BỘ NÔNG NGHIỆP VÀ PHÁT TRIỂN NÔNG THÔN
TRƯỜNG ĐẠI HỌC THỦY LỢI

BÀI TẬP LỚN

HỌC PHẦN: LẬP TRÌNH ANDROID

ĐỀ TÀI: Xây dựng ứng dụng My Reading Manga App

Mã Sinh Viên	Họ và Tên	Ngày Sinh	Lớp
2151061187	Nguyễn Viết Sơn	04/05/2003	63CNTT.NB
2151062706	Nguyễn Quỳnh Anh	09/05/2003	63CNTT.NB
2151062832	Trịnh Công Minh	20/01/2003	63CNTT.NB

Hà Nội, năm 2024

BỘ NÔNG NGHIỆP VÀ PHÁT TRIỂN NÔNG THÔN
TRƯỜNG ĐẠI HỌC THỦY LỢI

BÀI TẬP LỚN

HỌC PHẦN: LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

ĐỀ TÀI: Xây dựng ứng dụng đọc truyện tranh MyReadingManga (Java Console App)

Mã Sinh Viên	Họ và Tên	Ngày Sinh	Điểm	
			Bằng Số	Bằng Chữ
2151061187	Nguyễn Viết Sơn	04/05/2003		
2151062706	Nguyễn Quỳnh Anh	09/05/2003		
2151062832	Trịnh Công Minh	20/01/2003		

CÁN BỘ CHẤM THI 1 **CÁN BỘ CHẤM THI 2**

Hà Nội, năm 2024

MỤC LỤC

MỤC LỤC HÌNH ẢNH

MỤC LỤC BẢNG

BẢNG CÁC TỪ VIẾT TẮT

STT	TỪ VIẾT TẮT	VIẾT ĐẦY ĐỦ
1	CSDL	Cơ sở dữ liệu
2		

CHƯƠNG 1. MÔ TẢ BÀI TOÁN

1.1. Giới thiệu

Ứng dụng đọc truyện tranh là một công cụ tiện lợi, cung cấp rất nhiều đầu truyện phong phú, từ các thể loại như hành động, lãng mạn, phiêu lưu, hài hước đến kinh dị, viễn tưởng, đáp ứng nhu cầu giải trí của người dùng. Ứng dụng cho phép người dùng tìm kiếm, đọc và tải truyện tranh, cũng như lưu lại truyện yêu thích và đánh dấu chương đã đọc.Ngoài ra, ứng dụng cho phép người dùng có thể đăng truyện tranh của mình lên cho 1 lượng lớn người đọc. Ứng dụng sẽ được xây dựng bằng Java và chạy trên giao diện console (command-line).

1.2. Chức năng chính

Ứng dụng MyReadingManga cần có các chức năng sau:

1. **Chức năng liên quan đến tài khoản người dùng:** đăng nhập, đăng ký, quên mật khẩu, đổi avatar, đổi tên hiển thị, đăng xuất.
2. **Chức năng xem danh sách truyện:** hiển thị danh sách truyện mới cập nhật, truyện được yêu thích nhiều nhất.
3. **Chức năng xem truyện:** hiển thị thông tin truyện, thêm/xóa danh sách truyện yêu thích, đọc truyện từ chương đầu/chương mới nhất, xem danh sách bình luận của truyện, thêm/sửa/xóa bình luận của mình; đánh giá truyện**:** đánh giá điểm của truyện trên thang 5.
4. **Chức năng đọc truyện:** xem danh sách ảnh của chương truyện muốn đọc, chuyển sang đọc chương trước/sau, chọn chương trong danh sách chương của truyện để đọc.
5. **Chức năng xem tủ sách:** hiển thị danh sách lịch sử truyện có chương gần nhất đã đọc và danh sách truyện yêu thích.
6. **Chức năng đăng truyện:** đăng truyện, chương truyện (chưa có trong kho truyện) lên kho truyện.
7. **Chức năng tìm kiếm truyện:** tìm kiếm truyện theo: tên truyện, thể loại.
8. **Chức năng thay đổi cài đặt giao diện:** bật/tắt chế độ tối
9. **Chức năng thông báo:** hiển thị thông báo các chương mới cập nhật của truyện có trong danh sách yêu thích.

1.3. Yêu cầu phi chức năng

- Dễ sử dụng: Giao diện cần rõ ràng, dễ hiểu và dễ sử dụng.
- Hiệu năng: Ứng dụng cần hoạt động ổn định, mượt mà

CHƯƠNG 2. PHÂN TÍCH YÊU CẦU VÀ THIẾT KẾ HỆ THỐNG

2.1. Phân tích yêu cầu:

2.1.1. Xác định người dùng:

- Người dùng :
    - Những người yêu thích đọc truyện tranh và muốn có trải nghiệm đọc trực tuyến thuận tiện trên thiết bị di động.
    - Người hâm mộ truyện tranh từ nhiều thể loại khác nhau như hành động,phiêu lưu , kinh dị, hài hước, lãng mạn và nhiều thể loại khác.
    - Giới trẻ sinh viên và những người lớn tuổi có nhu cầu giải trí và thư giãn bằng cách đọc truyện tranh.
    - Những tác giả trẻ có thể đăng truyện tranh của mình lên cho nhiều người đọc

2.1.2. Thu thập yêu cầu:

1. **Thư viện truyện tranh đa dạng:** Ứng dụng cần cung cấp một thư viện lớn với nhiều thể loại truyện tranh khác nhau để đáp ứng nhu cầu của đa dạng người dùng.
2. **Tìm kiếm và lọc truyện tranh:** Người dùng cần có khả năng tìm kiếm truyện tranh theo tên, tác giả, thể loại, người dịch.
3. **Đánh dấu truyện yêu thích**: Người dùng muốn có thể đánh dấu các trang truyện tranh yêu thích của mình để có thể dễ dàng truy cập sau này.
4. **Theo dõi lịch sử đọc**: Ứng dụng cần lưu trữ lịch sử chương truyện đã đọc của người dùng.
5. **Đăng truyện đã dịch**: Người dùng có thể đăng truyện, chương truyện của mình (mới, chưa có trong kho truyện) lên kho truyện để mọi người cùng đọc.
6. **Tùy chỉnh giao diện**: Tính năng chế độ đọc ban đêm giúp mắt người dùng không bị mỏi trong lúc sử dụng ứng dụng. Ngoài ra, người dùng muốn có thể chuyển đổi ngôn ngữ của ứng dụng theo nhu cầu của họ.
7. **Bình luận**: Tính năng bình luận truyện tranh cho phép người dùng chia sẻ ý kiến và tương tác với cộng đồng yêu thích truyện tranh.
8. **Thông báo:** Ứng dụng cần thông báo cho người dùng biết các chương mới cập nhật của truyện có trong danh sách yêu thích.

Phân tích yêu cầu:

- **Chức năng liên quan đến tài khoản người dùng:** đăng nhập, đăng ký, quên mật khẩu, đổi avatar, đổi tên hiển thị, đăng xuất, xóa tài khoản.
- **Chức năng xem danh sách truyện:** hiển thị danh sách truyện mới cập nhật, truyện được yêu thích nhiều nhất.
- **Chức năng xem truyện:** hiển thị thông tin truyện, thêm/xóa danh sách truyện yêu thích, đọc truyện từ chương đầu / chương mới nhất, xem danh sách bình luận của truyện, thêm/sửa/xóa bình luận cuả mình.
- **Chức năng đọc truyện:** xem danh sách ảnh của chương truyện muốn đọc, chuyển sang đọc chương trước/sau, chọn chương trong danh sách chương của truyện để đọc.
- **Chức năng xem tủ sách:** hiển thị danh sách lịch sử truyện có chương gần nhất đã đọc và danh sách truyện yêu thích.
- **Chức năng thông báo:** hiển thị thông báo các chương mới cập nhật của truyện có trong danh sách yêu thích.
- **Chức năng đăng truyện, chương:** đăng truyện, chương mới của chính user.

