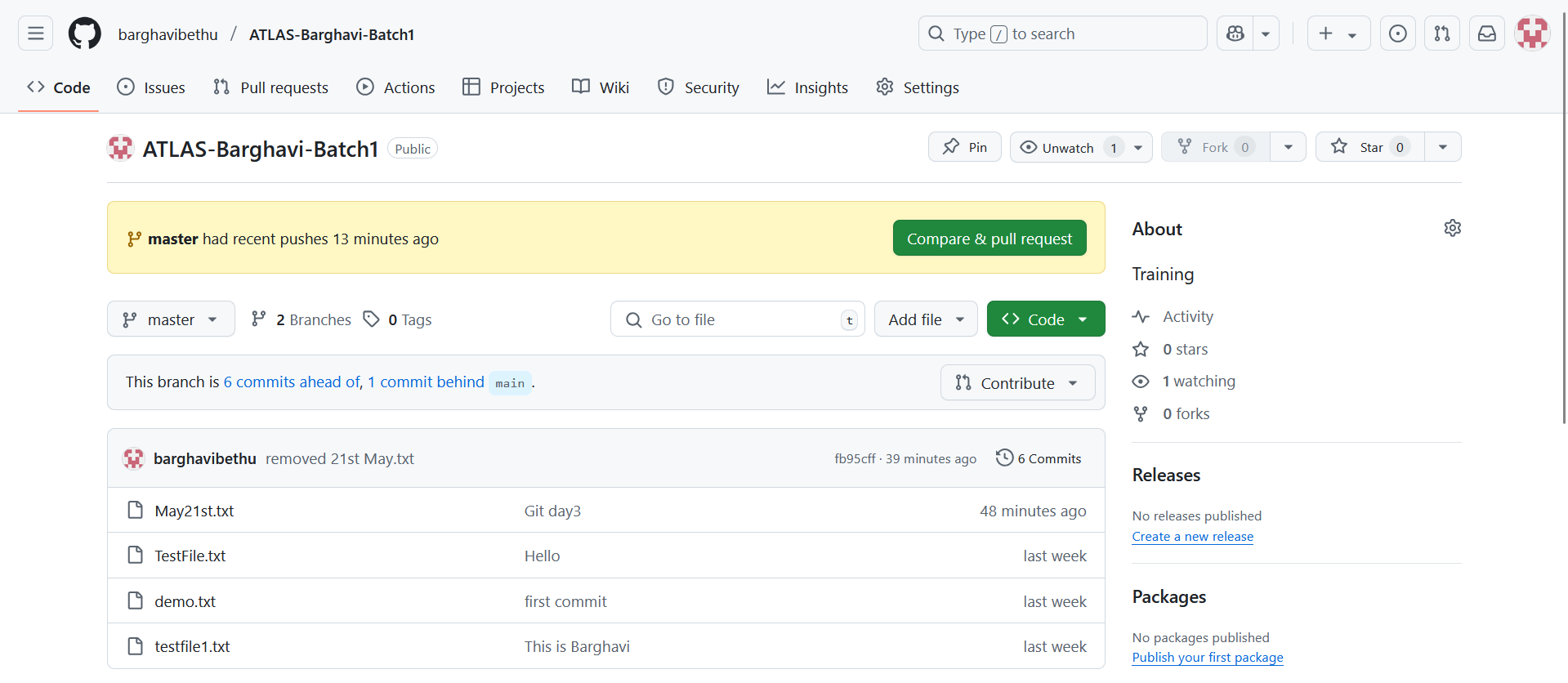
**Day 3 - 21/5/2025**

**Barghavi Bethu   
112728766 - bethubal**

**Task 1**

Create a file names 21st May.txt and push it to your git hub.



**Task 2**

Atomicity, Consistency, Isolation, Durability

**Atomicity:** All or Nothing.

it means that either the entire transaction completes fully or doesn't execute at all. There is no in-between state i.e. transactions do not occur partially. If the transaction is successful, the changes are permanently applied. If the transaction fails, any changes made during the transaction are discarded.

**Consistency:** Maintaining Valid Data States

Consistency ensures that a database remains in a valid state before and after a transaction. a transaction should only take the database from one valid state to another

Isolation: Ensuring Concurrent Transactions Don't Interfere

This property ensures that **multiple transactions** can occur concurrently without leading to the **inconsistency** of the database state. Transactions occur independently without interference. Changes occurring in a particular transaction will not be visible to any other transaction until that particular change in that transaction is written to memory or has been committed.

