# C:\Users\Messom\Desktop\Monash.jpg

# FIT5032 Design Report (Major Application Development Credit/Distinction/High Distinction)

# 

Hongliang Tang: 27135519

INSTRUCTIONS: Substitute all RED text with your information. DELETE all BLUE instructions before final submission (in PDF format). Feel free to edit the format of the document to improve presentation

**Contents Page**

**Your design report must include the following:**

1. **Overview** ( of your application’s goals)

**Title:** StudyEasy Education & Training System

**Description:** This application is designed to provide education and training services to users and to help the company manage their course and users. This educational company provide various training services that include:

• Personal productivity

• Language learning

• Health education (my project will focus on this service part)

• Sports skills

• Business development

• Science & Technology training

• Vocational education

• Career development

• Financial literacy

• …

Under each service, users can see various courses available for enrolling. For each course, which will be offered in studios and at different times. Users can choose any one studio under this course to fit their time schedule. Users cannot choose 2 studios that are hosted at the same time.

The end users can choose the services they want to receive, select tailored courses inside each educational service, pick the studio for courses enrolled and give ratings to services and courses. The staff can manage users' enrolments, organize the location and time that studios are provided, accept user enrolments and send emails to users.

1. **User stories and Use case diagrams (**that are driving your design decisions**)**

As a guest, I want to register my account so that I can become a user.

As a guest, I want to view the score rating so that I can know which course is the best.

As a user, I want to login so that I can see my panel.

As a user, I want to view ratings so that I can know which course is the best.

As a user, I want to view services so that I can know what curriculums are available.

As a user, I want to discontinue my course so that I can withdraw the course I don’t want to learn.

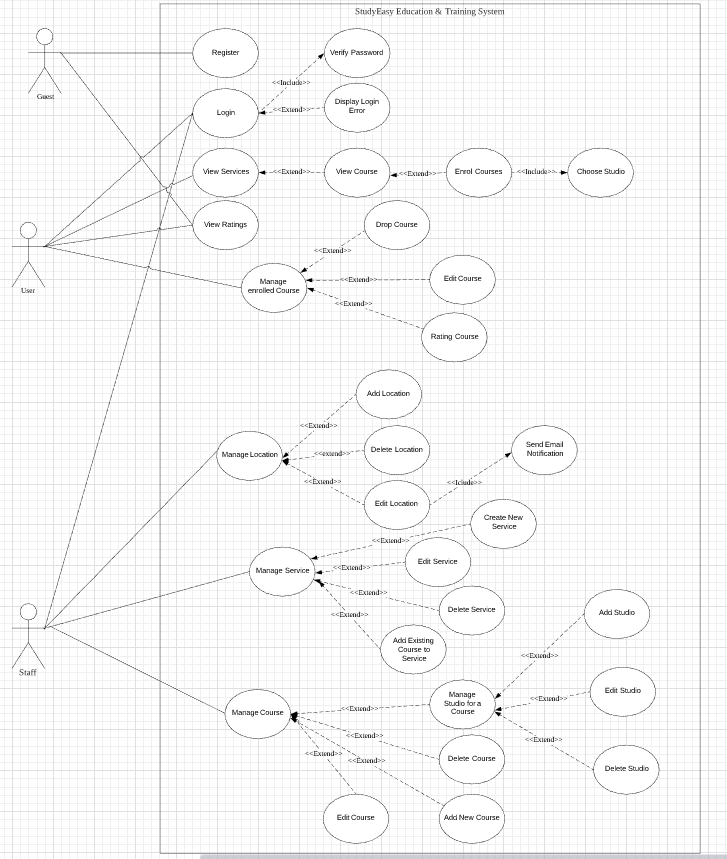
As a user, I want to change the studio for my course, so that I can organize my time schedule.

As a staff, I want to manage locations for the studio so that I can allocate locations.

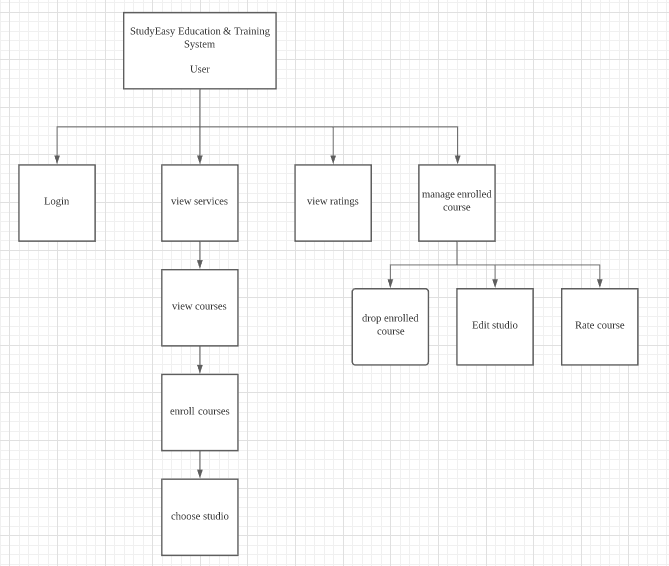
As a staff, I want to manage courses so that I can give user courses they can choose from.

