OccSpec mock task 1 B, Proposed solution.

Table of Contents

[Introduction 1](#_Toc93926030)

[Visual/interface designs 2](#_Toc93926031)

[Data requirements 3](#_Toc93926038)

[Selection of algorithm designs 4](#_Toc93926039)

[Test strategy. 7](#_Toc93926040)

[What going to be tested. 7](#_Toc93926041)

[Types of tests to be done. 7](#_Toc93926042)

# Introduction

In this document I will be outlining a digital solution to clients need. They want me to create product that can: provide information and advice to the user about fitness training and healthy living, provide access to digital content to support customers, support customer with their training and healthy lifestyle, and encourage customers to use more of their products. This should be relativity simple to create and design.

From the looks of the brief, it would best to create a website that display all this information to the user, and we have the back end which we will house all the data and information to be displayed. I suggest we use HTML and JavaScript for the website (This is a programming language to create websites) and use Java to create the back end to house all the data. I also recommend these languages because I know Java and HTML relatively well, but JavaScript should be easy to pick up while doing this as it integrates with HTML. We could have the website as main display for the product and information and data can be changed in the back end.

I will make a simple looking design for both; I will try to create a nice-looking website with a nice back end. It should be easy to use and understand for the average user.

# Visual/interface designs

For the back end, I thought we could create a simple admin program that stores all the user’s passwords and usernames, what information can be sent to them, what products they own and more. I will create a simple looking login page for client to use and this will be the central hub for all the data.

I have the basics of what a simple login page could look like. We could have a tiny window that pops up asking for your details to login into the network. I have this as a simple looking design.

Graphical user interface, application

Description automatically generated

It not the nicest but can be improved upon to make it look better, this is the start of it. Most of the designs in this document are subjected to change. This would lead into a GUI that would have sections about each user, what information you want to display to them, what products you have. These are few things we could add. Another design was the main page when you enter the details.

**Main window, this is where you would navigate to access data and other information.**

---

X

Users Saved

Multiple imagines.

Logo

Extra section

Extra section

Products owned

For the website I want to make a simple login page with the user information displayed to them as one window. Such as fitness, health their account, contact and help, and more.

# 

Welcome user, what would you like to workout to today.

Logo!

Log out

# 

User account details:

Name

Email

Fitness points

Profile pic

# 

Image.

# 

Fitness choices and advice for the user if they need.

# 

Products advertised.

# 

Contact for help, extra information, policies.

This what I had in mid for a simple website, it is still subject to change as the client might not like it and wants to add more changes. When I start creating the website and back end the designs might change drastically because they will function and work better and will be organised better. These are the basics and will better in the final design, I decided that it would look nice if we had the user account information displayed on the website as gives them, more control over data and how it looks.

I made a quick design of the login page with IntelliJ built in GUI builder, I shorten it down to make it small and easy to use. It will have 4 buttons for each feature and clicking on them would open another window to use that feature. If a very rough design but could be improved upon to make it look nice.

Graphical user interface, application

Description automatically generatedI made it small, so it does not cover much of the screen. It has a user data button; this would open a window to show all the data and can be searched to find the user you want. It could have an email system that could message specific clients or people who contacted for help or to send other emails if you wish. Upload would be used to send information to variety or users with the information you want. Finally, a product button to see all the products you own and what users have that product to get better understanding of your clients.

# Data requirements

The data types will be using are:

* Strings: to hold information such as: login details, password details, emails, anything that text related.
* Floats: used to store numbers that are decimals such as currency; because not a whole number.
* Boolean: true or false to allow data pass and pass at times. Like incorrect passwords.
* Int: might come in useful in certain parts such as age.
* Long: for user ID as they get longer overtime.

# 

# Selection of algorithm designs

**Website flowchart.**

A picture containing text, sky, map

Description automatically generated

**Back-end flowchart.**

Diagram

Description automatically generated

Diagram

Description automatically generated

# Test strategy.

## What going to be tested.

The main parts that will be tested are:

* Outputs/inputs: such as login inputs, error outputs, correct window that get outputted, the input get registered by website and back-end.
* Data is being loaded: this would be checking that data can be loaded, it can be inputted into strings, and that it does not get corrupted.
* API: check that the API can communicate between website and back-end well and does not slow down or fail at working.
* Load time: make sure that both the website is fast and do not slow down at any time, this would ensure that code can be shorten down and more readable.

## Types of tests to be done.

These are the main one I believe we will use.

Unit testing: it is the process of testing parts of the code individual instead of running the whole program. This ensure that all the functions and methods are running correctly.

Smoke testing: it is the process to check that the most important functions work as intended and do not fail.

Integration testing: it is to test those certain functions/modules are working together as a group. It is to test a group of modules together to see if they can communicate with each other.

System testing: it is to check that the whole program works as intended and functions as it supposed to.

Compatibility testing: it is the process to check that your software/program can run on most software version on devices.

Code/script performance and functionality: it is the process of checking that all the GUI functions work correctly and can run.

Browser compatibility: it is to check that the software can work and run-on different browsers.

Loading times: to check how long it takes a page to load.

Responsiveness: to check how the program reacts to inputs/outputs and how fast it does it.

Acceptance testing: is used to test that the users like the product and works as intended.