2.2. Thiết kế hệ thống:

2.2.1. Thiết kế kiến trúc

i. Bảng "users": Chứa danh sách tài khoản người dùng

Tên trường	Kiểu dữ liệu	Khoá	Mô tả
id	String	Chính	Mã người dùng
username	String		Tên người dùng
email	String		Địa chỉ email
password	String		Mật khẩu
avt_url	String		Đường link avatar
created_at	String		Thời gian tạo

ii. Bảng "authors": Chứa danh sách tác giả

Tên trường	Kiểu dữ liệu	Khoá	Mô tả

id	String	Chính	Mã tác giả
name	String		Tên tác giả

iii. Bảng “genre”: Chứa danh sách thể loại

Tên trường	Kiểu dữ liệu	Khoá	Mô tả
id	String	Chính	Mã thể loại
name	String		Tên thể loại

iv. Bảng “manga”: Chứa danh sách truyện

Tên trường	Kiểu dữ liệu	Khoá	Mô tả
id	String	Chính	Mã truyện
title	String		Tiêu đề truyện
authorId	String		Tên tác giả
imageUrl	String		Ảnh bìa truyện
description	String		Mô tả truyện
created_at	Long		Thời gian tạo

v. Bảng “chapters”: Chứa danh sách chương truyện

Tên trường	Kiểu dữ liệu	Khoá	Mô tả
id	String	Chính	Mã chương truyện
order	String		Số thứ tự chương truyện
manga_id	String		Mã truyện
pdf_url	String		File pdf chương truyện
created_at	Long		Thời gian tạo

vi. Bảng “favorites”: Chứa danh sách truyện yêu thích của người dùng

Tên trường	Kiểu dữ liệu	Khoá	Mô tả
id	String	Chính	Mã danh sách yêu thích
manga_id	String		Mã truyện

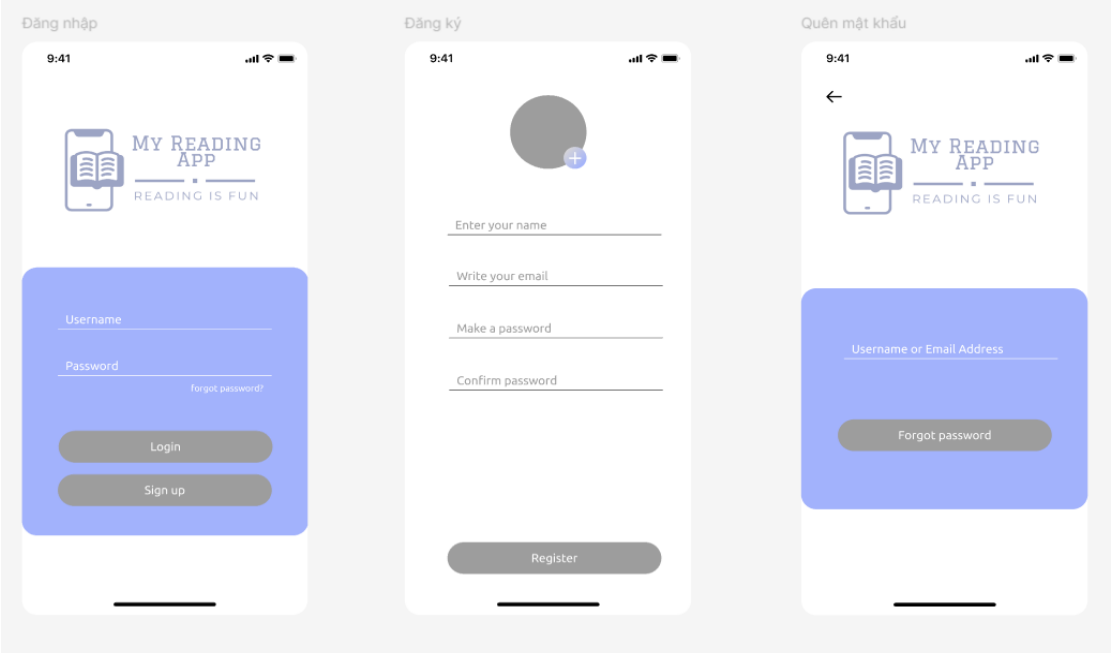
user_id	String		Mã người dùng
created_at	Long		Thời gian tạo

vii. Bảng “history”: Chứa danh sách truyện đã đọc của người dùng

Tên trường	Kiểu dữ liệu	Khoá	Mô tả
id	String	Chính	Mã danh sách lịch sử
user_id	String		Mã người dùng
chapter_id	String		Mã chương
last_date	Long		Thời gian đọc cuối

2.2.2. Thiết kế giao diện

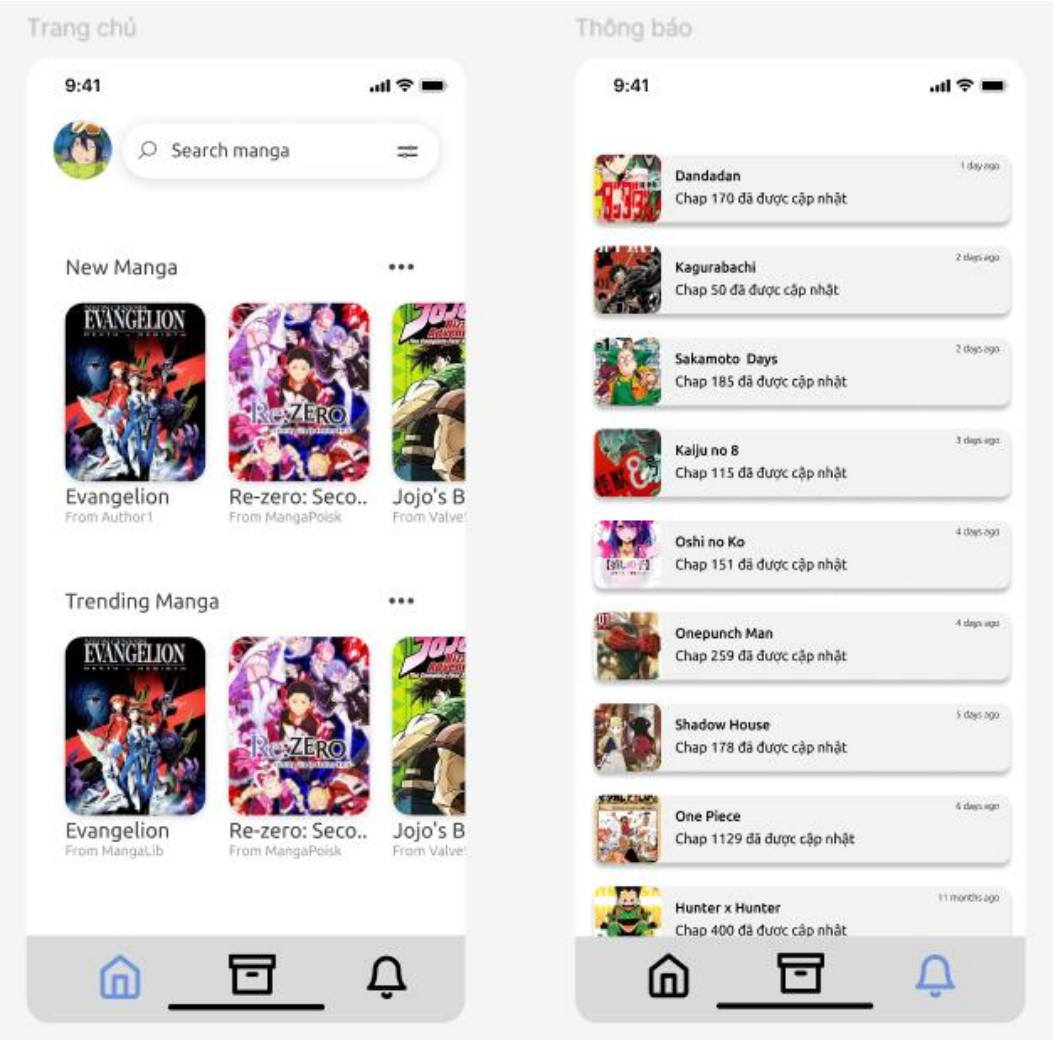
a. Người dùng



a) b) c)

