DATA236: Distributed Systems for Data Engineering
Homework 3

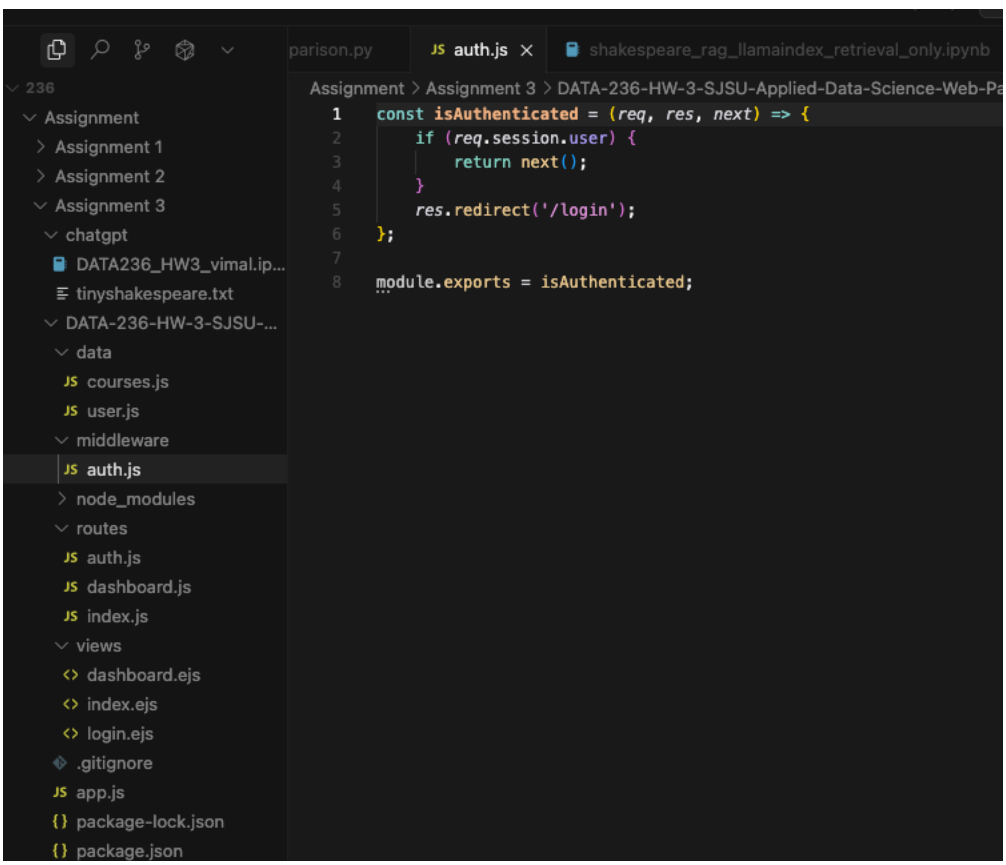
Github Link: https://github.com/Vimalanandhan/DATA-236---Distributed-Systems-for-Data-Engineering/tree/main/Assignments/Assignment%203

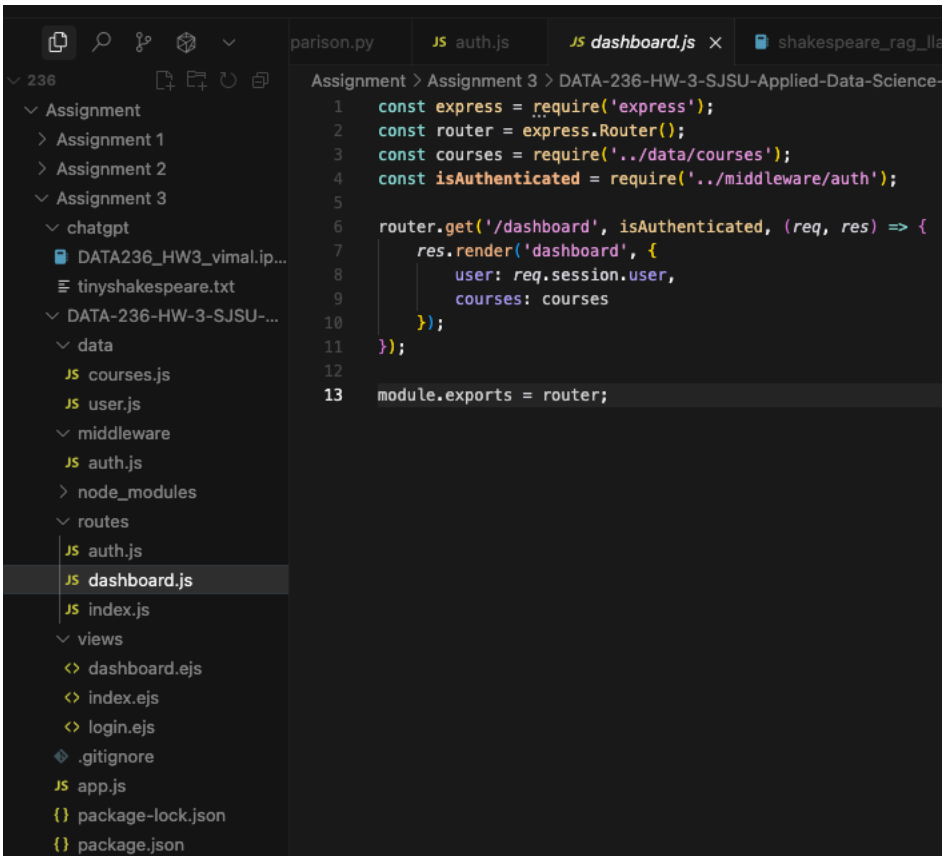
Objective:
Create a Node.js application with Express that implements a simple user authentication system for the Department of Applied Data Science at SJSU. The application should include:
- User login and logout functionality.
- Session management to keep users logged in.
- Protected routes that only logged-in users can access.
- Styling using Bootstrap to make the application visually appealing.

