

Faculty of Information Technology

Computer Science Department

Free Distribution Resources System

(FDRS)

Graduation Project (1) Report

Prepared by:

|  |  |
| --- | --- |
| **Students Name** | **Student ID** |
| **Wasef jayousi** | **201920074** |
| **Anas asyed** | **202020067** |
| **Saif karborani** | **202020088** |

Supervised by:

Ashraf Odeh

Semester / Academic Year

Copyright © 2022-2023 - All rights reserved

Middle East University

Declaration

We hereby acknowledge that the work presented in this document report and the ideas based upon are the group members own unless stated otherwise and properly cited in text and referenced at the end of the document.

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Signature | Students Name | Student ID |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

..

Table of Contents

[List of Figures and tables 4](#_Toc135920481)

[Chapter 1: Introduction 6](#_Toc135920482)

[1.1 Description of the Problem 6](#_Toc135920483)

[1.2 Description of the Solution 7](#_Toc135920484)

[1.3 Literature Review (related work) 8](#_Toc135920485)

[1.4 Technology and tools to be used 8](#_Toc135920486)

[Chapter 2 : Project Plan 9](#_Toc135920489)

[2.1 Project Objectives 9](#_Toc135920489)

[2.2 Project Scope 10](#_Toc135920490)

[2.3 Software Process Model 12](#_Toc135920491)

[2.4 Project Schedule 13](#_Toc135920492)

[2.5 Project Schedule Chart(s) (Bar/Gantt Chart) 14](#_Toc135920493)

[Chapter 3: Requirements and Analysis 15](#_Toc135920494)

[3.1 Functional Requirements 15](#_Toc135920497)

[3.2 Non-Functional Requirements 16](#_Toc135920498)

[Chapter 4: Architecture and Design 17](#_Toc135920499)

[4.1 Architecture 17](#_Toc135920501)

[4.2 Use Case Diagram 18](#_Toc135920502)

[4.3 Use Cases Descriptions/Flow of Events 19](#_Toc135920503)

[4.4 Class Diagram 30](#_Toc135920504)

[4.5 Entity Relationship Diagram 31](#_Toc135920505)

[Chapter 5: Conclusion & Future Work 32](#_Toc135920506)

[5.1 Conclusion 32](#_Toc135920508)

[5.2 Future work 32](#_Toc135920509)

[References 33](#_Toc135920510)

# List of Figures and tables

**List of figures**

[2.1 Software Process Model 12](#_Toc135920491)

[2.2 Project Schedule 13](#_Toc135920492)

[2.3 Project Schedule Chart(s) (Bar/Gantt Chart) 14](#_Toc135920493)

[4.1 Architecture 17](#_Architecture)

[4.2 Use Case Diagram 18](#_Toc135920502)

[4.5 Entity Relationship Diagram 31](#_Toc135920505)

**List of tables**

[Table 2.1…………………………………………………………………………………………………………………………………………………10](#_Toc135920490)

[Table 4. 1 20](#_Toc135984979)

[Table 4. 2 21](#_Toc135984980)

[Table 4. 3 22](#_Toc135984981)

[Table 4. 4 23](#_Toc135984982)

[Table 4. 5 24](#_Toc135984983)

[Table 4. 6 25](#_Toc135984984)

[Table 4. 7 26](#_Toc135984985)

[Table 4. 8 27](#_Toc135984986)

[Table 4. 9 28](#_Toc135984987)

[Table 4. 10 29](#_Toc135984988)

[Table 4. 11 30](#_Toc135984989)

[Table 4. 12 31](#_Toc135919149)

Abstract

One of the biggest issues students suffer from while studying a course in there university is finding enough resources to make them understand the course they are taking. A tool like a resources library for each course and for each major in university will help and support students. The resources main source on the website will be uploaded and categorized from students who previously took the course, they will upload the documents, videos, websites that helped them through taking the course.

# Chapter 1: Introduction

This chapter is about illustrating the description of the challenge (problem statement) and the solution, related work to be done and the Technology and tools that were going to use later so we can implement our project (website)

## Description of the Problem

The challenge facing students and researchers in universities today is the availability of reliable and up-to-date academic references. Many students struggle to find relevant resources for their studies, which can impact their academic performance.

The challenge is to make an FDRS that will be experienced and rated from students to make it easier for future students on how to be more productive and keep the library updated.

