- a MongoDB database; you may use MongoDB from another cloud provider eg. MongoDB Atlas
- a service to deploy the Spring Boot application. The given Spring Boot is written in Java 19. Make sure that you configure Railway to use Java 19. See https://nixpacks.com/docs/providers/java.

S3 Bucket

Provision a new S3 bucket (or compatible) either in DigitalOcean, Amazon Web Services, Cloudflare, etc.

You may use an existing S3 bucket.

Angular Application

Generate an Angular application inside the `vttp2023-batch3-csf-assessment-template` directory.

Important: before generating the Angular application, initialise `vttp2023-batch3-csf-assessment-template` directory as a Git repository.

Assessment

In this assessment, you will be writing an application to track trending news by their hashtags. The frontend of the application is written in Angular and the backend is a Spring Boot application.

The application consist of 3 views

- View 0 - this is the 'landing page' of the application. This view shows the top 10 hashtags in the past 5, 15, 30, 45 or 60 minutes
- View 1 - the list of news for a specific hashtag in the past 5, 15, 30, 45 or 60 minutes
- View 2 - allows users to post news articles

The flow is shown in the following Figure 1

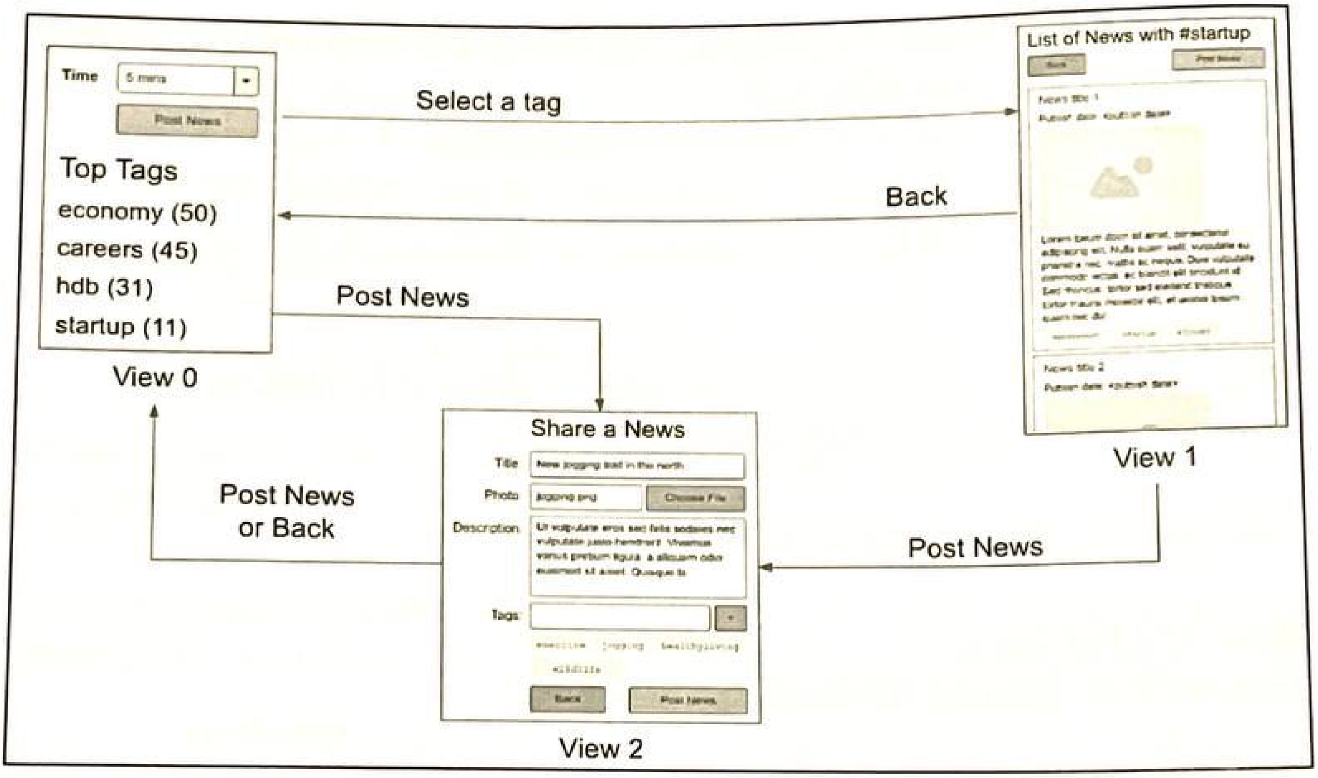


Figure 1 Application flow

Users can navigate between views by either

- Selecting a hashtag (view 0)
- Clicking on the Back or Post News button

Detailed description and behaviour of each view will be given in subsequent tasks.