Hình 1

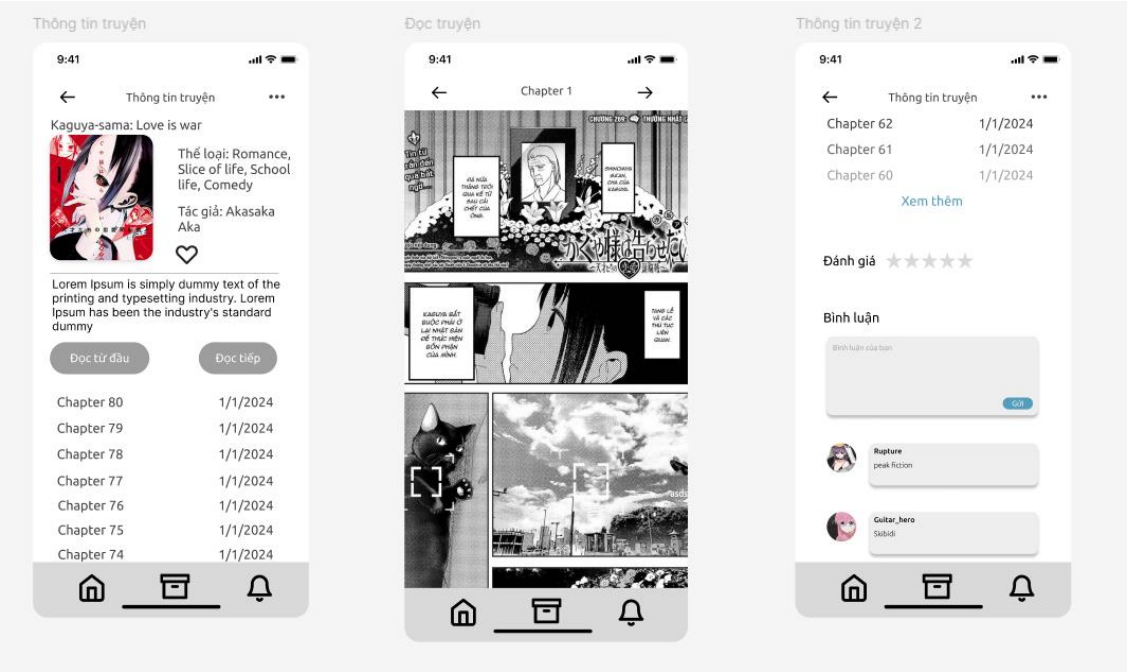
a) Giao diện Đăng nhập tài khoản của người dùng
b) Giao diện Đăng ký tài khoản của người dùng
c) Giao diện Quên mật khẩu của người dùng



a) b)

Hình 2

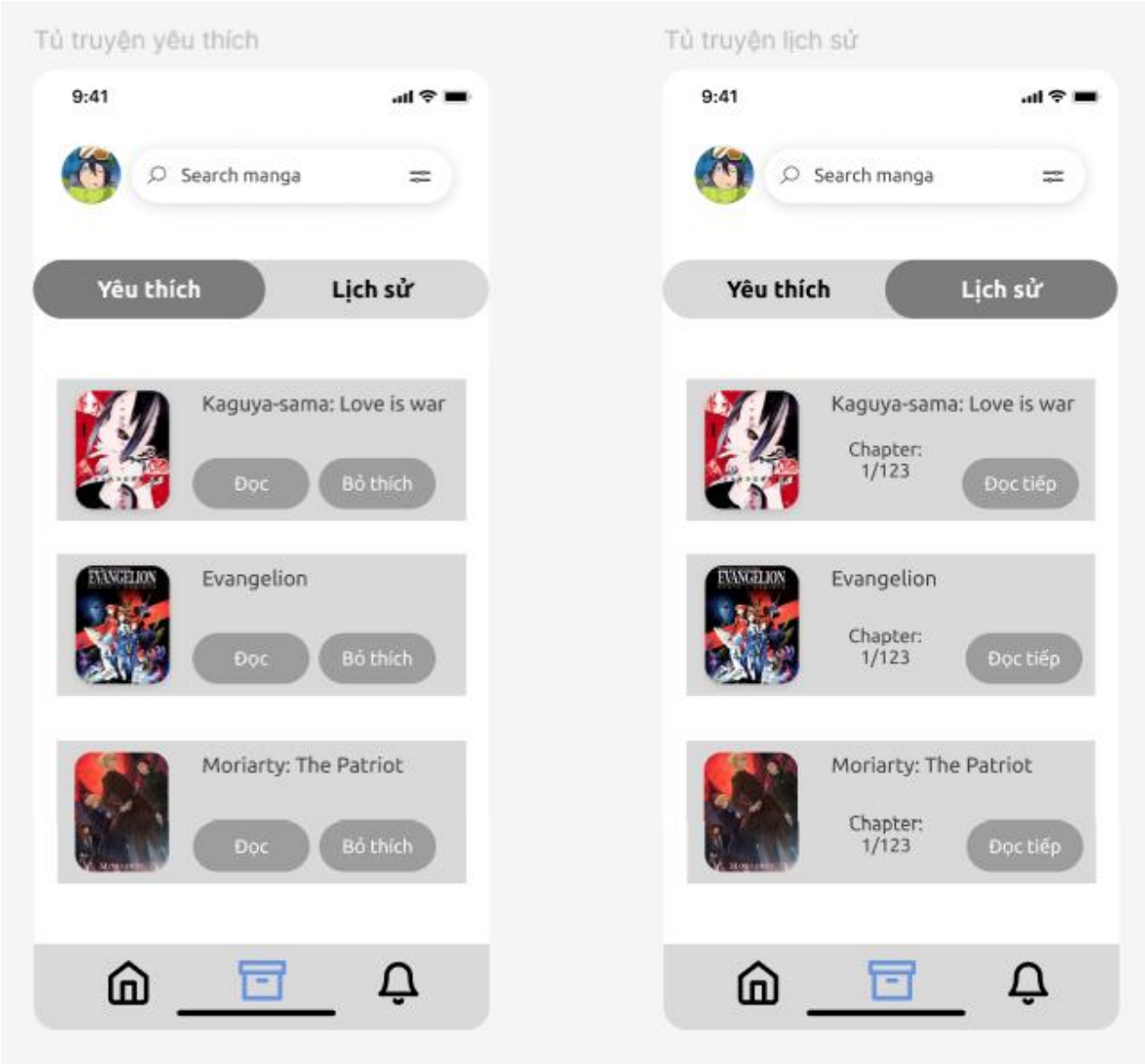
a) Giao diện trang chính (sau khi đăng nhập tài khoản hoặc khi nhấn nút bên trái ngoài cùng ở đáy giao diện)
b) Giao diện trang thông báo (sau khi nhấn nút bên phải ngoài cùng ở đáy giao diện)



a) b) c)