The problem is that many existing libraries may not have resources that are specific to a particular major or field of study. In addition, many academic resources are expensive and not readily available to students, making it difficult for them to access the latest research and information.

This problem is compounded by the fact that traditional libraries may not have the latest updates or revisions to academic resources, which can negatively impact students' studies.

The opportunity is to create an FDRS that caters to different majors and provides a comprehensive collection of academic references. An FDRS that has student-generated content can be a valuable tool for students, allowing them to share their resources and experiences with their peers. Additionally, an FDRS that is regularly updated with the latest research and information can help students stay up-to-date with their studies and improve their academic performance.

## Description of the Solution

The FDRS is a digital resource that allows users to upload and share their academic references with other users. This library caters to various majors in universities, including but not limited to Information Technology, Business, Engineering, Medicine, Law, and Social Sciences. The library's collection includes books, videos, research papers, and other relevant resources, which are contributed by other students with experience in the major or subject.

The resource library is designed to facilitate collaboration between users, allowing them to share their academic resources and experiences with other users “Students”. Students can upload their own resources, which they find useful and share them with their peers, ensuring that the resource library collection is always up-to-date and comprehensive, each user can rate, review, and make a favorite list with favorite users that upload frequently, and a history list with every downloaded resources by user will be automatically initiated in the users profile.

The resource library has a user-friendly interface that is easy to navigate, allowing users to search for resources based on their specific needs and interests. The library is organized by majors, with each major having a dedicated section that contains resources specific to their field of study. The library's search feature also allows users to find resources based on keywords or topics.

The resource library encourages users to contribute to the library, ensuring that the library's collection continues to grow and expand. The library's community-driven approach promotes collaboration and knowledge-sharing among users and researchers.

## Literature Review (related work)

In this related work, we are going to see and maybe upgrade on other students project that made this project similar.

Related work:

In this related work I have looked on another project that is similar to our Problem and I found that their system is mainly about book issues for students, what is meant here by book issues are the librarian (Admin) writes your name and details and gives you the book & tells you the time period of returning the book, so this is in real life library. but in the system it actually does a lot more, the admin can monitor the whole system .It also has facility of an online notice board where teachers and student can put up information about workshops being held in our colleges or nearby colleges and librarian after proper verification from the concerned institution organizing the workshop can add it to the notice board. The students when they log in and have an account they can see list of books issued and its date , their return date and the can request from the librarian which is the admin to add new books by filling a request form the student made [1].

Their interface is much uglier. We will make ours much easier to navigate and probably more modern. Their system is about a student user and a teacher user and they both have similar features, Our system improvement would be rather than a student that has to log in and a teacher that has also to log in, we can put a guest automatic log in with less features so that with anybody wants to enter the website without logging in he can still do the core thing that the system provides.

## Technology and tools to be used

* VSCODE
* GITHUB
* External embedded applications.
* Visual Paradigm
* FRONTEND: HTML , CSS , JAVASCRIPT
* BACKEND:
* DATABASE:

# Chapter 2: Project Plan

In this chapter, it illustrates about the project objectives, project scope, and the software process model that we are going to use for us to do the project, also project scheduling.



## Project Objectives

* Provide users in the university with the new up to date academic references.
* Provide users in the university to ability to share and upload their academic resources.
* Provide users to search for resources (Books...Etc.) Based on their specific needs.
* Provide users a user friendly interface that it easy to navigate.
* Provide users the ability to download the required resource if available.
* Provide users with video tutorials.
* Provide users the ability to review the Resource (book…etc.)
* Provide a guest with less features
* An admin login page where admin can add books, videos or page sources.
* Open link for learning websites.

An admin who has a complete control of what he wants to distribute, for the user who wants to sign up

Will have a feature of a history list of what the book that he searched for and viewed or downloaded

, and also can give feedbacks and have some extra features, unlike the guest who only wants to view or download a resource.

Also will give the admin the ability to control the user on what resources he uploads and reviews and the ability to ban or delete the user.