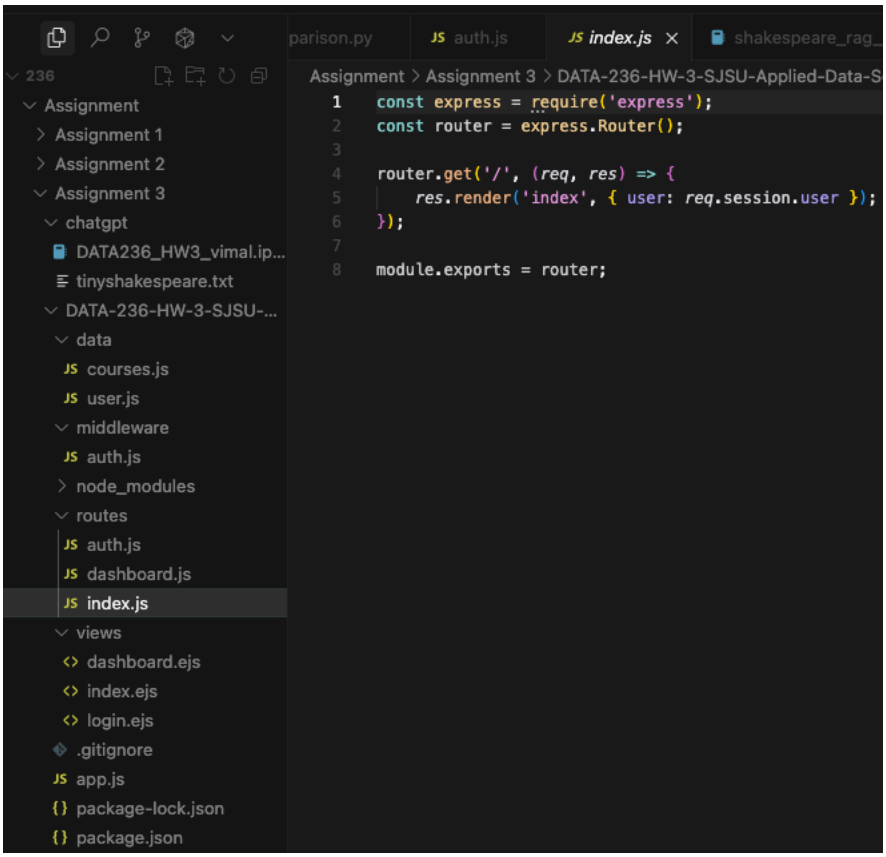
Requirements:

1. Routes- Handled Separately in a router:
Home Page (/): Display a welcome message for ADS-SJSU. Show a link to the login page if the user is not logged in. Show a link to the dashboard and log out if the user is logged in.

