CS3203 Test 2 (AY20/21 Sem 2)

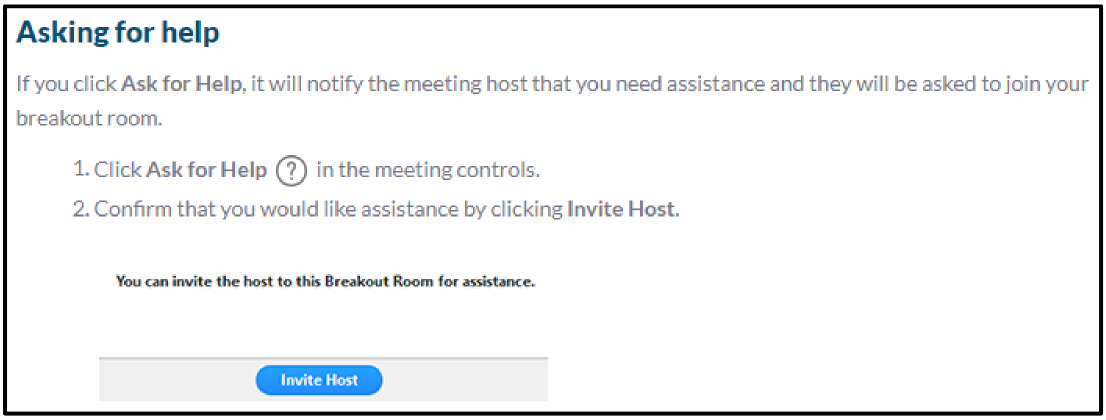
Tuesday, 30 Mar, 2pm

[Part II]

Duration: 30 minutes   Max Marks: 22

Part II Instructions:

1. Part II is a team component, where all team members work together and make only **ONE** (1) submission.
2. Submit your answers as a PDF file on **LumiNUS Files > Student Submissions > Test 2 Part II** by 30 Mar, within 30 minutes of start of part-2.
   a. Name your PDF as **`TeamXX.pdf`**, where **XX** is your team number e.g. **`Team06.pdf`**.
   b. If you are scanning your answers, make sure that it can be read once included in the PDF. Illegible answers will not be graded.
   c. We will only grade your most recent submission as a team.
3. Any clarifications should be made in LumiNUS Forums. Should you need an invigilator in your breakout room, you can use the Ask for Help feature in Zoom.



Asking for help

If you click **Ask for Help**, it will notify the meeting host that you need assistance and they will be asked to join your breakout room.

1. Click **Ask for Help** ? in the meeting controls.
2. Confirm that you would like assistance by clicking **Invite Host.**

You can invite the host to this Breakout Room for assistance.

[22] Component Design & Component Interactions

Context & Requirements

The popularity of customizable mechanical keyboards has been rising over the past few years. However, there are a lot of different vendors manufacturing different parts (keyboard case, mechanical switches, keycaps etc.) that makes up a customizable mechanical keyboard.

Traditionally, as these parts are not produced in mass, vendors will conduct an **Interest Check** to check if there is enough interest to start manufacturing the parts. Once there is sufficient interest, vendors will choose to conduct a **Group Buy**, where a duration is designated for users to participate by paying upfront. Then, vendors will start manufacturing the parts after the duration ends.

Your team is tasked to develop an application that provides a one-stop service for vendors to post **Interest Checks** and **Group Buys** listings, and for users to indicate their interest during **Interest Checks** and participate in **Group Buys**. The application requires the following features:

(i) Each Vendor and User will have a unique identifier and email address. Application can authenticate Vendors and Users via identifiers.

(ii) Application should allow Vendors to post an Interest Check listing. A listing contains description about the keyboard part, pictures, and its estimated manufacturing duration.

(iii) Application should allow Users to indicate their interest for a successful listing made by a Vendor. Application should also allow Vendors to check only the number of Users interested in their listing.

(iv) Application should allow Vendors to convert their Interest Check listing into a Group Buy listing. Vendors should provide a start date and end date for the Group Buy listing. Application should notify Users who were interested that the listing has been converted.

(v) Application should remove Interest Check listings automatically if it is not converted to a Group Buy listing after 6 months and notify the vendor via email.

(vi) Application should allow Users to participate in Group Buy listings in the duration of the Group Buy, regardless of whether they have indicated their interest previously. Application should also allow Vendors to check on the number of Users participating in the Group Buy, and exactly which Users participated.