## Project Scope

|  |
| --- |
| **Date:** March 2023 |
| ***General Project Information***  **Project Name:**  Resource library for students  **Sponsor: Middle east university**  **Project Manager: Dr. Ashraf Odeh**  **Prepared by :**   * Anas Alseid * Wasef Jayousi * Saif Karborani | |
| ***Project Objectives :***   * Provide students in the university with the new up to date academic references. * Provide students in the university to ability to share and upload their academic resources. * Provide students to search for resources (Books...Etc.) Based on their specific needs. * Provide students a user friendly interface that it easy to navigate. * Provide students the ability to download the required book if available.   Provide students with video tutorials. | |
| ***Project Description:***  ***One of the biggest issues students suffer from while studying a course in there university is finding enough resources to make them understand the course they are taking. A tool like a resources library for each and for each major in university will help and support students. The resources main source on the website will be uploaded and categorized from students who previously took the course, they will upload the documents, videos, websites that helped them through taking the course. students who downloaded any document a history list will track there downloads to help*** | |
| ***Business Benefits:***   * ***Improving the way students contribute with each other*** * ***Improving how the students study and save time while researching*** * ***Making important resources more easy to access by users*** * ***Easy to search in resources library*** | |
| ***Project Deliverables:***   * ***User friendly interface*** * ***User history of downloads list*** * ***User login page*** * ***User can upload documents under his name*** * ***User can review , rate , different documents that got uploaded to the website*** | |
| ***Estimated Project Duration:***  7 Months | |

Table 2.1

## Software Process Model

The Software Development Life Cycle (SDLC) is a structured process that enables the production of high-quality, low-cost software, in the shortest possible production time. The goal of the SDLC is to produce superior software that meets and exceeds all customer expectations and demands. The SDLC defines and outlines a detailed plan with stages, or phases, that each encompass their own process and deliverables. Adherence to the SDLC enhances development speed and minimizes project risks and costs associated with alternative methods of production.

We are going to implement it because it fits our project and because it fits our Problem the system we are doing [2].



Figure 2.1

## Project Schedule

Graduation project 1

Figure 2.2

## Project Schedule Chart(s) (Bar/Gantt Chart)

Figure 2.3

Notice: to open this link you need to first install the project professional App by Microsoft

This is alive link on the project schedule <Project1.mpp>

# Chapter 3: Requirements and Analysis



## Functional Requirements

1. User Registration and login - Users should be able to create an account, login, and access only the features that they are authorized to use.

2. Recourse Catalog - The system should provide a searchable catalog of all recourse available in the FDRS along with their metadata such as title, author, genre, publication year, publisher, and availability status.

3. Recourse view - Users should be able to view and read recourse online or download them for offline reading.

4. Recourse Search - Users should be able to search for recourse by title, author, keyword, or category.

5. User Interaction - Users should be able to rate, review, and comment on recourse, as well as share them on social media.

6. Admin Panel - The system should have an admin panel to manage users, recourse, and get feedback from users for unfound search resources.

7. User Profile Management - Users should be able to manage their profile information, update their preferences, and view their reading history.

8. User Upload – Users should be able to upload resources on the website.

9. User Favorite List – Users should be able to add their favorite Resource on their List.

10. Forget password – User can regain access to their account by resetting their forgotten password.

## Non-Functional Requirements

1. Performance - The system should handle many concurrent users and transactions without slowing down or crashing, with fast page load times and smooth scrolling for online reading.
2. Availability - The system should be available 24/7, with minimal downtime for maintenance or upgrades, and with high availability to ensure users can access the system whenever they need it.
3. Reliability - The system should be reliable and provide accurate information, without losing data or causing errors, with frequent backups to ensure that data is not lost.
4. Scalability - The system should be able to scale up or down as needed to accommodate changes in user traffic or library collections, with the ability to add more servers and storage space as required.
5. Security - The system should be secure, with measures in place to protect user data, prevent unauthorized access, and detect and respond to security breaches, with SSL encryption and strong password policies.
6. Usability - The system should be easy to use, with an intuitive user interface and clear instructions for performing tasks, with clear and concise user documentation available.
7. Compatibility - The system should be compatible with different web browsers and operating systems, and support different languages and character sets, with a responsive design to ensure that it works on different devices.
8. Maintainability - The system should be easy to maintain and upgrade, with clear documentation and well-structured code, with change management policies in place to prevent unplanned changes.

# Chapter 4: Architecture and Design



## Architecture

Since we are building our project on a website, our Architecture will be following an Architecture pattern.