**Durability:** Persisting Changes

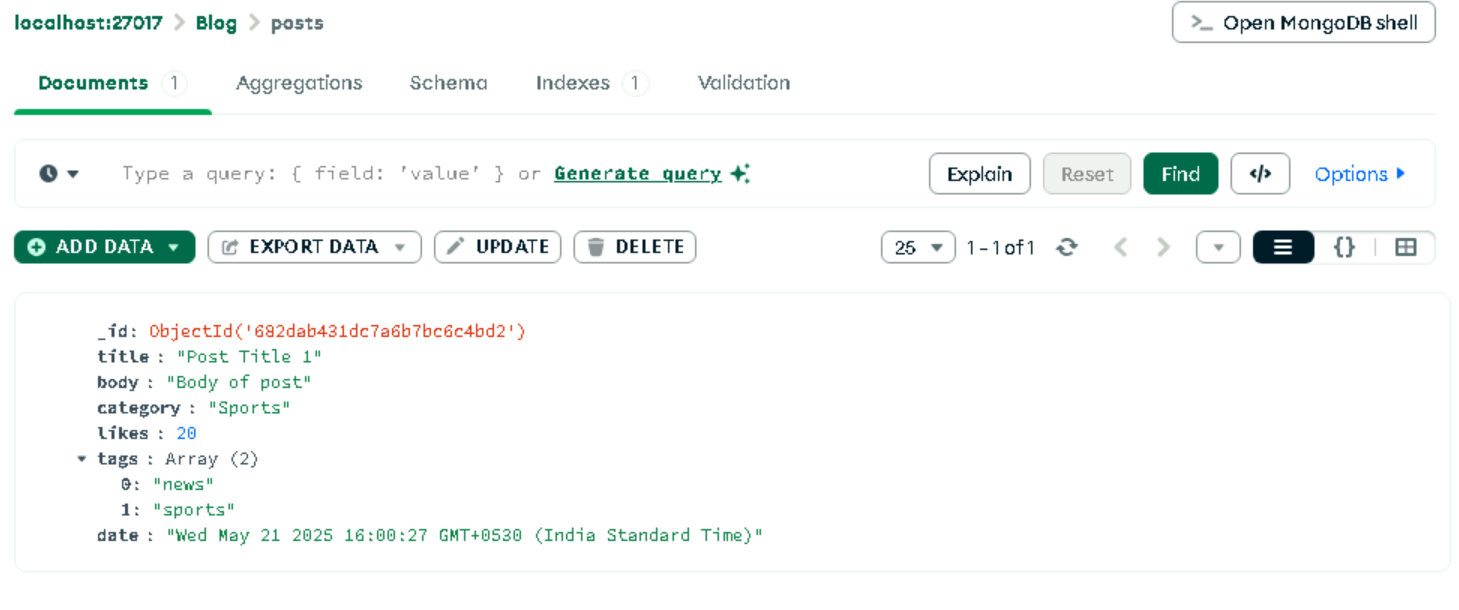
This property ensures that once the transaction has completed execution, the updates and modifications to the database are stored in and written to disk and they persist even if a system failure occurs. These updates now become permanent and are stored in **non-volatile memory**. In the event of a failure, the DBMS can recover the database to the state it was in after the last committed transaction, ensuring that no data is lost.

Installed MongoDB

Installed Mongo shell

Installed MongoDB Compass

Created database



**Inserting documents**

insertOne()

Blog> db.posts.insertOne([

{

...   title: "Post Title 1",

...   body: "Body of post.",

...   category: "Sports",

...   likes: 1,

...   tags: ["news", "Sports"],

...   date: Date()

... })

{

  acknowledged: true,

  insertedId: ObjectId('682da717ae74a79c09c4e49c')

}

**Find the data**

Blog> db.posts.find()

[

  { \_id: ObjectId('682da63fae74a79c09c4e49b'), title: 'my 1st post' },

  {

    \_id: ObjectId('682da717ae74a79c09c4e49c'),

    title: 'Post Title 1',

    body: 'Body of post.',

    category: 'News',

    likes: 1,

    tags: [ 'news', 'events' ],

    date: 'Wed May 21 2025 15:42:39 GMT+0530 (India Standard Time)'

  }

]

db.posts.findOne()

**Querying Data**

Blog> db.posts.findOne('my 1st post')

MongoInvalidArgumentError: Query filter must be a plain object or ObjectId

Blog> db.posts.find({category:"News"})

[

  {

    \_id: ObjectId('682da717ae74a79c09c4e49c'),

    title: 'Post Title 1',

    body: 'Body of post.',

    category: 'News',

    likes: 1,

    tags: [ 'news', 'events' ],

    date: 'Wed May 21 2025 15:42:39 GMT+0530 (India Standard Time)'

  }

]