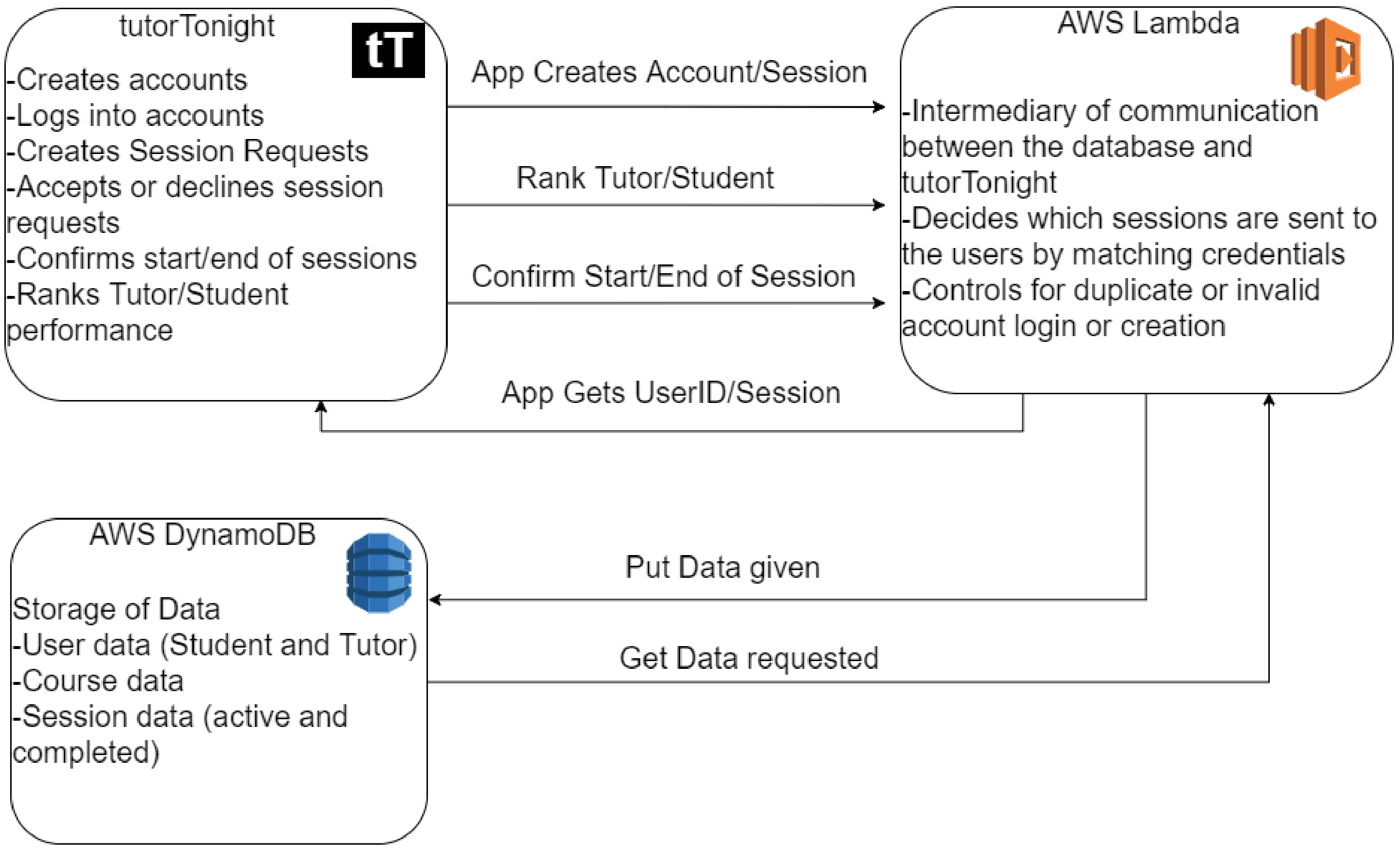
All communication from tutorTonight to the server and back are real communications. The account creation is real and will therefore modify our database contents during the demonstration. The newly created account will be used to populate data fields within other views of the application such as user name and user email. There are no simulated components within this demonstration. The session that is created by the Student will modify the sessions database and will be reflected in the Tutor's list of available session requests. The Tutor can accept the session which again modifies the session within the database and should then be reflected in both users lists of upcoming sessions on the home page.

Status Report

The user interface and database communication are mostly functional. Application functionality is almost entirely complete and performs the basic operations required for organizing sessions between a Student and Tutor. The communication with the database requires a function to implement push notifications for Tutors when there are available sessions that match the Tutor's credentials. Application functionality does not yet implement any form of payment system as it is something we need to be very careful with. We would like to use an established and secure system for this and not develop it ourselves. For this reason, we are prioritizing general application functionality. Another issue of security is user account access. Currently we are storing an unencrypted userID variable that grants a user access to a user account without need of a password. This userID variable is written to disk when a successful login is performed, and deleted from disk when a user manually logs out. We are intending to eventually use Amazon's Cognito to manage user account creation and login.

We are currently facing general user interface design issues such as object alignment, assigning values to object members, and prototype creation. Regarding database management, we are working on maintaining separate databases for active and completed sessions.

In the next month, we are going to implement a secure form of logging into the application and storing the userID so that secure password-less access is obtained. We will implement small amounts of text communication between the Tutor and Student for the purpose of finalizing a meeting location. Any remaining minor UI discrepancies throughout the application will be fixed. Once we are satisfied with the stability and performance of the application functionality, we will begin to implement a payment processing system that is known to be secure and reliable.


tutorTonight
tT
-Creates accounts
-Logs into accounts
-Creates Session Requests
-Accepts or declines session requests
-Confirms start/end of sessions
-Ranks Tutor/Student performance
App Creates Account/Session
Rank Tutor/Student
Confirm Start/End of Session
App Gets UserID/Session
AWS Lambda
-Intermediary of communication between the database and tutorTonight
-Decides which sessions are sent to the users by matching credentials
-Controls for duplicate or invalid account login or creation
AWS DynamoDB
Storage of Data
-User data (Student and Tutor)
-Course data
-Session data (active and completed)
Put Data given
Get Data requested