And that is the layered architecture, the layered Architecture have the most common sense for our system

Its major layers are the PRESENTATION LAYER, APPLICATION LAYER, BUSINESS LOGIC LAYER ,And the Data Layer, each layer have different components that communicate with each other and pass from higher layer to lower layers , for each layer it uses the layer below it to get the service it needs.

As for now since our project is on a website, we have the Front END which is the presentation layer using html, css, JavaScript, and for the BACKEND it’s going to be either the Application layer and business logic layer and sometimes these 2 layers are combined, finally we need to have a database which will be the data layer.

Now we can say that the 3 layers are in another terminology is the frontend, the backend, the database (part of the backend).

* Now for the first layer (presentation) the components are: The browser, user interface, dashboards, configurational settings and etc. so in general this is the client side view.
* Using html, css, JavaScript and some frame works we are going to build this first layer.
* For the second layer(business logic layer layer) this will be the server-side component is the key component of the web application architecture that receives user requests, performs business logic and delivers the required data to the front-end systems.
* Data Layer: A Database is a key component of a web application that stores and manages information for a web app. using a function, you can search, filter and sort information based on user request and present the required info to the end user. They allow role-based access to maintain data integrity.

As it seems our architecture is a 3 tier architecture

The 3-tier architecture is more secure as the client does not directly access the data. The ability to deploy application servers on multiple machines provides higher scalability, better performance and better re-use. You can scale it horizontally by scaling each item independently. You can abstract the core business from the database server to efficiently perform load balancing. Data integrity is improved as all data goes through the application server which decides how data should be accessed and by whom. For that reason, a change of management is easy and cost-effective. The client layer can be a thin-client which means hardware costs are reduced. This modular model allows you to modify a single tier without affecting the remaining components [3].



Figure 4.1

## Use Case Diagram

A use case Diagram that Describes who is the actors of the system and the Action (Use Case Base) they do.