Hình 3

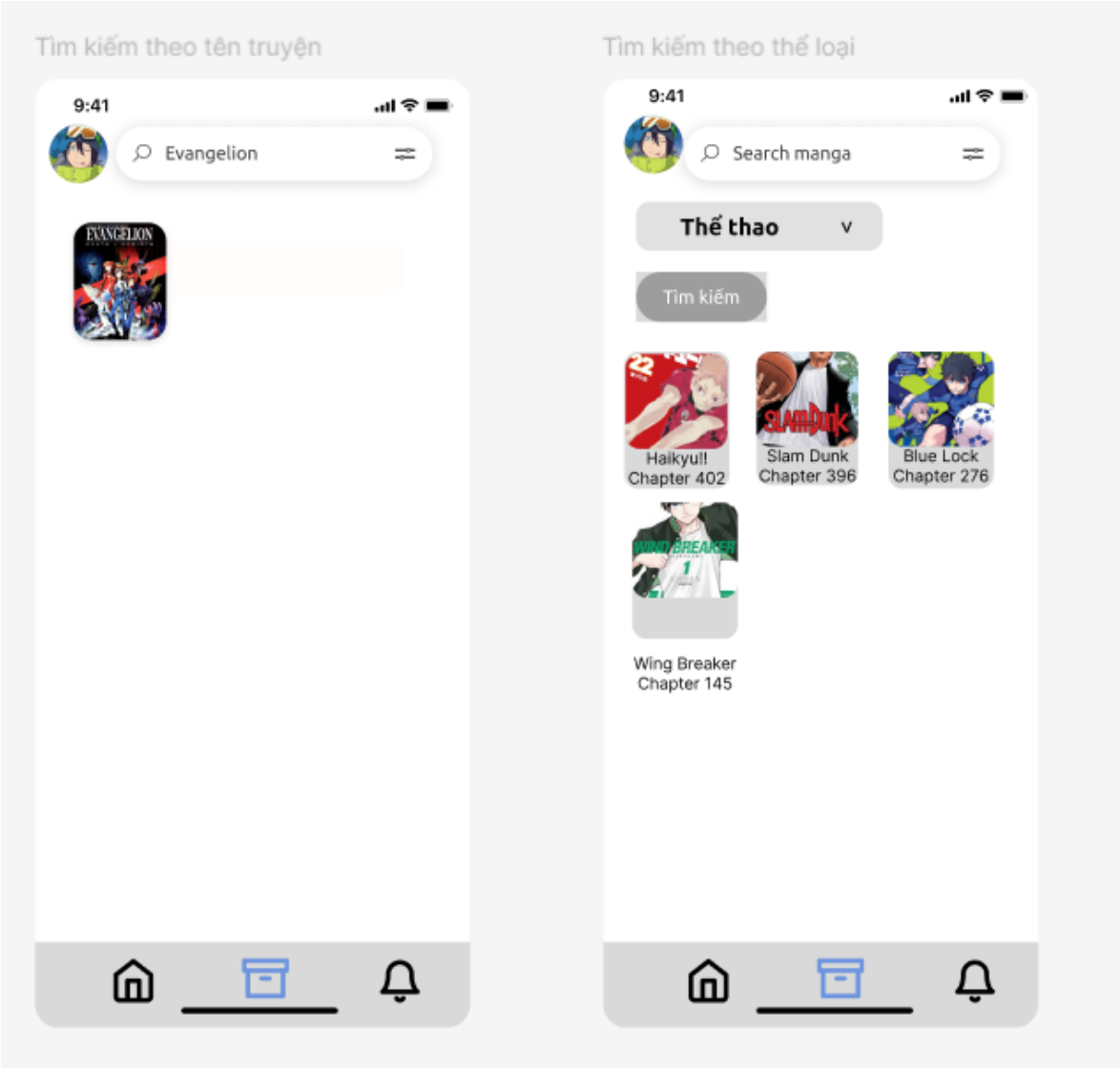
a) Giao diện thông tin chi tiết truyện (sau khi bấm chọn vào 1 truyện)
b) Giao diện đọc truyện (sau khi chọn vào chương để đọc)
c) Giao diện chuyển chương với bình luận sau khi bấm mũi tên trên cùng phải



a) b)

Hình 4: Giao diện tủ sách (sau khi nhấn vào nút thứ 2 từ trái sang ở đáy giao diện) bao gồm:

a) Giao diện hiển thị danh sách chương truyện đã đọc
b) Giao diện hiển thị danh sách truyện yêu thích



a) b)

Hình 5

a) Giao diện

2.3. Giai đoạn lập trình

2.3.1. Công nghệ sử dụng

1. Ngôn ngữ sử dụng: Java
2. Nền tảng: Android
3. Phần mềm xây dựng: Android Studio

2.3.2. Chức năng chính

Chương 3. KẾT QUẢ VÀ ĐÁNH GIÁ

3.1. Đạt được

3.2. Hạn chế

3.3. Hướng phát triển tương lai

Chương 4. PHỤ LỤC

4.1. Biểu đồ

4.2. Mã nguồn

Biểu đồ lớp cho mô hình miền:

2.2.1. Các thực thể chính và mối quan hệ:

- Manga: Đại diện cho các bộ truyện manga. Mỗi manga có thể có nhiều chương (Chapter) và thuộc nhiều thể loại (Category).
    + id: String
    + title: String
    + description: String
    + author: String
    + coverImage: String (URL của ảnh bìa)

- Chapter: Đại diện cho từng chương của một bộ manga. Một manga có thể có nhiều chương.
    + id: String
    + chapterNumber: int
    + pages: List<String> (Danh sách URL của các trang ảnh)
    + releaseDate: Date

- User: Đại diện cho người dùng ứng dụng, bao gồm thông tin đăng nhập và các thuộc tính liên quan.
    + id: String
    + username: String
    + email: String
    + password: String
    + favoriteManga: List<Manga> (Danh sách manga yêu thích)
    + readingHistory: List<Chapter> (Danh sách các chapter đã đọc)

- Category: Đại diện cho thể loại của manga (ví dụ: hành động, phiêu lưu, hài hước,...). Một manga có thể thuộc nhiều thể loại và một thể loại có thể chứa nhiều manga.
    + id: int
    + name: String
    + description: String

- Rating: Đại diện cho đánh giá của người dùng đối với một bộ manga. Một người dùng có thể đánh giá nhiều manga, và mỗi manga có thể nhận được nhiều đánh giá từ các người dùng khác nhau.
    + id: int
    + user: User
    + manga: Manga
    + score: int (thang điểm 1-5)

Mối quan hệ giữa các lớp:

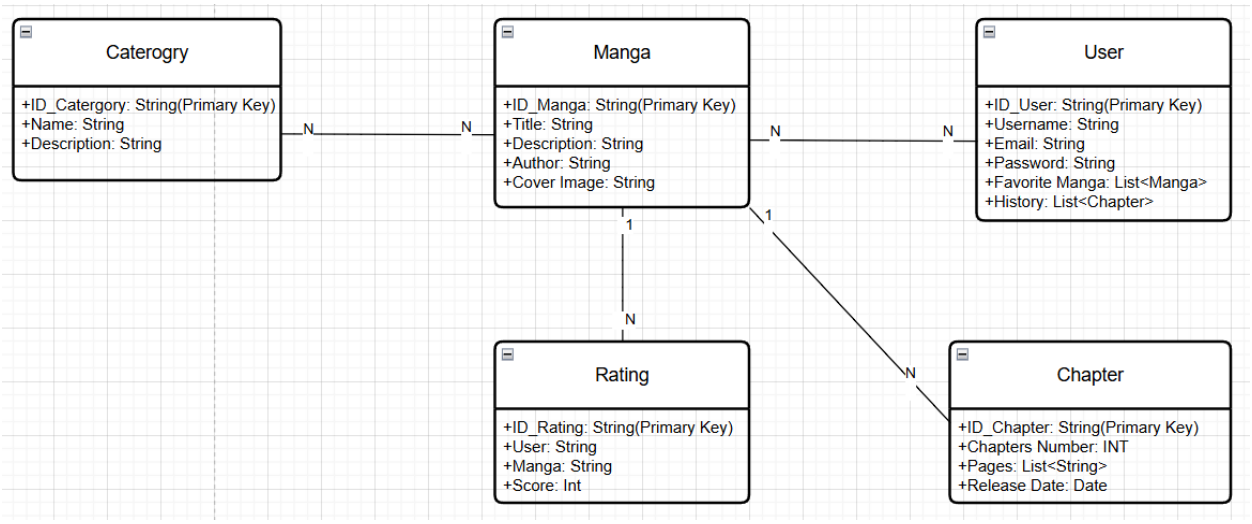
Manga 1 --- N Chapter: Một manga có nhiều chapter.

Manga n --- n Category: Một manga có thể thuộc nhiều thể loại và một thể loại có thể chứa nhiều manga.

User 1 --- n Rating: Một người dùng có thể đánh giá nhiều manga.

Manga 1 --- n Rating: Một manga có thể nhận nhiều đánh giá từ các người dùng.

User n --- n Manga: Người dùng có thể yêu thích nhiều manga (Favorite).



Biểu đồ lớp cho MVC:

Thiết kế kiến trúc (MVC):

- **Model:**
  - Các lớp đại diện cho dữ liệu như Manga, Chapter, Comment, User, v.v.
  - Các lớp truy xuất dữ liệu từ Firebase Realtime Database và Firebase Storage như MangaModel, ChapterModel, CommentModel, UserModel…
  - Đóng vai trò quản lý dữ liệu và logic nghiệp vụ liên quan đến dữ liệu.

- **View:**
  - Activity hiển thị danh sách truyện tranh.

- Activity hiển thị chi tiết truyện tranh.
- Activity hiển thị giao diện đọc truyện.
- Activity hiển thị phần bình luận.
- Activity dành cho người dùng, bình luận.
- Sử dụng các thành phần giao diện như RecyclerView để hiển thị dữ liệu.

- **Controller:**
  - MangaController: Xử lý các sự kiện và logic liên quan đến quản lý truyện tranh (lấy danh sách, chi tiết, cập nhật trạng thái đã đọc).
  - ChapterController: Xử lý các sự kiện và logic liên quan đến quản lý chương truyện (lấy nội dung chương, điều khiển navigation).
  - CommentController: Xử lý các sự kiện và logic liên quan đến quản lý bình luận (lấy danh sách, thêm/sửa/xóa).
  - UserController: Xử lý các sự kiện và logic liên quan đến quản lý người dùng (đăng nhập, đăng ký, cập nhật thông tin).

Dựa trên kiến trúc MVC đã chọn, ta có thể xác định các lớp sau:

- **Lớp Task:**
  - **Thuộc tính:** ID, tiêu đề, mô tả, deadline, mức độ ưu tiên, trạng thái.
  - **Phương thức:**
    - Getters và setters cho các thuộc tính.
    - toString(): trả về chuỗi mô tả công việc.
- **Lớp TaskList:**
  - **Thuộc tính:** List<Task> tasks (danh sách các công việc).
  - **Phương thức:**
    - addTask(Task task): thêm công việc mới vào danh sách.
    - removeTask(int taskId): xóa công việc theo ID.

        - getTaskById(int taskId): tìm kiếm công việc theo ID.
        - updateTask(Task task): cập nhật thông tin công việc.
        - getAllTasks(): lấy danh sách tất cả công việc.
        - getTasksByStatus(String status): lọc công việc theo trạng thái.
        - getTasksByPriority(String priority): lọc công việc theo mức độ ưu tiên.
        - getTasksByDeadline(Date deadline): lọc công việc theo deadline.

- **Lớp GoogleSheetHandler:**
    - **Thuộc tính:**
        - String spreadsheetId: ID của Google Sheet.
        - String sheetName: Tên của sheet chứa dữ liệu.
        - Sheets service: Đối tượng Sheets dùng để tương tác với Google Sheets API.
    - **Phương thức:**
        - readTasks(): đọc dữ liệu công việc từ Google Sheet và trả về List<Task>.
        - writeTasks(List<Task> tasks): ghi danh sách công việc vào Google Sheet.
        - updateTask(Task task): cập nhật công việc trên Google Sheet.
        - deleteTask(int taskId): xóa công việc trên Google Sheet.
- **Lớp TaskView:**
    - **Phương thức:**
        - displayMenu(): hiển thị menu chính.
        - displayTasks(List<Task> tasks): hiển thị danh sách công việc.
        - getInput(String prompt): lấy input từ người dùng.
        - showMessage(String message): hiển thị thông báo.
- **Lớp TaskController:**
    - **Thuộc tính:**

        - TaskList taskList: đối tượng TaskList.
        - GoogleSheetHandler sheetHandler: đối tượng GoogleSheetHandler.
        - TaskView taskView: đối tượng TaskView.
    - **Phương thức:**
        - handleUserChoice(int choice): xử lý lựa chọn của người dùng từ menu.
        - addNewTask(): thêm công việc mới.
        - displayTasks(): hiển thị danh sách công việc.
        - markTaskAsCompleted(): đánh dấu công việc hoàn thành.
        - editTask(): chỉnh sửa công việc.
        - deleteTask(): xóa công việc.