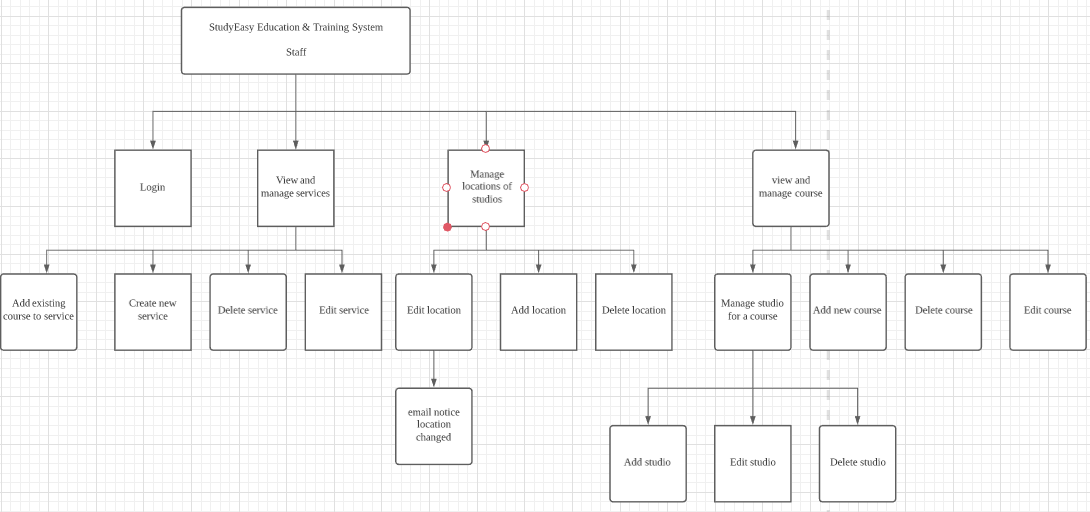
As a staff, I want to manage services so that I can present user services details.

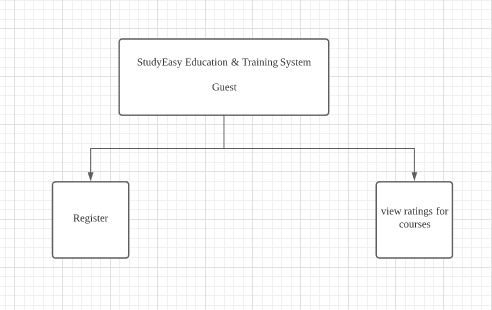
As a staff, I want to sent user email, so that I can notify them if studio rooms changes.



1. **Functional diagram** ( illustrating how the core aspects of the program fit together. You can provide either a functional diagram or a System Use Case diagram)







1. **Usability Design Review** (You can use either the Flow Bohl usability principle or Donald Norman Principles)

**Donald Norman Principles**

1. Visibility

There will be appropriate color, space and font size for navbar, and the description is easy to understand.

The home page is tidy and well-organized, consistent with all other pages, and has additional information provided for user if user want to know more.

For user enroll in a course, it will pop message to tell user enrollment successful.

User can see the location clearly with clear markings to guide them.

1. Feedback

Clear validation for all user inputs and checking user input. If user input incorrect information, it will ask user to correct it with proper guidance information on what should he type in.

Page navigation is fast and clear(when user click a button which jumps the page, the new page will show to the user right after click.

1. Affordance

Attributes and function icon will provide user clue on how to perform interactions. There will be tooltip guide user if he hovers the mouse above an interactive button.

1. Mapping

There will be clear relationships between controls and the effect. When user click a dropdown box, there will be choices listed. When user click a ranking button, the page will rank it accordingly.

1. Constraints

There are proper constraints for what user can do. It limits the range of interaction possibilities. For example, user cannot choose two studios that are hosted on the same time, if the user chooses one, the other one will be rendered grey and user won’t be able to choose it. For guest, some functionalities will be hiding from his page.

1. Consistency

The interface layout and style are consistent.

the button for submit will have the same design and color.

Having similar operations and similar elements for achieving similar tasks.