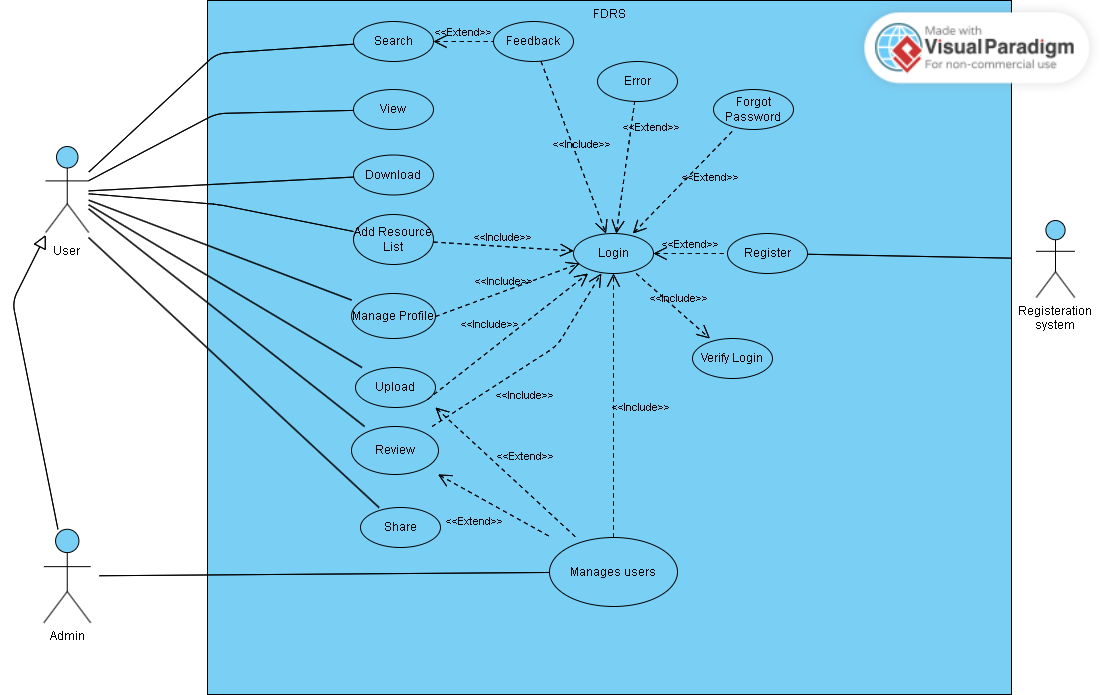


Figure 4.2

## Use Cases Descriptions/Flow of Events

|  |  |
| --- | --- |
| **Use Case name:** | Login |
| **Use Case ID:** | 1 |
| **Description:** | This use case describes how the actor logs into the System. |
| **Primary Actor:** | User/admin. |
| **Secondary Actor:** | Registration system. |
| **Pre-condition:** | The user must register (use case must be performed first). |
| **Main flow of events:** | User/admin:   1. The system requests that the actor enter his/her E-mail and password. 2. The actor enters his/her E-mail and password and presses ‘login’ option. 3. The system validates the entered E-mail and password and logs the actor into the system (Checks against the database).   Sub-Flow :   1. The system requests that the user enter his email 2. User enter his email 3. The system sends an email to the user mailbox , a link to change his password |
| **Alternative Flow of events:** | * The system displays an error message that the entered E-mail and/or password is invalid. * The system requests that the actor re-enters his/her E-mail and password. |
| **Post-condition:** | The user that logins in will be able to do his features (Search, view, download, review, upload, track List, Share).  The admin That login in will have the Extra feature of managing the users. |

Table 4.

|  |  |
| --- | --- |
| **Use Case name:** | Register(Create new account) |
| **Use Case ID:** | 2 |
| **Description:** | This use case describes how an actor creates a new user account. |
| **Primary Actor:** | User. |
| **Secondary Actor:** | Registration system. |
| **Pre-condition:** | None. |
| **Main flow of events:** | 1. The actor clicks on (Register) option from the home page. 2. The system prompts the actor to enter his/her full name, E-mail and password. 3. The system validates the entered E-mail and password via the registration system meaning checking if the email is reused and then redirects the actor to the login page (A1). |
| **Alternative Flow of events:** | A1:   * The system displays an error message that the E-mail and/or password is invalid. * The system requests that the actor re-enters his/her E-mail and password. * The system displays to the user that the Email is already in use. |
| **Post-condition:** | Added the user to the database via the registration system actor. |

Table 4.

|  |  |
| --- | --- |
| **Use Case name:** | Search |
| **Use Case ID:** | 3 |
| **Description:** | This use case describes how the actor Search through resources. |
| **Primary Actor:** | User/admin. |
| **Secondary Actor:** | ... |
| **Pre-condition:** | None. |
| **Main flow of events:** | User/admin:   1. The user uses the search bar to type the resources that he is looking for. 2. The system will scan the database based on the search type. 3. Then the system returns the desired Search resource. |
| **Alternative Flow of events:** | * The system tells the user that the search is not Found/invalid |
| **Post-condition:** | The user should then be able to see the resource. |

Table 4.

|  |  |
| --- | --- |
| **Use Case name:** | View |
| **Use Case ID:** | 4 |
| **Description:** | This use case describes how the actor view’s the resource. |
| **Primary Actor:** | User/admin. |
| **Secondary Actor:** | ... |
| **Pre-condition:** | None. |
| **Main flow of events:** | 1. The user clicks on the view button. 2. The system redirects the user into another page. |
| **Alternative Flow of events:** | * The system return an error. |
| **Post-condition:** | When the resources accepted from the admin the user can view it |

Table 4.

|  |  |
| --- | --- |
| **Use Case name:** | Download |
| **Use Case ID:** | 5 |
| **Description:** | This use case describes how the actor can Download a resource from the website |
| **Primary Actor:** | User |
| **Secondary Actor:** |  |
| **Pre-condition:** | None |
| **Main flow of events:** | 1. The user selects a resource to download. 2. The system transfers the resource from the database (server) to the user website downloads. |
| **Alternative Flow of events:** | * The systems return to the user that the resource isn’t available to download. |
| **Post-condition:** | The user has the resource on his device, drive. |

Table 4.

|  |  |
| --- | --- |
| **Use Case name:** | Add resource list |
| **Use Case ID:** | 6 |
| **Description:** | This use case describes how the user can add his Favorite list of resources |
| **Primary Actor:** | User/admin |
| **Secondary Actor:** |  |
| **Pre-condition:** | The user should have an account and logged in |
| **Main flow of events:** | 1. The user selects the resource. 2. The user clicks on add to list. 3. The system returns that it has been added to the list. |
| **Alternative Flow of events:** | * The system returns to the user that he can’t add more to the list. * The system displays an error message. |
| **Post-condition:** | The user can view his resource list. |