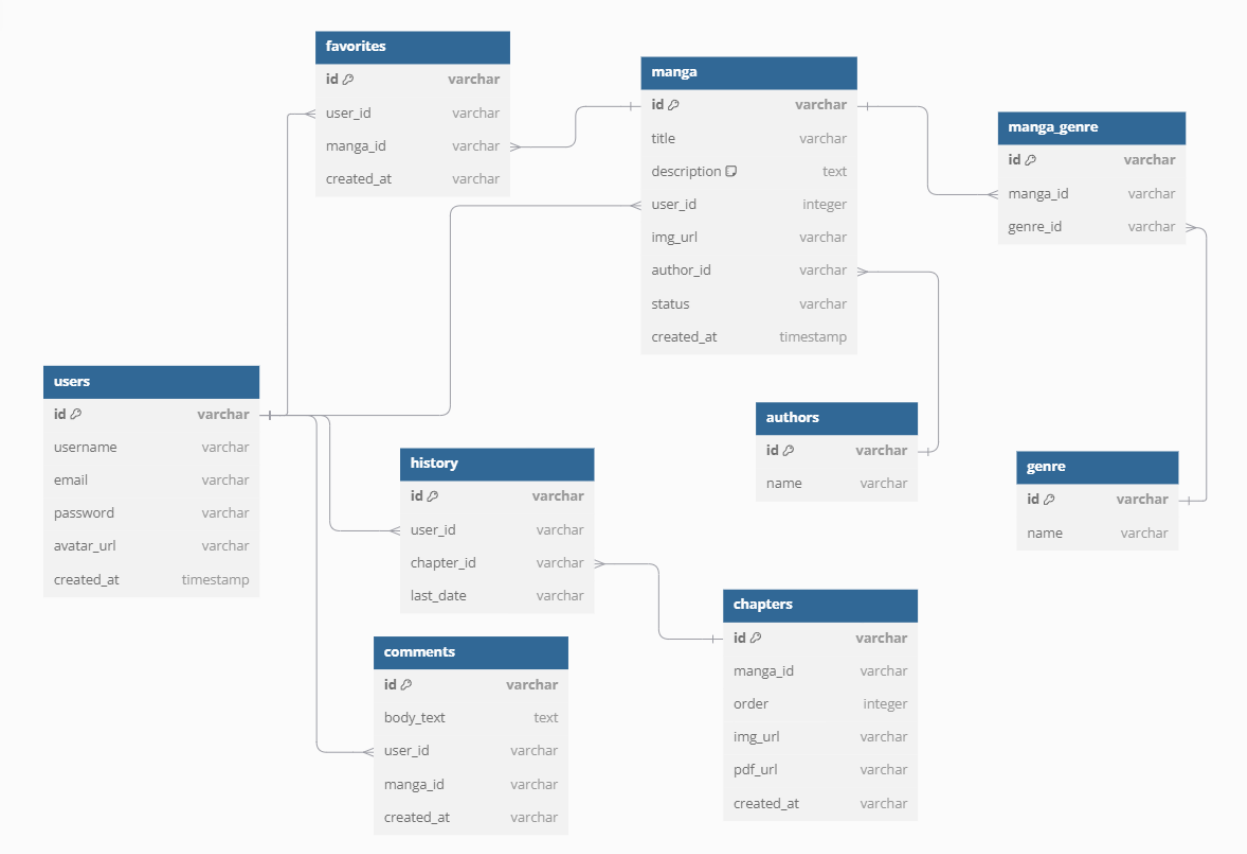
Mối quan hệ giữa các lớp:

- TaskController sử dụng TaskList và GoogleSheetHandler để quản lý và lưu trữ dữ liệu.
- TaskController sử dụng TaskView để tương tác với người dùng.
- TaskList chứa danh sách các đối tượng Task.

Lưu ý: Biểu đồ lớp có thể được vẽ chi tiết hơn bằng cách sử dụng UML (Unified Modeling Language) để thể hiện rõ ràng các thuộc tính, phương thức, và mối quan hệ giữa các lớp.

<<Chèn biểu đồ lớp cho mô hình MVC ở đây>>

Thiết kế cơ sở dữ liệu:



- Tạo một Google Sheet mới để lưu trữ dữ liệu.
- Xác định tên sheet và các cột tương ứng với các thuộc tính của công việc (ID, tiêu đề, mô tả, deadline, mức độ ưu tiên, trạng thái).

Thiết kế giao diện:

- Hiển thị menu chính với các lựa chọn:
    1. Thêm công việc mới
    2. Hiển thị danh sách công việc
    3. Đánh dấu hoàn thành công việc
    4. Chỉnh sửa công việc
    5. Xóa công việc
    6. Thoát
- Sử dụng các thông báo rõ ràng để hướng dẫn người dùng nhập liệu và hiển thị kết quả.

- Minh họa

```
===== TODO APP =====
1. Thêm công việc mới
2. Hiển thị danh sách công việc
3. Đánh dấu hoàn thành công việc
4. Chỉnh sửa công việc
5. Xóa công việc
0. Thoát
Chọn chức năng: 2

===== Danh sách công việc =====
ID  | Tiêu đề        | Deadline     | Ưu tiên | Trạng thái
-----------------------------------------------------------
1   | Học bài Java   | 2024-10-05   | Cao     | Chưa hoàn thành
2   | Viết báo cáo   | 2024-10-10   | Trung bình | Chưa hoàn thành
3   | Đọc sách       |              | Thấp    | Đã hoàn thành
```

2.3. Triển khai:

- **Viết code:** Sử dụng Java để cài đặt các class trong mô hình MVC, đọc/ghi file, xử lý dữ liệu và hiển thị giao diện console.
    - Cài đặt GoogleSheetHandler sử dụng thư viện API của Google Sheets (ví dụ: Google API Client Library for Java).
    - Xử lý xác thực để ứng dụng có quyền truy cập vào Google Sheet của bạn.
    - Thay đổi code trong TaskController để sử dụng GoogleSheetHandler cho việc đọc/ghi dữ liệu.
- **Kiểm thử:** Viết các test case để kiểm tra các chức năng của ứng dụng, đảm bảo ứng dụng hoạt động đúng theo yêu cầu.

2.4. Vận hành và bảo trì:

- Cài đặt và triển khai:
    - Hướng dẫn người dùng cách chạy ứng dụng từ console (ví dụ: java -jar TodoApp.jar).
    - Ngoài hướng dẫn chạy ứng dụng, cần hướng dẫn người dùng cách tạo Google Sheet và chia sẻ quyền truy cập cho ứng dụng.
    - Có thể cần cài đặt thêm các thư viện cần thiết cho việc kết nối với Google Sheets.

- Bảo trì: Sửa lỗi phát sinh, cập nhật chức năng mới (nếu có) và cải thiện hiệu năng của ứng dụng.

CHƯƠNG 3. KẾT QUẢ THỰC HIỆN

3.1. Công nghệ đã sử dụng

- Ngôn ngữ lập trình: Java
- Công cụ: IntelliJ IDEA
- Thư viện: Glide, PDF Viewer

3.2. Tiến độ thực hiện

Link github tới dự án:

https://github.com/NguyenVietSon45/-CSE441_MyReadingApp

Hướng dẫn các bước đã thực hiện:

B1. Tạo dự án mới:

- Mở IntelliJ IDEA.
- Chọn "Create New Project".
- Chọn "Java" làm ngôn ngữ lập trình.
- Chọn JDK phù hợp (ví dụ: JDK 11 hoặc mới hơn).
- Nhập tên dự án (ví dụ: "TodoApp").
- Chọn vị trí lưu trữ dự án.
- Nhấn "Finish" để tạo dự án.

B2. Tạo các package:

- Trong cửa sổ "Project", click chuột phải vào thư mục "src".
- Chọn "New" -> "Package".
- Tạo các package sau:
    - model: chứa các lớp Task và TaskList.
    - view: chứa lớp TaskView.
    - controller: chứa lớp TaskController.
    - handler: chứa lớp GoogleSheetHandler.

B3. Tạo các lớp:

- Trong mỗi package, click chuột phải và chọn "New" -> "Java Class" để tạo các lớp tương ứng.

- Cài đặt các thuộc tính và phương thức cho từng lớp dựa trên thiết kế đã phân tích.

B4. Cài đặt thư viện Google Sheets API:

- Mở file pom.xml (nếu bạn đang sử dụng Maven) hoặc build.gradle (nếu bạn đang sử dụng Gradle).
- Thêm dependency cho thư viện Google API Client Library for Java. Ví dụ, trong Maven:
- IntelliJ IDEA sẽ tự động tải về và thêm thư viện vào dự án.

B5. Viết code:

- Bắt đầu viết code cho từng lớp, thực hiện các chức năng của ứng dụng:
    - **Task:** cài đặt các thuộc tính và phương thức cơ bản.
    - **TaskList:** cài đặt các phương thức để quản lý danh sách công việc.
    - **GoogleSheetHandler:** cài đặt các phương thức để kết nối, đọc, ghi, cập nhật, xóa dữ liệu trên Google Sheet.
    - **TaskView:** cài đặt các phương thức để hiển thị giao diện console và tương tác với người dùng.
    - **TaskController:** cài đặt logic xử lý yêu cầu của người dùng, điều phối các lớp khác để hoàn thành các chức năng.