(vii) Application should allow Vendors to update any of their Interest Check and Group Buy listings. Any updates to a listing will notify all the Users who were interested in the listing or have participated in the Group Buy via email. Users who were interested in a previous Interest Check listing, but did not participate in the Group Buy are notified as well.

Questions

1) Design the components for the application.

   a) **[8]** Based on the context and requirements, propose a high-level component diagram and justify your component design using appropriate design principles. Use design notations similar to SPA, and label all arrows appropriately. Write down any assumptions made.

   b) **[4]** Identify 2 design principles you followed with specific examples on where and how they are followed.

For question 2 & 3, you should only choose one question depending on your team size during the test. E.g. If you are in a 6-person team, and there is 1 absentee, your team should attempt Qns 2.

2) **[10] (For 4/5-person teams only)** Draw a sequence diagram that addresses the functionalities related to **Group Buy** listings (Feature (v), (vi), (vii)). For API calls within the sequence diagram, you do not need to state the parameters. However, the name and return type of each API should be unambiguous.

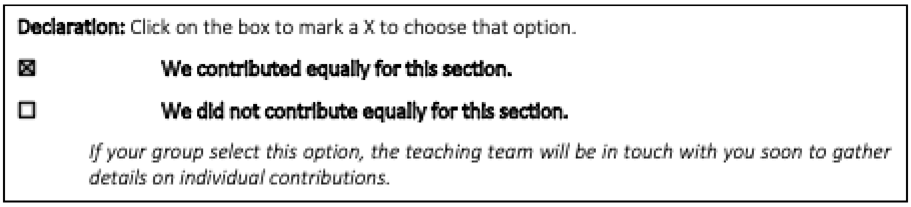
3) **[10] (For 6-person teams only)** Draw a sequence diagram that addresses the functionalities related to **Interest Check** listings (Feature (ii), (iii), (iv), (vii)). For API calls within the sequence diagram, you do not need to state the parameters. However, the name and return type of each API should be unambiguous.

SUBMISSION
CS3203 Test 2 (AY20/21 Sem 2)
Tuesday, 30 Mar, 2pm
[Part II]

Declaration

Write your names & matriculation no. of your team who have contributed to the submission.

No.	Name	Matriculation No.
1	Abdul Haliq	A0125431U
2	Guo Bohao	A0117428E
3	Wong Chuan Kai	A0136221W
4	Yap Dian Hao	A0184679H
5	Ho Zong Sien	A0171374E
6	Pakorn Uareeworakul	A0170861E



Declaration: Click on the box to mark a X to choose that option.

☒ **We contributed equally for this section.**

□ **We did not contribute equally for this section.**

If your group select this option, the teaching team will be in touch with you soon to gather details on individual contributions.

Assumptions :

If you made any assumptions about the context or requirements while attempting Question 1 or Question 2/3, clearly list here in this section.

Question 1

Write your answer for Question 1 below

1. Design the components for the application.

A) [8] Based on the context and requirements, propose a high-level component diagram and justify your component design using appropriate design principles. Use design notations similar to SPA, and label all arrows appropriately. Write down any assumptions made.

B) [4] Identify 2 design principles you followed with specific examples on where and how they are followed.

A)

USERS DB

- Each entry contains the type of the user (common user/vendor), unique identifier, email address, and other personal information required.

PRODUCT DB

- Each entry contains the type of the entry (interest check/group buy), description of the product, images url list, expected manufacturing duration, vendor identifier, group buy period (if applicable), and lists of participating users.

AuthenticationController

- Add and remove users from the USERS DB.
- Return credentials upon successful authentication given the users/vendors unique identifier.

ListingController

- Update the database upon user participation in group buy or interest check.
- Update/create listings upon vendor request.
- Return information such as number of participating user upon request from the view layer

CrontaskHandler

- Sweep through the database in fixed intervals to remove old interest check listings and notify vendors.