Table 4.

|  |  |
| --- | --- |
| **Use Case name:** | upload |
| **Use Case ID:** | 7 |
| **Description:** | This use case describes how the user can upload resources on the website |
| **Primary Actor:** | Users. |
| **Secondary Actor:** | Admin. |
| **Pre-condition:** | Logged in. |
| **Main flow of events:** | 1. The user clicks on the upload button and selects the resource he wants to upload. 2. The website displays a proper form with fields for the user to upload what kind of resource. 3. The admin reviews it and accepts (A1) 4. The system proceeds to the database and add the resource. 5. The system returns to the user that upload is successful and has been added to the website. |
| **Alternative Flow of events:** | * The system returns that the upload is invalid. * The admin Declines the upload (A1). |
| **Post-condition:** | Users can see the uploaded resource. |

Table 4.7

|  |  |
| --- | --- |
| **Use Case name:** | Review |
| **Use Case ID:** | 8 |
| **Description:** | The use case describes how the user can review on the resource, in which he can put a comment or rate. |
| **Primary Actor:** | User. |
| **Secondary Actor:** | Admin. |
| **Pre-condition:** | Logged in. |
| **Main flow of events:** | 1. User selects the resource to review. 2. The website displays a review form with fields for the user to enter their review information (comment, rating) and submits. 3. The website displays the review on the resource for other users to see. |
| **Alternative Flow of events:** | * The review may be reviewed by the admin and deletes it. |
| **Post-condition:** | Users can see the review on the resource. |

Table 4.

|  |  |
| --- | --- |
| **Use Case name:** | Share |
| **Use Case ID:** | 9 |
| **Description:** | This use case describes how the user can share the resource |
| **Primary Actor:** | User/admin |
| **Secondary Actor:** |  |
| **Pre-condition:** | None. |
| **Main flow of events:** | 1. The user selects the resource that he wants to share. 2. The system returns a link of the resource to the user. 3. The user copies the link. |
| **Alternative Flow of events:** | * The system fails providing the link. |
| **Post-condition:** | The system delivers The link to the user. |

Table 4. 9

|  |  |
| --- | --- |
| **Use Case name:** | Manage users |
| **Use Case ID:** | 10 |
| **Description:** | This use case describes how the admin validates the users uploads and reviews. |
| **Primary Actor:** | Admin. |
| **Secondary Actor:** | User. |
| **Pre-condition:** | Logged in as an admin. |
| **Main flow of events:** | 1. The system shows to the admin the upload’s of the users. 2. Admin accepts them. 3. The system returns to the user that their upload has been approved. 4. As for the review (comments of users) 5. The admin clicks on the users comment 6. Hit delete button 7. System returns that comment has been deleted. |
| **Alternative Flow of events:** | * The admin declines. * The system returns to the users that the upload has been rejected. |
| **Post-condition:** | The admin can Reject users uploads and reviews. |

Table 4.

|  |  |
| --- | --- |
| **Use Case name:** | Manage profile |
| **Use Case ID:** | 11 |
| **Description:** | This use case describes how the user can edit his username or password. |
| **Primary Actor:** | User. |
| **Secondary Actor:** |  |
| **Pre-condition:** | Logged in |
| **Main flow of events:** | 1. The user clicks on user icon or manages profile button 2. The system displays the user information. 3. The user edits their profile information and saves. 4. The system displays to the user that the Profile changes has been done. |
| **Alternative Flow of events:** | * The system returns that the edit is invalid. |
| **Post-condition:** | User is able to change his profile settings. |

Table 4.

|  |  |
| --- | --- |
| **Use Case name:** | Feedback |
| **Use Case ID:** | 12 |
| **Description:** |  |
| **Primary Actor:** | User. |
| **Secondary Actor:** | Admin |
| **Pre-condition:** | Search and log in use case must be performed first. |
| **Main flow of events:** | 1. If the resource is not found. 2. User Shares his feedback by clicking a button 3. The system will catch the result text and username. 4. The system will store it in the database for the admin to read. |
| **Alternative Flow of events:** | * System fails to get user feedbacks. |
| **Post-condition:** | User will be able to share his feedbacks. |