B6. Xử lý xác thực Google Sheets:

- Tạo tài khoản dịch vụ trên Google Cloud Platform và tải xuống file JSON chứa khóa API.
- Tham khảo tài liệu của Google Sheets API để biết cách sử dụng file JSON này để xác thực ứng dụng của bạn.
- Cài đặt code xử lý xác thực trong GoogleSheetHandler.

B7. Chạy và kiểm thử:

- Chạy ứng dụng từ IntelliJ IDEA bằng cách click chuột phải vào lớp Main (hoặc lớp chứa phương thức main) và chọn **"Run 'Main.main()'"**.
- Kiểm tra các chức năng của ứng dụng, sửa lỗi và hoàn thiện code.

B8. Triển khai (tùy chọn):

- Nếu muốn đóng gói ứng dụng thành file JAR để dễ dàng chia sẻ và chạy trên các máy khác, bạn có thể sử dụng chức năng "Build Artifacts" của IntelliJ IDEA.

3.3. Hình ảnh sản phẩm

<<Chụp hình Ảnh đưa vào đây>>

KẾT LUẬN

(Trình bày thành 3 đoạn văn nêu Ưu điểm, nhược điểm và hướng phát triển chủ đề)

DANH MỤC TÀI LIỆU THAM KHẢO

[1]. Nguyễn Hồng Sơn (2007), *Giáo trình hệ thống Mạng máy tính CCNA* (Semester 1), NXB Lao động xã hội.

[2]. Phạm Quốc Hùng (2017), *Đề cương bài giảng Mạng máy tính*, Đại học SPKT Hưng Yên.

[3]. James F. Kurose and Keith W. Ross (2013), *Computer Networking: A top-down approach sixth Edition*, Pearson Education.

PHỤ LỤC

- Xác thực: Cần có bước xác thực để ứng dụng có thể truy cập vào Google Sheet. Bạn có thể sử dụng OAuth2 để cho phép ứng dụng truy cập vào tài khoản Google của bạn.
- API Google Sheets: Tìm hiểu và sử dụng API của Google Sheets để thực hiện các thao tác đọc, ghi, cập nhật dữ liệu trên sheet.
- Hiệu năng: Tốc độ đọc/ghi dữ liệu có thể phụ thuộc vào tốc độ mạng và API của Google Sheets.
- Bảo mật: Cần bảo mật thông tin xác thực của bạn để tránh rò rỉ thông tin.
- **Code minh họa đầy đủ đã thực hiện**
    - **Task.java**

```java
package model;

import java.util.Date;

public class Task {
    private int id;
    private String title;
    private String description;
    private Date deadline;
    private String priority; // "Cao", "Trung bình", "Thấp"
    private boolean completed;

    // Constructor, getters và setters
    public Task(int id, String title) {
        this.id = id;
        this.title = title;
        this.description = "";
        this.deadline = null;
        this.priority = "Trung bình";
        this.completed = false;
    }

    // Getters and setters cho tất cả các thuộc tính

    @Override
    public String toString() {
        return "Task{" +
                "id=" + id +
                ", title='" + title + '\'' +
                ", description='" + description + '\'' +
                ", deadline=" + deadline +
                ", priority='" + priority + '\'' +
                ", completed=" + completed +
                '}';
    }
}
```

- **TaskList.java**

```
package model;

import java.util.ArrayList;
import java.util.List;

public class TaskList {
    private List<Task> tasks;

    public TaskList() {
        tasks = new ArrayList<>();
    }

    public void addTask(Task task) {
        tasks.add(task);
    }

    public void removeTask(int taskId) {
        tasks.removeIf(task -> task.getId() == taskId);
    }

    public Task getTaskById(int taskId) {
        for (Task task : tasks) {
            if (task.getId() == taskId) {
                return task;
            }
        }
        return null;
    }

    public void updateTask(Task task) {
        // Tìm kiếm công việc theo ID và cập nhật thông tin
    }

    public List<Task> getAllTasks() {
        return tasks;
    }

    // Các phương thức lọc công việc theo trạng thái, mức độ ưu tiên, deadline
}
```

- **TaskView.java**

```
package view;
import model.Task;
import java.util.List;
import java.util.Scanner;

public class TaskView {
    private Scanner scanner;

    public TaskView() {
        scanner = new Scanner(System.in);
```

```
    }

    public void displayMenu() {
        System.out.println("\n===== TODO APP =====");
        System.out.println("1. Thêm công việc mới");
        System.out.println("2. Hiển thị danh sách công việc");
        System.out.println("3. Đánh dấu hoàn thành công việc");
        System.out.println("4. Chỉnh sửa công việc");
        System.out.println("5. Xóa công việc");
        System.out.println("0. Thoát");
        System.out.print("Chọn chức năng: ");
    }

    public void displayTasks(List<Task> tasks) {
        System.out.println("\n===== Danh sách công việc =====");
        // ... (Code hiển thị danh sách công việc) ...
    }

    public String getInput(String prompt) {
        System.out.print(prompt);
        return scanner.nextLine();
    }

    public void showMessage(String message) {
        System.out.println(message);
    }
}
```

- **TaskController.java**

```
package controller;

import handler.GoogleSheetHandler;
import model.Task;
import model.TaskList;
import view.TaskView;

import java.io.IOException;
import java.security.GeneralSecurityException;
import java.util.List;

public class TaskController {
    private TaskList taskList;
    private GoogleSheetHandler sheetHandler;
    private TaskView taskView;

    public TaskController() {
        taskList = new TaskList();
        sheetHandler = new GoogleSheetHandler();
        taskView = new TaskView();
    }

    public void handleUserChoice(int choice) {
        switch (choice) {
            case 1:
                addNewTask();
```

```
                    break;
                case 2:
                    displayTasks();
                    break;
                // ... (Xử lý các lựa chọn khác) ...
                case 0:
                    exitApp();
                    break;
                default:
                    taskView.showMessage("Lựa chọn không hợp lệ!");
            }
        }

        public void addNewTask() {
            // ... (Code thêm công việc mới) ...
        }

        public void displayTasks() {
            try {
                List<Task> tasks = sheetHandler.readTasks();
                taskView.displayTasks(tasks);
            } catch (IOException | GeneralSecurityException e) {
                  taskView.showMessage("Lỗi khi đọc dữ liệu từ Google Sheet: " +
    e.getMessage());
            }
        }

        // ... (Các phương thức khác: markTaskAsCompleted, editTask, deleteTask) ...

        public void exitApp() {
            System.out.println("Thoát ứng dụng.");
            System.exit(0);
        }
    }
```