1. **Checklist of site functionality**

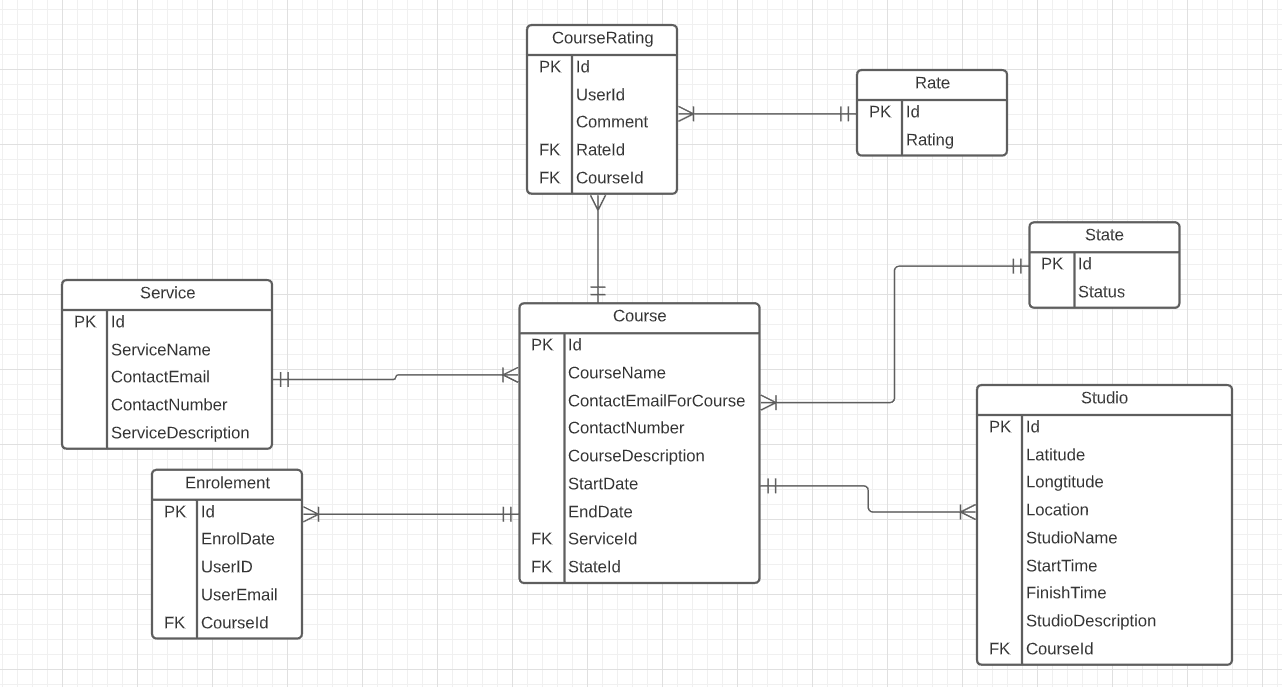
|  |  |
| --- | --- |
| **1. (Layout Page)** | **TICK**  **if complete** |
| Good Design |  |
| Stylesheet | done |
| JavaScript | done |
| Menu | done |
|  |  |
| **2. (Home page)** |  |
| Design and content | done |
| Banner Image | done |
|  |  |
| **3. (User Log in)** |  |
| Web form and validation controls | done |
| Formatted data entry display | done |
| Overall page design | done |
|  |  |
| **4. (Customised Views and Controllers)** |  |
| Customised Views |  |
| Customised Controllers | done |
| Other customisations |  |
|  |  |
| **5. (Documentation)** |  |
| Code Comments |  |
| Attribution of Source of any code used |  |
|  |  |
| **6 Business Requirements** |  |
| **BR(A1): for C to C+** | done |
| **BR(A2): for C to C+** | done |
| **BR(B1): for C to C+** | done |
| **BR(B2): for C to C+** | done |
| **BR(C1): for C+ to C++** | done |
| **BR(C2): for C+ to C++** | done |
| **BR(C3): for C+ to C++** | done |
| **BR(D1): for D to D++** | done |
| **BR(D2): for D to D++** | done |
| **BR(D3): for D to D++** |  |
| **BR(D4): for D to D++** | done |
| **BR(E1): for HD to HD+** |  |
| **BR(E2): for HD to HD+** |  |
| **BR(E3): for HD to HD+** |  |
| **BR(E4): for HD to HD+** |  |
| **BR(E5): for HD to HD+** |  |
|  |  |
| **Audit** |  |
| No breaking of copyright |  |

**Additional Distinction Level (the above and the following)**

1. **Your selected approach when constructing the application.** (Model First, Code First or Database First)

Database First

1. **Class Diagram or Entity Relation Diagram** (If you are using the Model First or Code First approach, you will need to provide a class diagram. An Entity Relationship Diagram (ERD using Crows Feet or Chen Notation) should be provided if you decide to use the Database First methodology.)