The Spring Boot application uses 2 persistence stores to save news and related artifacts. They are

- MongoDB for saving the news. You are free to name the Mongo database.
- S3 for the images associated with the news article. You are free to name the S3 bucket

The following Figure 2 shows how Spring Boot uses these data stores.

To minimise cost, the deployment strategy is to have the Angular application served by the backend Spring Boot application.

Read all the required tasks first before attempting the assessment.

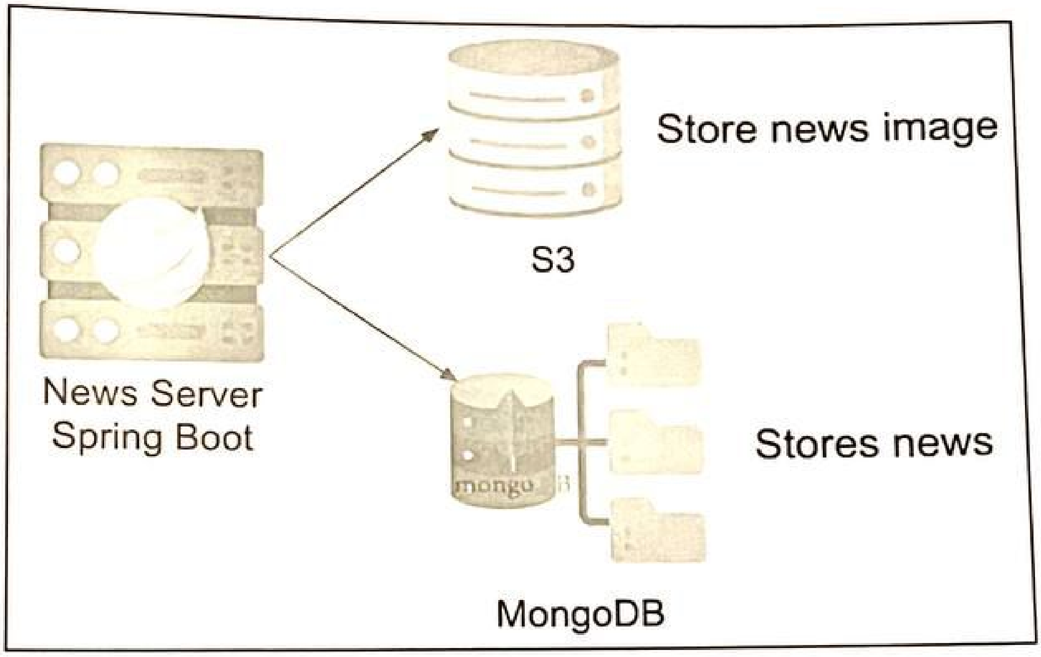


Figure 2 News server persistence store

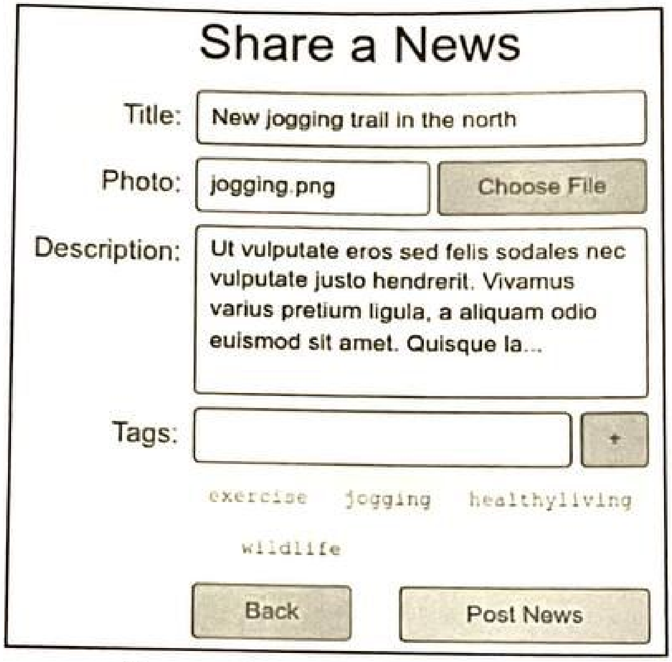
Task 1 (53 Marks)

Create View 2 to allow users to post news articles. The view should include the following form fields for capturing the news article details. These fields are shown in Table 1 below

Field	Description	Constraints
title	news title	mandatory, must be at least 5 characters in length
photo	photo	mandatory, photo is to be uploaded from the local computer
description	news description	mandatory, must be at least 5 characters in length
tags	a list of space separated words	optional

Table 1 List of fields in View 2

An example of View 2 is shown in Figure 2 below.