NotificationHandler

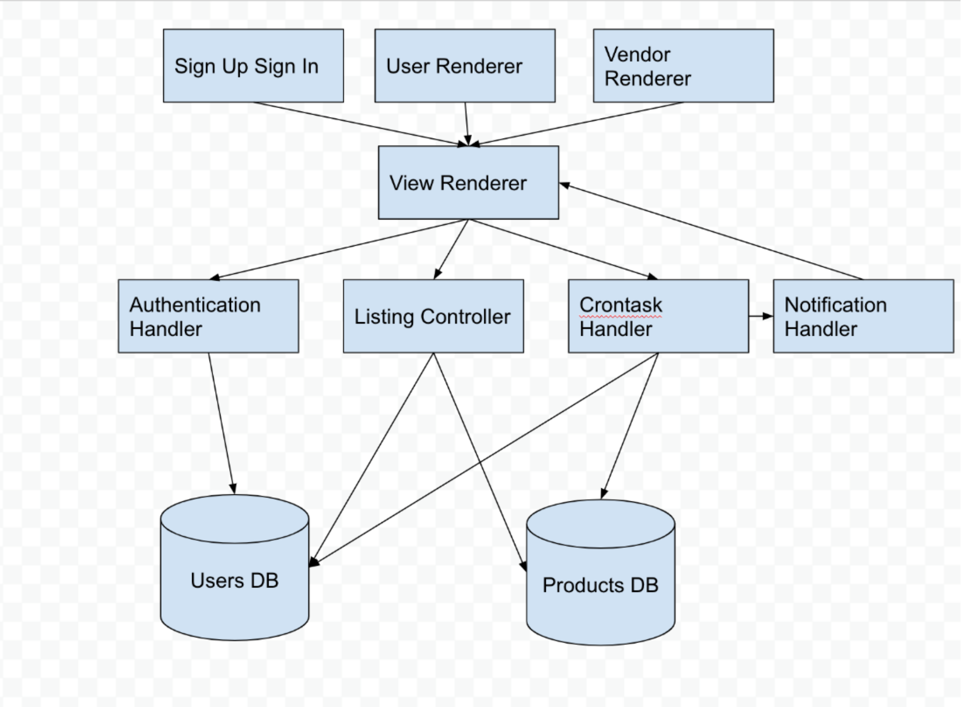
- Send notification to users/vendors upon events that requires them to be notified.

Vendor View

- Registered Product list Page
- Add New Product Listing Page
- Product View

User View

- Browse Products for interest list page
- Browse Products for group buy list page
- View products registered for interest page
- View products registered for group buy page
- Product View



Justification:

We base our design on the MODEL-VIEW-CONTROLLER design. By separating each component into MODEL, VIEW, CONTROLLER groups, it decreases coupling across component groups and increases cohesion within each component group. Our design adheres to the single responsibility principle by making sure each component is in charge of only one functionality of the program and each component group MODEL, VIEW, CONTROLLER is in charge of either data (model), logic (controller) or view (view) functionality of the program. We also use interface segregation principle to ensure the dependencies across components is reduced.

Model – userDB and productDB
View – view renderer
Controller – authentication handler, listing controller, Cron task handler, notification handler

Pros:

1. Easy to maintain if there is extension in future.
2. Better testing due to single responsibility and separation of concern.
3. Debugging is easier as we can more easily pinpoint the point of failure
4. Because each component does separate task, the components can be optimized.
5. Each developer can work on different component simultaneously and integrated after each component is completed..

Cons:

1. Need to do integration testing between components of MVC
2. More code is needed for separating into each component.

Overall:

MVC is good design model due to its diverse advantage in maintenance of code and testing.

B)

- Single Responsibility Principle: we made sure that every component has responsibility over a single part of the program’s functionality, such as Users DB for storing user data and Listing Controller for controlling the operations of the product listings.

- Interface Segregation Principle: we separated the application interface into 2 different interfaces: the vendor’s interface and the user’s interface so that they will only be exposed to methods that are related to them, such as indicating interest only for clients, and convert group buy listings only for vendors.

Question 2 / 3

Write your answer for Question 2 or 3 below.