1. **Data dictionary** (This includes the justification of the data types which you use)

**int:Id**

**String:Rating, Status,Latitude,Longitude,StudioName,StudioDescription,CourseName,ContactEmailForCourse,ContactNumber,CourseDescription,UserEmail,UserId,ContactEmail,ContactNumber,ServiceDescription,ServiceName,Comment**

**DateTime:StartDate,EndDate**

**Time: StartTime,FinishTime**

**The use of int is for id automatic increment. The use of string for the majority of datas due to its flexibility and validation simplicity**

**The user of DateTime is for Date criteria as required in the assignment and Time is for Studio Time span.**

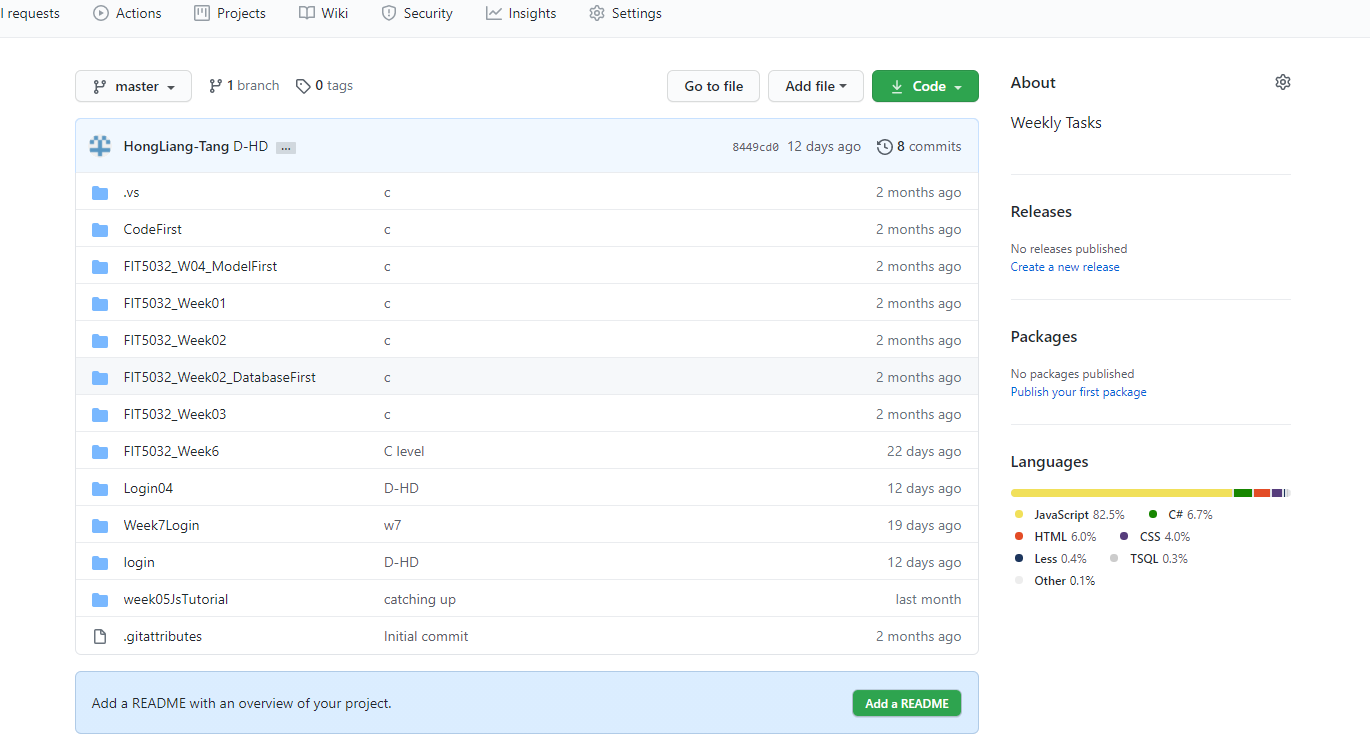
**Additional High Distinction Level (the above and the following)**

1. **Development Methodology** (A description of your development methodology for this portfolio submission. (For example, did you approach the portfolio submission using a Code and Fix method or using a Test Driven Development). Most students are expected to use the Code & Fix model. (This is completely normal for all assignments))

Code & Fix model. The code and fix model probably is the most frequently used development methodology in software engineering. It starts with little or no initial planning. You immediately start developing, fixing problems as they occur, until the project is complete.

1. **Versioning** (Description of how you handled the versioning of your project. For example, if you have used [Git](https://www.atlassian.com/git/tutorials/what-is-git), [Mercurial](https://www.mercurial-scm.org/), [CVS](https://en.wikipedia.org/wiki/Concurrent_Versions_System), or [SVN](https://subversion.apache.org/). Please provide evidence of your repository if you have done so. (If you did not use it, justify why). It is not a requirement to use these tools but if you have done so, please state it.)

**Github.**

****