Share a News

Title: New jogging trail in the north

Photo: jogging.png

Description: Ut vulputate eros sed felis sodales nec vulputate justo hendrerit. Vivamus varius pretium ligula, a aliquam odio euismod sit amet. Quisque la...

Tags:

exercise jogging healthyliving wildlife

Figure 2 View 2 - Post a news

The hashtags are entered as a space separated list of words. When the 'Add Tag' button is pressed (the + button in Figure 2), the list is added to the hashtag list (displayed below the tags input field).

For example, entering the following space delimited string

```
exercise jogging healthyliving wildlife
```

will produce 4 hashtags: `exercise`, `jogging`, `healthyliving` and `wildlife`. Users can add as many tags as they wish. You can ignore duplicate hashtags.

You should also provide a way for users to remove unwanted tags, for example, with a delete button or by clicking on the hashtag.

When the Post Button is pressed, View 2 takes all the news information from the form and sends it to the backend Spring Boot application for processing.

The news image is saved to the S3 bucket.

The news article is save to a MongoDB collection called `news` in the following JSON document structure

```
{
      _id: ObjectId("..."),
      postDate: <long>,
      title: <string>,
      description: <string>,
      image: <string, url>,
      tags: [ <string>, <string>, ... ]
}
```

The following describes of the document fields

- `_id` - the news id saved as `ObjectId`
- `postDate` - news posting date at the time the document is inserted into the `news` collection. The time should be in the number of milliseconds since 12 AM Jan 1 1970. You can use `System.currentTimeMillis()` to get the time
- `title` - news title
- `description` - news description
- `image` - image URL of the news
- `tags` - an array of hashtags. Each hashtag is stored in the array as a string. If the news has no hashtags, this attribute should not be present

When the news is successfully inserted, return the `_id` from the document in the following JSON payload to the frontend Angular application

```
{ newsId: <_id in string> }
```

Display the returned news id in an `alert()` dialog; navigate to View 0 when the `alert()` is dismissed.

If the post news operation fails, return the error message. View 0 should display the error in an `alert()` dialog and remain in View 0.

The Post Button should be disabled and only enabled if all the mandatory fields are filled and their constraints met.

Add a Back button to navigate back to View 0.

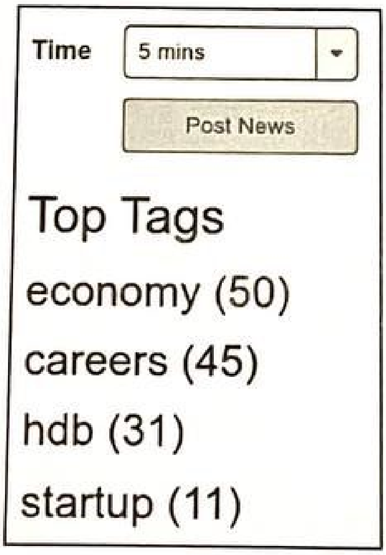
You may layout View 2 according to your preference but all the required fields must be present.

Implement the Spring Boot backend for Task 1 in the following classes: `NewsController`, `NewsService.postNews()`, `NewsRepository` and `ImageRepository`. You may create additional classes. Marks will be awarded for proper 'layering' of your implementation. Write the native Mongo query in the comments above the Java method in the repository class.

Use the S3 client is configured in the given `AppConfig` class. Do not change or modify the `AppConfig` class.

Task 2 (43 marks)

View 0 is the 'landing page' of the application. An example is shown in Figure 3 below.



Top Tags

economy (50)

careers (45)

hdb (31)

startup (11)

Figure 3 View 0 - Top news hashtags

The view shows the top 10 tags in the past 5, 15, 30, 45 and 60 minutes. The default duration is 5 minutes. This duration can be changed by selecting the drop down. Whenever the duration is changed eg from 30 to 5 minutes, View 0 will update all the hashtags of news that are posted within the last 5 minutes.

The top 10 hashtags are displayed along with the count, the number of times the hashtags appears in news posted within the duration. For example, in Figure 3, the hashtag 'hdb' appears in 31 news articles posted in the last 5 minutes.

A Post News button navigates to View 2.

Create View 0 as described above. You may layout View 0 according to your preference but all the required information must be present.

Write the Spring Boot request handler to process the request from View 0.