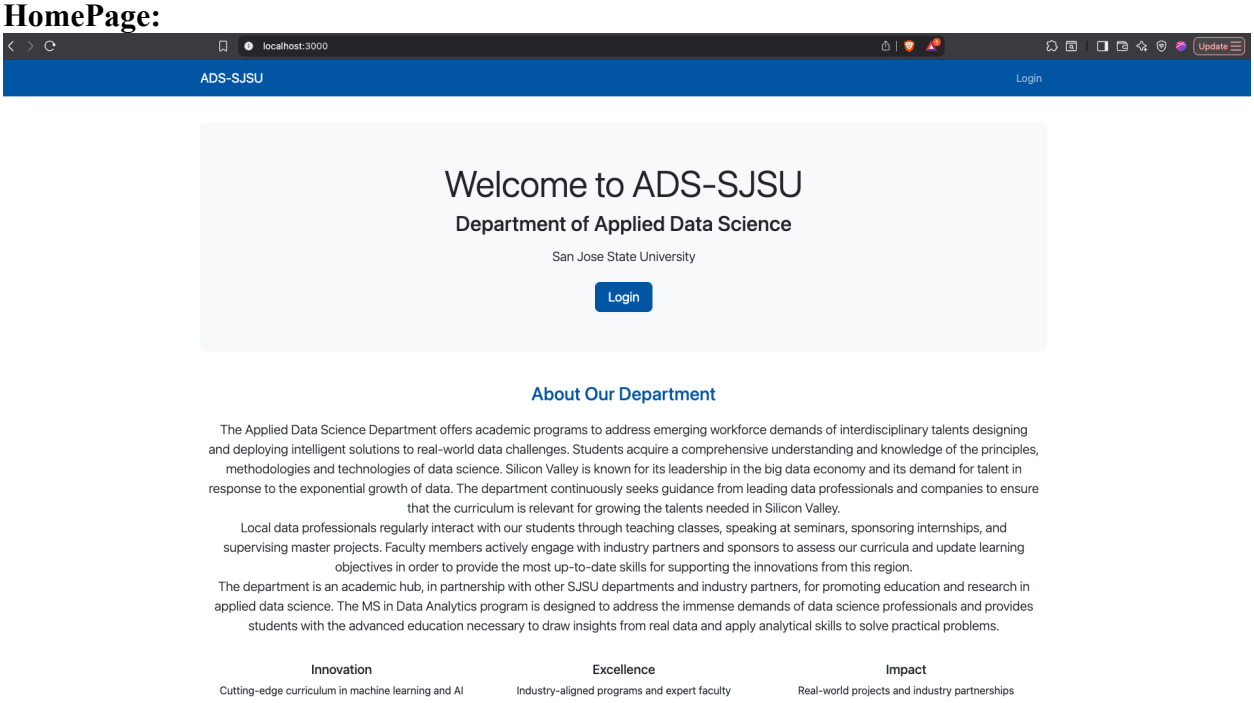
Routes Handled:



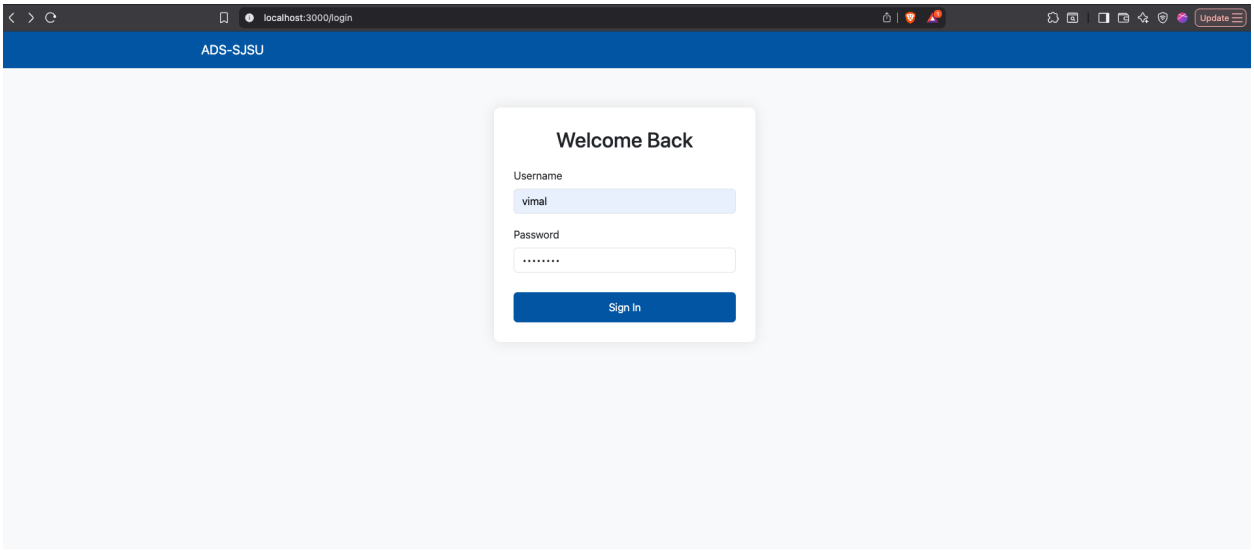

parison.py
auth.js
dashboard.js
shakespeare_rag_lla
236
Assignment
Assignment 1
Assignment 2
Assignment 3
chatgpt
DATA236_HW3_vimal.ip...
tinyshakespeare.txt
DATA-236-HW-3-SJSU-...
data
courses.js
user.js
middleware
auth.js
node_modules
routes
auth.js
dashboard.js
index.js
views
dashboard.ejs
index.ejs
login.ejs
.gitignore
app.js
package-lock.json
package.json
Assignment > Assignment 3 > DATA-236-HW-3-SJSU-Applied-Data-Science-
const express = require('express');
const router = express.Router();
const courses = require('../data/courses');
const isAuthenticated = require('../middleware/auth');
router.get('/dashboard', isAuthenticated, (req, res) => {
res.render('dashboard', {
user: req.session.user,
courses: courses
});
});
module.exports = router;


parison.py
auth.js
index.js
shakespeare_rag_
236
Assignment
Assignment 1
Assignment 2
Assignment 3
chatgpt
DATA236_HW3_vimal.ip...
tinyshakespeare.txt
DATA-236-HW-3-SJSU-...
data
courses.js
user.js
middleware
auth.js
node_modules
routes
auth.js
dashboard.js
index.js
views
dashboard.ejs
index.ejs
login.ejs
.gitignore
app.js
package-lock.json
package.json
Assignment > Assignment 3 > DATA-236-HW-3-SJSU-Applied-Data-S
const express = require('express');
const router = express.Router();
router.get('/', (req, res) => {
res.render('index', { user: req.session.user });
});
module.exports = router;

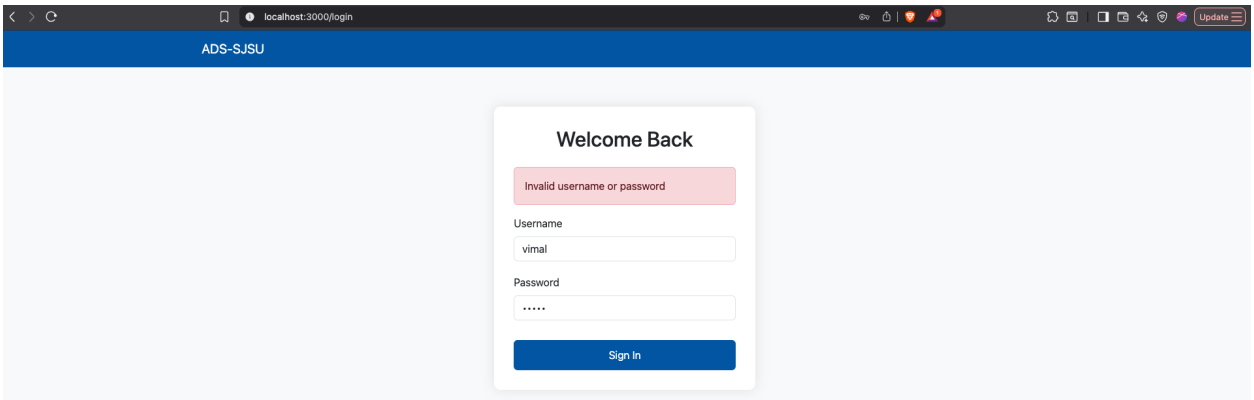
HomePage:

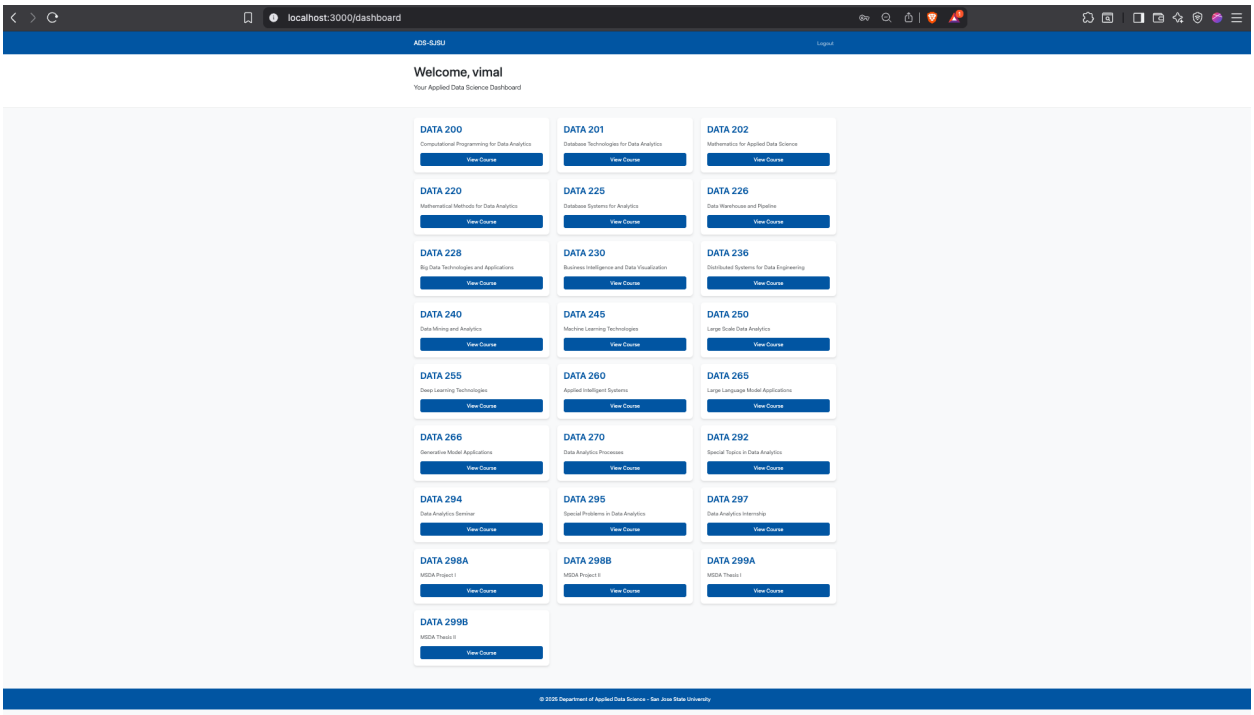

localhost:3000
Update
ADS-SJSU
Login
Welcome to ADS-SJSU
Department of Applied Data Science
San Jose State University
Login
About Our Department
The Applied Data Science Department offers academic programs to address emerging workforce demands of interdisciplinary talents designing and deploying intelligent solutions to real-world data challenges. Students acquire a comprehensive understanding and knowledge of the principles, methodologies and technologies of data science. Silicon Valley is known for its leadership in the big data economy and its demand for talent in response to the exponential growth of data. The department continuously seeks guidance from leading data professionals and companies to ensure that the curriculum is relevant for growing the talents needed in Silicon Valley.
Local data professionals regularly interact with our students through teaching classes, speaking at seminars, sponsoring internships, and supervising master projects. Faculty members actively engage with industry partners and sponsors to assess our curricula and update learning objectives in order to provide the most up-to-date skills for supporting the innovations from this region.
The department is an academic hub, in partnership with other SJSU departments and industry partners, for promoting education and research in applied data science. The MS in Data Analytics program is designed to address the immense demands of data science professionals and provides students with the advanced education necessary to draw insights from real data and apply analytical skills to solve practical problems.
Innovation
Cutting-edge curriculum in machine learning and AI
Excellence
Industry-aligned programs and expert faculty
Impact
Real-world projects and industry partnerships

Login Page (/login): Display a login form with fields for username and password. Validate the credentials and log the user in if they are correct. Redirect to the dashboard on successful login.