- **GoogleSheetHandle.java**

```
import com.google.api.client.auth.oauth2.Credential;
import
com.google.api.client.extensions.java6.auth.oauth2.AuthorizationCodeInstalledApp;
import com.google.api.client.extensions.jetty.auth.oauth2.LocalServerReceiver;
import com.google.api.client.googleapis.auth.oauth2.GoogleAuthorizationCodeFlow;
import com.google.api.client.googleapis.auth.oauth2.GoogleClientSecrets;
import com.google.api.client.googleapis.javanet.GoogleNetHttpTransport;
import com.google.api.client.json.JsonFactory;
import com.google.api.client.json.gson.GsonFactory;
import com.google.api.client.util.store.FileDataStoreFactory;
import com.google.api.services.sheets.v4.Sheets;
import com.google.api.services.sheets.v4.SheetsScopes;
import com.google.api.services.sheets.v4.model.*;

import java.io.FileNotFoundException;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.security.GeneralSecurityException;
```

```
import java.util.Arrays;
import java.util.List;

public class GoogleSheetHandler {
    private static final String APPLICATION_NAME = "Todo App";
    private static final JsonFactory JSON_FACTORY =
GsonFactory.getDefaultInstance();
    private static final String TOKENS_DIRECTORY_PATH = "tokens";

    private static final List<String> SCOPES =
            Arrays.asList(SheetsScopes.SPREADSHEETS_READONLY);
    private static final String CREDENTIALS_FILE_PATH = "/credentials.json";

    private static Credential getCredentials(final GoogleNetHttpTransport
HTTP_TRANSPORT)
            throws IOException {
        // Load client secrets.
        InputStream in =
GoogleSheetHandler.class.getResourceAsStream(CREDENTIALS_FILE_PATH);
        if (in == null) {
            throw new FileNotFoundException("Resource not found: " +
CREDENTIALS_FILE_PATH);
        }
        GoogleClientSecrets clientSecrets =
                GoogleClientSecrets.load(JSON_FACTORY, new
InputStreamReader(in));

        // Build flow and trigger user authorization request.
        GoogleAuthorizationCodeFlow flow = new
GoogleAuthorizationCodeFlow.Builder(
                HTTP_TRANSPORT, JSON_FACTORY, clientSecrets, SCOPES)
                .setDataStoreFactory(new FileDataStoreFactory(new
java.io.File(TOKENS_DIRECTORY_PATH)))
                .setAccessType("offline")
                .build();
        LocalServerReceiver receiver = new
LocalServerReceiver.Builder().setPort(8888).build();
        Credential credential = new AuthorizationCodeInstalledApp(flow,
receiver).authorize("user");
        //returns an authorized Credential object.
        return credential;
    }

    public void readData() throws IOException, GeneralSecurityException {
        // Build a new authorized API client service.
        final GoogleNetHttpTransport HTTP_TRANSPORT =
GoogleNetHttpTransport.newTrustedTransport();
        final String spreadsheetId = "YOUR_SPREADSHEET_ID"; // Thay bằng ID của
Google Sheet
        final String range = "Sheet1!A:F"; // Thay bằng tên sheet và phạm vi dữ
liệu
        Sheets service =
                new Sheets.Builder(HTTP_TRANSPORT, JSON_FACTORY,
getCredentials(HTTP_TRANSPORT))
                        .setApplicationName(APPLICATION_NAME)
                        .build();
        ValueRange response = service.spreadsheets().values()
                .get(spreadsheetId, range)
                .execute();
```

```
        List<List<Object>> values = response.getValues();
        if (values == null || values.isEmpty()) {
            System.out.println("No data found.");
        } else {
            System.out.println("Name, Major");
            for (List row : values) {
                // Print columns A and E, which correspond to indices 0 and 4.
                System.out.printf("%s, %s\n", row.get(0), row.get(4));
            }
        }
    }
}
```

QUY ĐỊNH TRÌNH BÀY TRONG BÁO CÁO CÀI TẬP LỚN

- Bài tập lớn được in trên một mặt giấy trắng khổ A4 (210 x 297mm), dày lớn hơn 30 trang, nhỏ hơn 100 trang, không kể hình vẽ, bảng biểu, đồ thị và danh mục tài liệu tham khảo.

- Phần nội dung trình bày trong bài tập lớn sử dụng Font chữ **Times New Roman** cỡ **13**, hệ soạn thảo Microsoft Word; mật độ chữ bình thường, không được nén hoặc kéo dãn khoảng cách giữa các chữ; dãn dòng đặt ở chế độ **1,5 lines**; lề trên **2,0 cm**; lề dưới **2,0 cm**; lề trái **2,5 cm**, lề phải **2,0 cm**. Số trang được đánh ở giữa, phía dưới trang giấy.

- Cách ghi trích dẫn tài liệu tham khảo: Cuối đoạn trích dẫn đánh số thứ tự tài liệu tham khảo (ví dụ: [1]: tham khảo tài liệu số 1; [3,4,8]: tham khảo 3 tài liệu số 3, 4, 8).

- Tuyệt đối không được tẩy, xoá, sửa chữa trong bài tập lớn.

- Quy cách trình bày nội dung

Đề mục	**Cỡ chữ**	**Định dạng**	**Canh lề trang**
Tên chương	14	In hoa, đậm	Giữa
Tên tiểu mục mức 1	13	Chữ thường, đậm	Trái
Tên tiểu mục mức 2	13	Chữ thường, đậm, nghiêng	Trái
Tên tiểu mục mức 3	13	Đánh chỉ mục bằng chữ cái thường a), ... b), ...	Trái
Nội dung	13	Chữ thường (Normal)	Đều hai bên
Nội dung bảng (table)	12	Normal	Giữa ô
Tên bảng	12	Chữ thường, nghiêng	Giữa, trên bảng
Tên hình	12	Chữ thường, nghiêng	Giữa, dưới hình
Tài liệu tham khảo	12	APA style	Chú thích bên dưới

Cách đánh dấu câu:

Các dấu: : , . ;) }] ! ? ” được gõ ngay sau ký tự cuối cùng (không khoảng cách), và gõ 1 phím cách (space) sau chúng. Sau các dấu: “{ ([không gõ dấu cách.

**** Cách đánh số các tiểu đề mục nhiều nhất là 3 mức và không lùi sang phải***

Hướng dẫn xếp tài liệu tham khảo

1. Tài liệu tham khảo xếp theo thứ tự trích dẫn trong bài tập lớn.
2. Tài liệu tham khảo là sách, luận án, báo cáo phải ghi đầy đủ các thông tin sau:
    - Tên các tác giả hoặc cơ quan ban hành (không có dấu ngăn cách)
    - (năm xuất bản), (đặt trong ngoặc đơn, dấu phẩy sau ngoặc đơn)
    - Tên sách, luận án hoặc báo cáo, (in nghiêng, dấu phẩy cuối tên)
    - Nhà xuất bản, (dấu phẩy cuối tên nhà xuất bản)
    - Nơi xuất bản, (dấu chấm kết thúc tài liệu tham khảo).

Tài liệu tham khảo là bài báo trong tạp chí, bài trong một cuốn sách… ghi đầy đủ các thông tin sau:

- Tên các tác giả (không có dấu ngăn cách);
- (Năm công bố), (đặt trong ngoặc đơn, dấu phẩy sau ngoặc đơn)
- "Tên bài báo", (đặt trong ngoặc kép, không in nghiên, dấu phẩy cuối tên)
- Tên tạp chí hoặc tên sách, (in nghiên, dấu phẩy cuối tên)
- Tập (không có dấu ngăn cách)
- (Số), (đặt trong ngoặc đơn, dấu phẩy sau ngoặc đơn)
- Các số trang, (gạch ngang giữa hai chữ số, dấu chấm kết thúc)

Cần chú ý những chi tiết về trình bày nêu trên. Nếu tài liệu dài hơn một dòng thì nên trình bày sau cho từ dòng thứ hai lùi vào so với dòng thứ nhất 1 cm để phần tài liệu tham khảo được rõ ràng và dễ theo dõi.

Ví dụ:

[4]. Nguyễn Hồng Sơn (2007), *Giáo trình hệ thống Mạng máy tính CCNA* (Semester 1), NXB Lao động xã hội.

[5]. Phạm Quốc Hùng (2017), *Đề cương bài giảng Mạng máy tính*, Đại học SPKT Hưng Yên.

[6]. James F. Kurose and Keith W. Ross (2013), *Computer Networking: A top-down approach sixth Edition*, Pearson Education.