Table 4. 12

## C:\Users\mrben\OneDrive\سطح المكتب\FDRS_ClassDia.jpgClass Diagram

## Entity Relationship Diagram

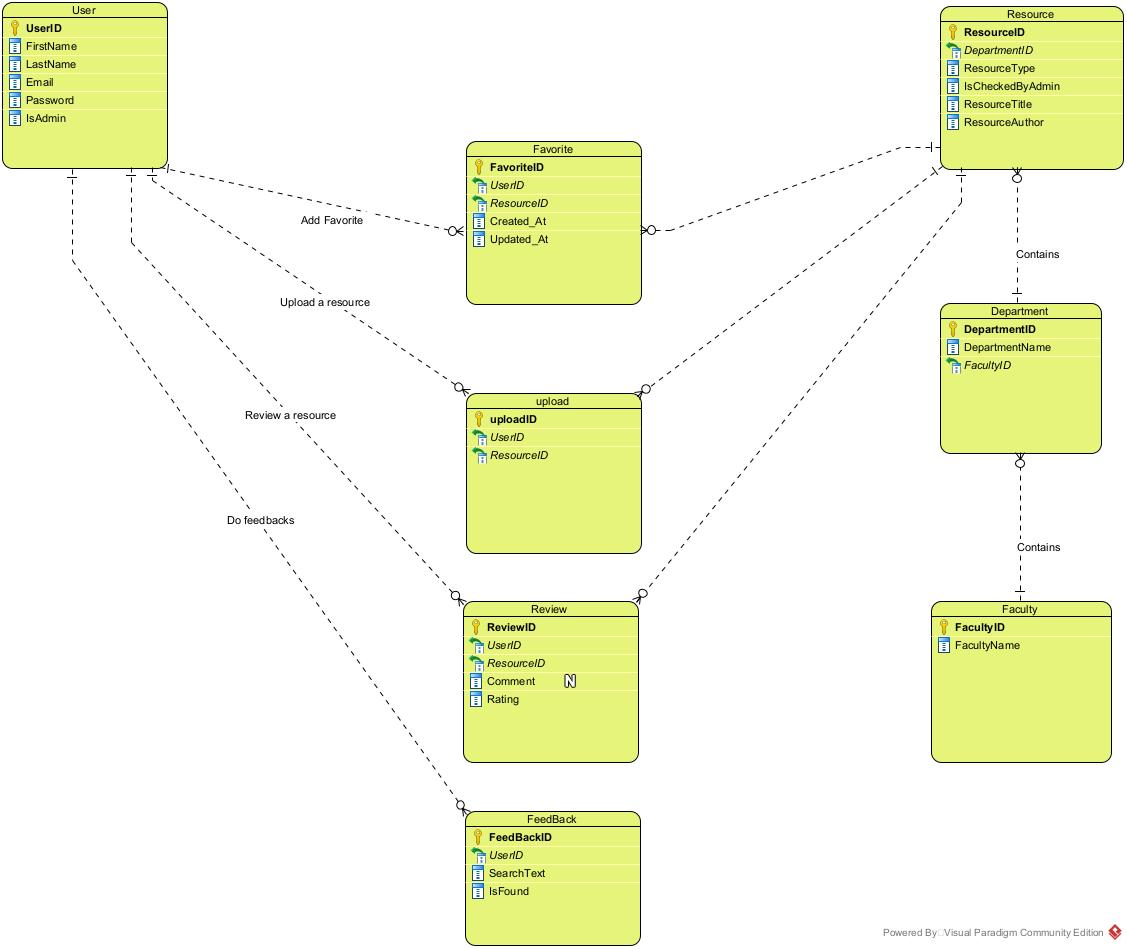


Figure 4.3

# Chapter 5: Conclusion & Future Work



## Conclusion

## Future work

# 

# References

1. Bernd Bruegge Allen H.Dutoit (2010) Object-Oriented Software Engineering Using UML, Patterns, and Java 3rd Edition.
2. Related Work [1]. <https://www.academia.edu/37726542/Library_Management_System_Mini_Project_Report_On_LIBRARY_MANAGEMENT_SYSTEM>
3. Why to use SDLC [2] <https://www.synopsys.com/glossary/what-is-sdlc.html#:~:text=The%20SDLC%20defines%20and%20outlines,with%20alternative%20methods%20of%20production>.
4. An Overview of Web Application Architecture. (2022).<https://www.clickittech.com/devops/web-application-architecture/> [3]