Acceptance Test Document

CITS3200 Group R

Project Acceptance Tests

Objectives

This document is designed to describe the conditions under which we can consider the project completed. It cannot be fully comprehensive because we anticipate the final product will contain features beyond those described here, but it describes the essential features for the sleep diary to be considered complete. Due to this the document describes the minimum final product to be delivered, with the expectation that future documentation will supersede it.

Test Summary

The system is a sleep diary app for iOS and Android devices. The app takes user input about their sleep patterns and uploads it to a SQL database where it is converted to an excel spreadsheet and sent to their clinician upon request. Further details are available in the General Scope of Work document.

Testing Strategy

For the system to be considered functional there are several subsystems that must be functional and correctly linked:

- iOS app
- Android App
- Server database
- Webservice

The individual subsystems will be tested based on function, performance and user interactability. Function can be tested through interaction with the subsystems and attempting many types of inputs. Performance will be tested by calculating how quickly the subsystem can perform tasks and how many users it can handle. User interactibility will need to be tested by external people for ease of use and verified by the client.

Subsystems to be tested

iOS/Android App Tests

*iOS and Android users have been considered the same because they require the same features.

Log In Users

Type of user trying to log in:

Input	Output expected
New user with keycode	Logs in permanently
New user without keycode	Logs in permanently, generates keycode for table and asks to share data with researchers
Previous user	Logs in automatically

Receive user input correctly

Input cases to test for:

Input	Output expected
Input is valid	Save to local database and send to server
Input is invalid	Request user re-enters data
Not all info is filled in	Request user re-enters data

Differentiate users

Must be able to service multiple simultaneous users.

Send user data to the server

Type of data sent:

Input	Output expected
Data is valid	Save to users table
Data is invalid	Log error

Save user data locally

Data is saved to a table in local storage to be used for visualization of data.

Visually display user data

Cases of data to show

Input	Output expected
Previously entered data	Display data
No data has been entered	Give a message suggesting inputting data

Server database

Receive user data

Type of data received:

Input	Output expected
Data is valid	Save to users table
Data is invalid	Log error
'Comment' section attempts a SQL injection attack	Sanitise input

Store user data in a SQL table

Type of data:

Input	Output expected
Same user tries to input twice in a day	Save most recent data
User not found in table	Create new table and save data

Convert SQL data into excel spreadsheet

Type of data:

Input	Output expected
Valid data of table	Excel file version of table
Invalid data of table	Error message and log error

Verify and send authorised data to clinicians/researchers

Type of user attempting:

Input	Output expected
Researcher	Data sent as excel file
Clinician	Choice of 'simple' or 'full' data sent as an excel file
Unauthorized user	Access denied

Webservice

The webservice must receive information from the mobile apps and parse the data then save the data to the MYSQL database. To do this it must:

Webservice availability

The web server must always be available e.g. 24 hours a day 7 days a week

Receive JSON file from Android and/or iOS device

Type of method:

Input	Output expected
Valid login code	True
Invalid login code	False
User input data	Data received

Parse JSON file at server side

All JSON data will be checked to be valid data before it is saved to the MYSQL database. The MYSQL database will be the single source of truth

Retrieve data from the MYSQL database

The webservice will retrieve from the database the necessary data given all the checks have passed e.g. valid user.

Unique code generation

The webservice will generate the unique codes by looking at the existing codes in the database and choosing a code that hasn't been generated before. A unique code can only be generated once the user requesting the code is an authorised user.