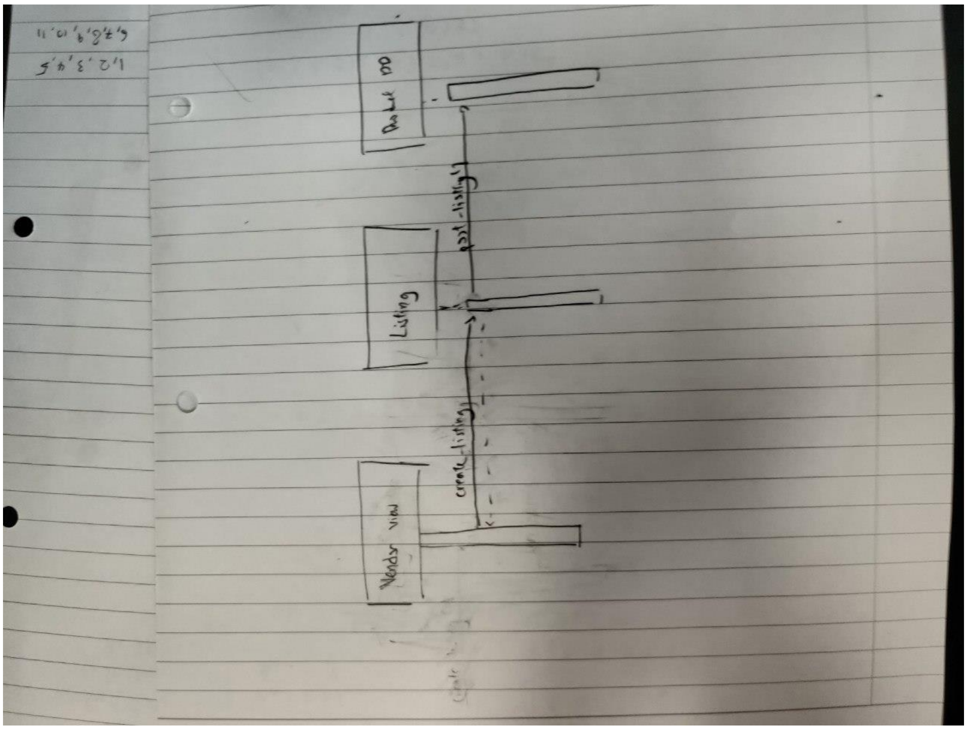
3. [10] **(For 6-person teams only)** Draw a sequence diagram that addresses the functionalities related to **Interest Check** listings (Feature (ii), (iii), (iv), (vii)). For API calls within the sequence diagram, you do not need to state the parameters. However, the name and return type of each API should be unambiguous.

Application should allow Vendors to post an Interest Check listing. A listing contains description about the keyboard part, pictures, and its estimated manufacturing duration.

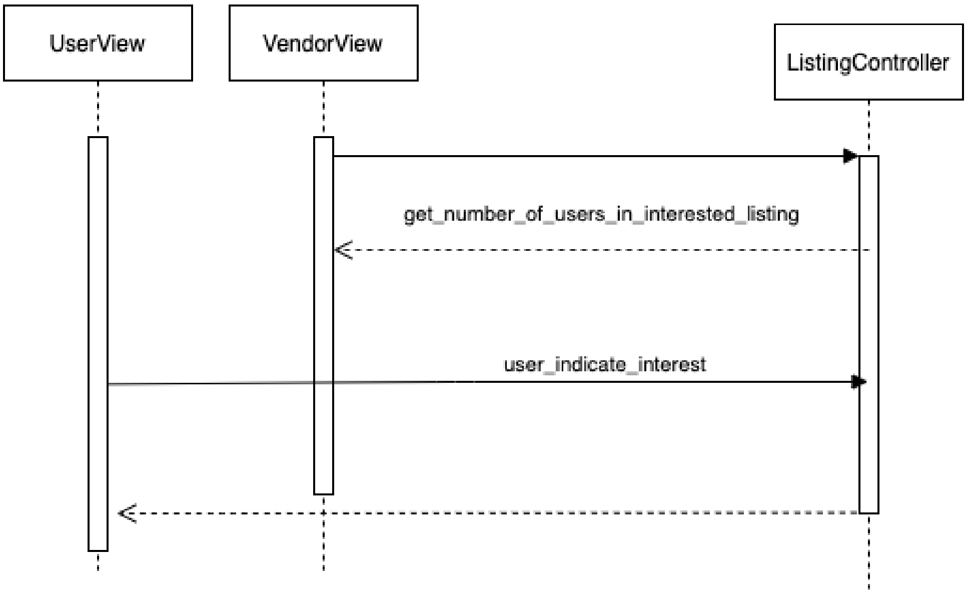
Application should allow Users to indicate their interest for a successful listing made by a Vendor. Application should also allow Vendors to check only the number of Users interested in their listing.