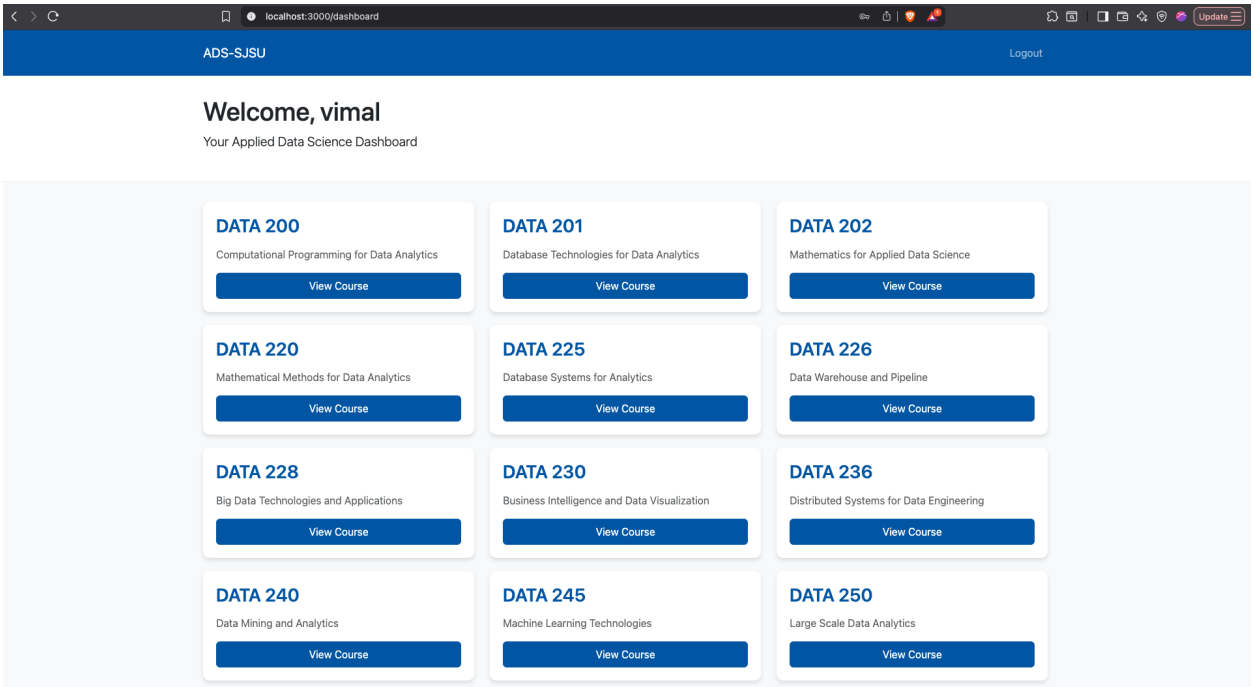


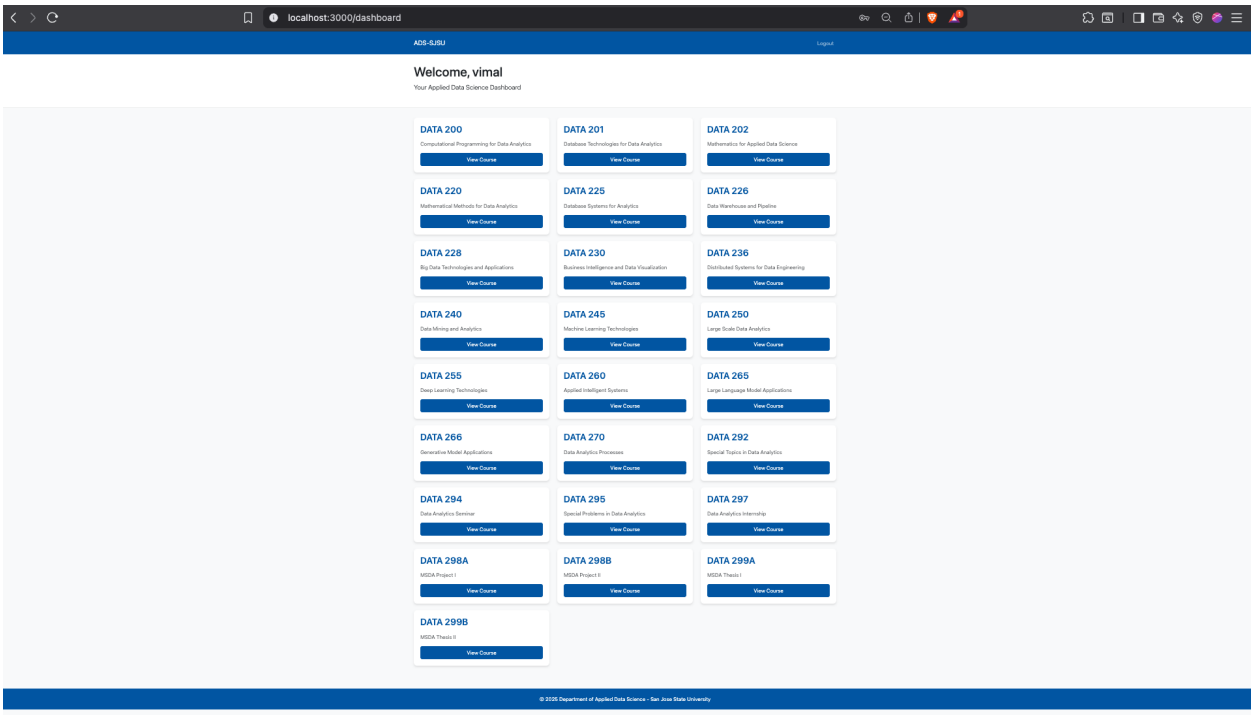
If we give wrong credentials below is the image



