High-Level Design

Revision Number: 1.0

 Last date of revision: 9/24/21

Brayan Fuentes, Curtis Nishihira

Application flow

Components

* Database
* GUI/Front End
* Server
* Main logic code

Hardware

* User’s device

Software

* Modules
  + Database -> main logic controller
  + GUI -> main logic  controller
  + User data module
  + Data grabber module

Class Notes:

Hardware

Clear understanding of what is a front end and what is a back end

Two concrete distinctions

Front End -> UI/UX

* Whatever the user interacts with
* Like a user interacting with a client application (program) sitting on the browser
* Client will talk to webserver (request)
* Do we need to include implementation language
* HTTP Request to the web server

Back End -> Features, logic, and data

* Consist of a web server which will give responses from client
  + Talks to SQL database (Cylinder)
    - Database Server
    - Relational database connects to set theory which is SQL
    - Relational Database management system ie Oracle MySQL
* Logic of implementing features

System should be adaptable to other technologies

Software

MVC

MVVM  
SPA  
PWA behave like mobile application

M->Model->Class

Represents description of entity

v-> view -> UI -> HTML, Swift, C++, Kotlin, Java

C -> Controller -> Routing/Navigation, Satisfy Request

MVVM

View model -> Class runs client side difference between controller and view model is where they live

Handles any logic needed for that specific view

One view model per view

SPA no service worker

Single

Page

Application

PWA service worker

Progressive

Web

Application

Client asks for new page from server server returns html

Initial request will give one html file

Subsequent requests will be async

Were using MVVM

We say the user is interacting with view which goes to vm which goes to back end

Were using SPA

User interacts with initial html then client side code then back end

User to html to vm to back end

How are you gonna handle input validation? Design choice

* Validating on client side is important to prevent roudtrips if email address is missing character
* Main logic of input validation

Error handling where will be error handling

* Program logic or user?
* Display messages to the user

Logging

* Theres no reason to have in front end

Security

* Only security right now is our login page

Diagram

Description automatically generated

Diagram

Description automatically generated

Diagram

Description automatically generated

Main Feedback: “The HL doesn’t need a specific section to cover points. A quality design will cover them either through diagrams or other content.”

HL office hour notes

Do we need to dicatate anything extra?

Remove feature logic from hardware

Queries are read operations

Write operations might not get results

Describing this machine is connecting with other machine might not get any output

Just annotate to show optional output UML Comment box links to response portion

You might get get results for all operations

Web app name

User is audience

Official deliverable not just a diagram start elaborations

Request first then response

Label interactions between modules

In system what other parts do you have to consider