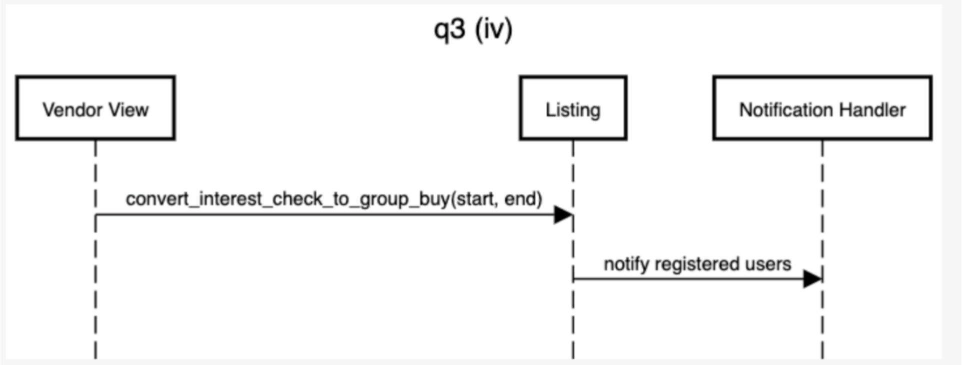
Application should allow Vendors to convert their Interest Check listing into a Group Buy listing. Vendors should provide a start date and end date for the Group Buy listing. Application should notify Users who were interested that the listing has been converted.

Q3 (ii)

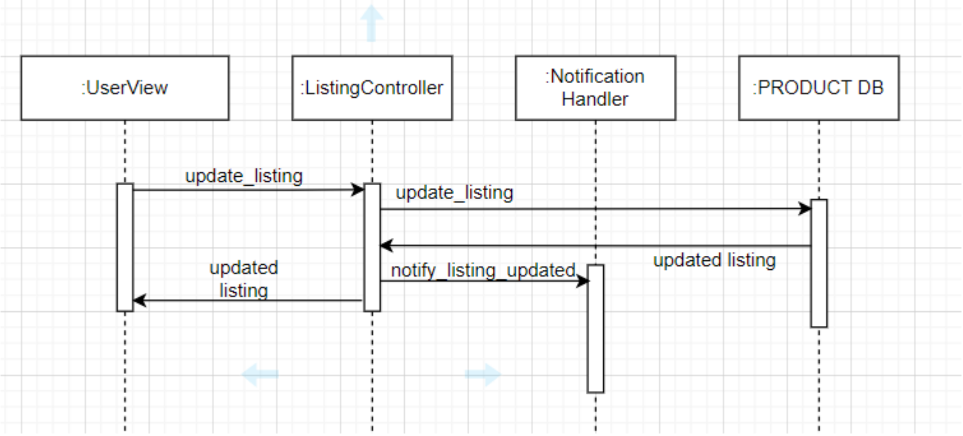


q3 (iii)





Question 3(vii)



Application should allow Vendors to update any of their Interest Check and Group Buy listings. Any updates to a listing will notify all the Users who were interested in the listing or have participated in the Group Buy via email. Users who were interested in a previous Interest Check listing but did not participate in the Group Buy are notified as well.

Sign Up Sign In

```
app sign_in(string email, string password)
app sign_up(string email, string password)
```

Listing Controller API

```
bool post_listing(listing_data data)
bool update_listing(listing_data data, listing_ref ref)
void user_indicate_interest(listing_ref ref)
int get_number_of_users_in_interested_listing(listing_ref ref)
Void remove_interested_listing(listing_ref ref)
List<listing_ref> convert_interest_check_to_group_buy (List<listing_ref>interest_check)
Void notify_users_in_listing(listing_ref ref)
Void notify_vendors()
List<listing_ref> get_vendor_interest_products(vendor_data data);
List<listing_ref> get_vendor_group_buy_products(vendor_data data);
List<listing_ref> get_user_interest_products(user_data data);
List<listing_ref> get_user_group_buy_products(user_data data);
```

Notification Handler

```
void notify_registered_users(listing_data data)
void notify_interest_expiry_to_vendor(product_data data)
Void notify_listing_updated
```

Cronhandler

```
Callback start_new_timer(int duration);
```

Vendor View

```
void show_number_interested_list(int)
bool create_listing(product_data data)
```

User View

```
Listing browse_interest_list(App app)
```

DB API

Bool add_listing_to_db(listing_ref ref)
Listing update_listing: update a listing and returns the updated listing