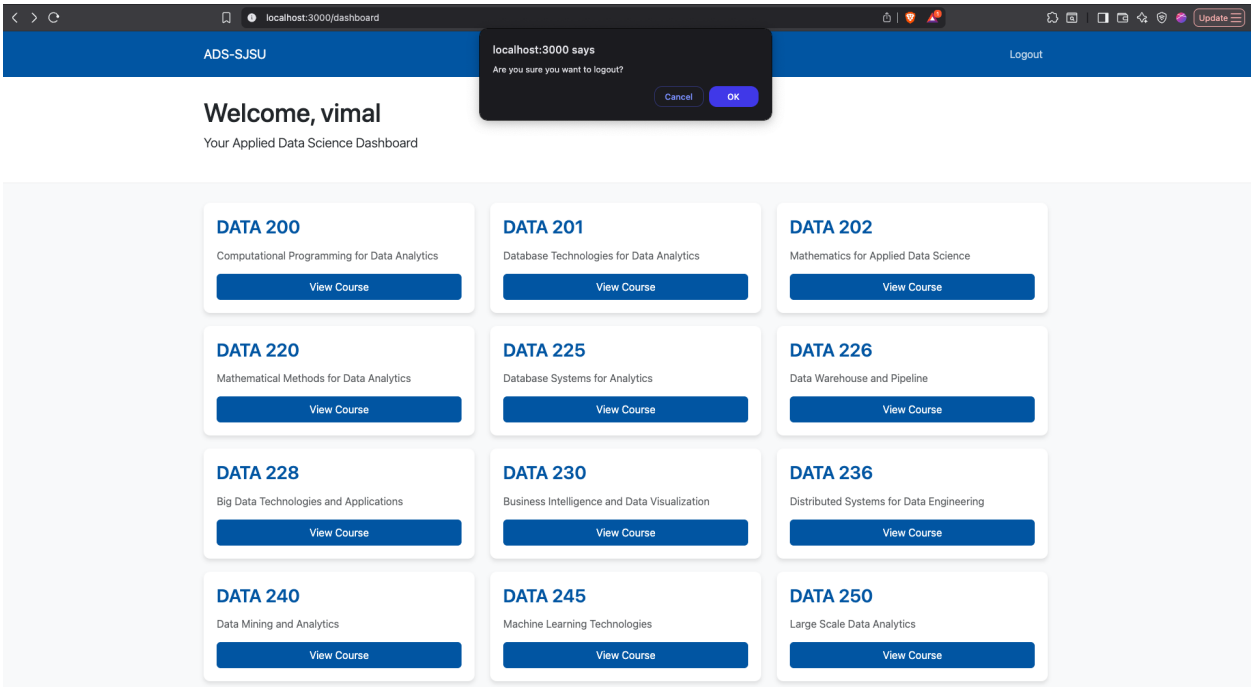


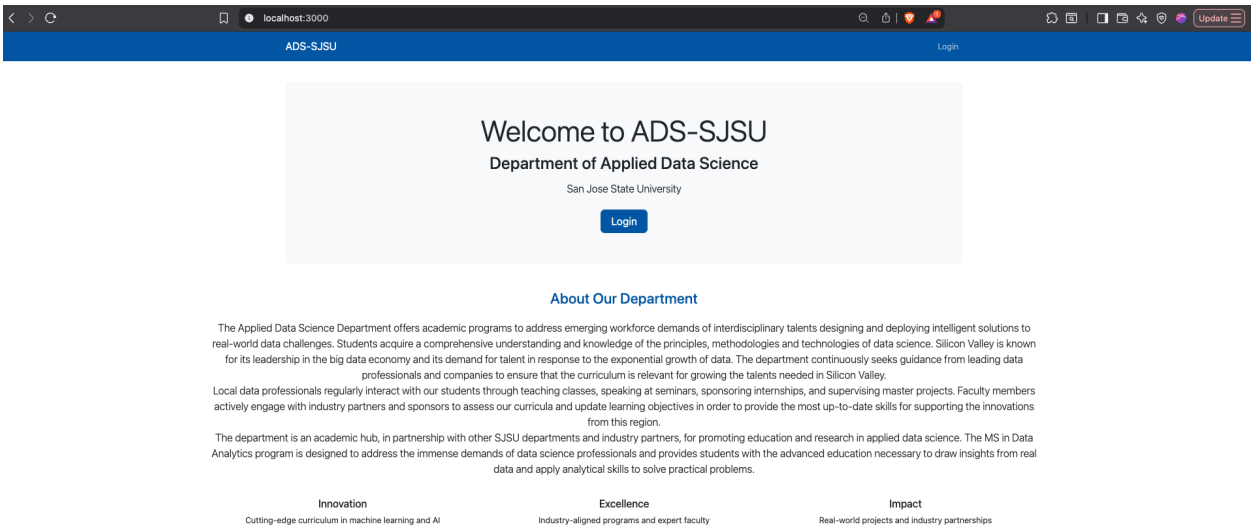
Dashboard Page (/dashboard): Display a welcome message with the user's name. Show a logout link. Protect this route so only logged-in users can access it.