Implement the Mongo query to return the hashtags and its corresponding count from the `news` collection in a single query.

The query should

- be efficient by only querying news posited in the specified duration eg. last 5 minutes in Figure 3
- sort the tag count in descending order followed by the tag name in ascending order
- return only the top 10 tags
- tag names must not be empty ("") or null

The Mongo query should return the result in the following JSON structure

```
[
    { "tag": "economy", "count": 50 },
    { "tag": "careers", "count": 45 },
    { "tag": "hdb", "count": 31 },
    ...
]
```

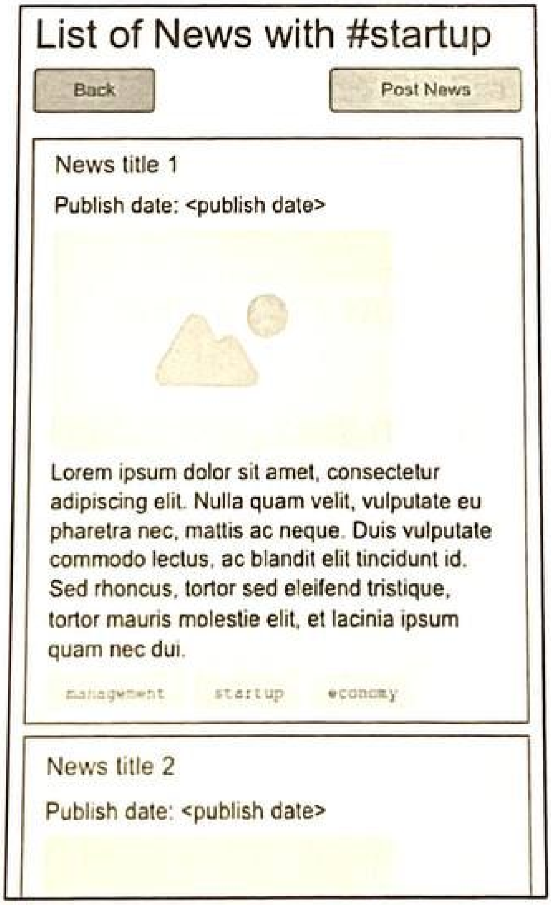
If the query over a duration returns no tags, then View 0 should display the message

No news hashtags found within the last <n> minutes

Implement the Spring Boot backend for Task 2 in the following classes: `NewsController`, `NewsService.getTags()` and `NewsRepository`. You may create additional classes. Marks will be awarded for proper 'layering' of your implementation. Write the native Mongo query in the comments above the Java method in the repository class.

Task 3 (25 marks)

When a user clicks on a hashtag in View 0, navigate to View 1 and display all the news of the selected hashtag within the duration; for example if View 0 is currently showing hashtags from the last 15 minutes and the user clicks on 'startup' hashtag, then View 1 should only display posted news from the last 15 minutes that has the selected hashtag viz. startup.


List of News with #startup
Back
Post News
News title 1
Publish date: <publish date>
management
startup
economy
News title 2
Publish date: <publish date>

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla quam velit, vulputate eu pharetra nec, mattis ac neque. Duis vulputate commodo lectus, ac blandit elit tincidunt id. Sed rhoncus, tortor sed eleifend tristique, tortor mauris molestie elit, et lacinia ipsum quam nec dui.

Figure 4 View 1 - Display news

An example of View 1 is shown in Figure 4. For each news article, display the following

- news title
- publish time and date
- news image
- description
- list of hashtags associated with the news

The news articles are displayed according to their publish date and time with the latest news displayed first viz. the news should be sorted in descending order of the posted date.

View 1 has 2 additional buttons; the Back button navigates back to View 0 and Post News navigates to View 2.

When navigating back to View 0 from View 1, ensure that the original state of View 0 is preserved; for example if the duration in View 0 was originally at 30 minutes when transitioning to View 1, then when transition back from View 1 to View 0, the duration should be set back to 30 minutes. The list of displayed hashtags should also be from this duration.

Implement the Spring Boot backend for Task 3 in the following classes: `NewsController`, `NewsService.getNewsByTag()` and `NewsRepository`. You may create additional classes. Marks will be awarded for proper 'layering' of your implementation. Write the native Mongo query in the comments above the Java method in the repository class.

Task 4 (10 marks)

Deploy your assessment to Railway. Remember that the Angular frontend should be served from the SpringBoot backend.

The databases configurations eg. password, etc. should not be expose either in `application.properties` or hard coded in the source coc Marks will be deducted if they are exposed.