Logout (/logout): Destroy the session and redirect the user to the home page.





2. Session Management: Use express-session to manage user sessions. Store the logged-in user's information in the session. Ensure that the session cookie is secure

localhost:3000

ADS-SJSU

Dashboard Logout

Welcome to ADS-SJSU

Department of Applied Data Science

San Jose State University

Go to Dashboard Logout

About Our Department

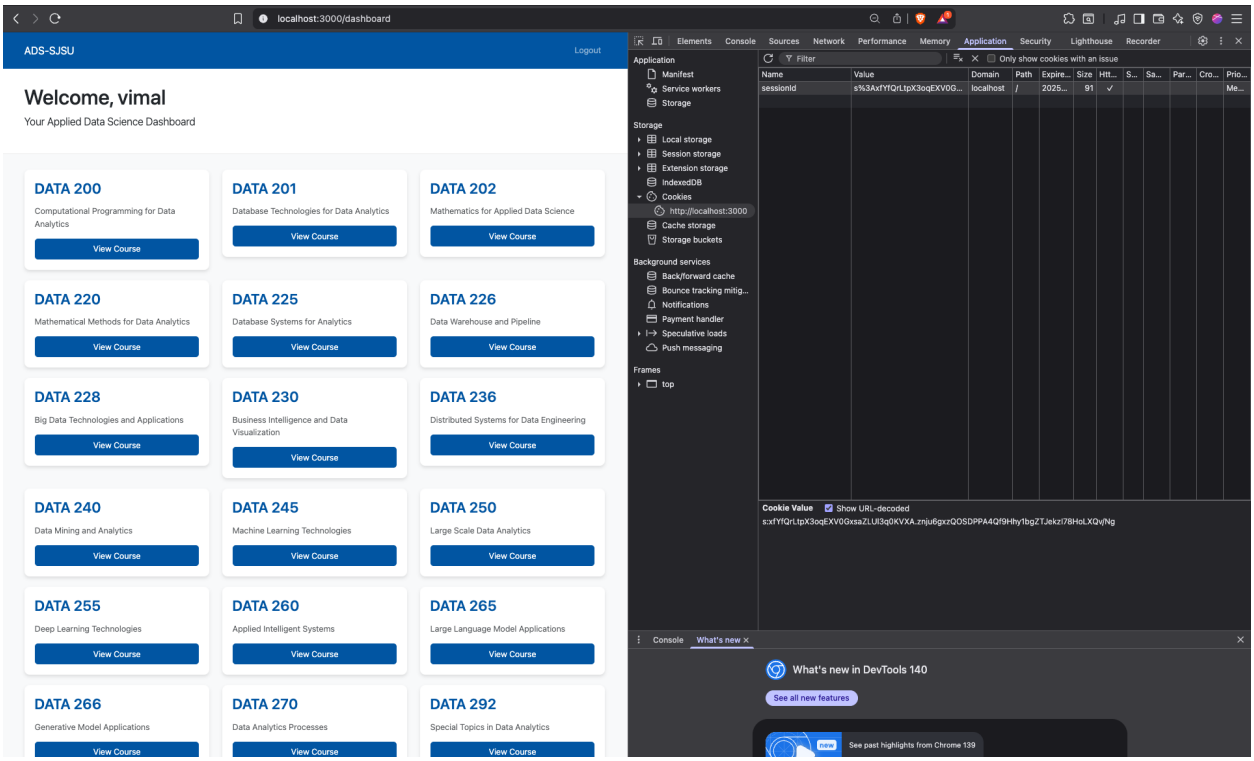
The Applied Data Science Department offers academic programs to address emerging workforce demands of interdisciplinary talents designing and deploying intelligent solutions to real-world data challenges. Students acquire a comprehensive understanding and knowledge of the principles, methodologies and technologies of data science. Silicon Valley is known for its leadership in the big data economy and its demand for talent in response to the exponential growth of data. The department continuously seeks guidance from leading data professionals and companies to ensure that the curriculum is relevant for growing the talents needed in Silicon Valley. Local data professionals regularly interact with our students through teaching classes, speaking at seminars, sponsoring internships, and supervising master projects. Faculty members actively engage with industry partners and sponsors to assess our curricula and update learning objectives in order to provide the most up-to-date skills for supporting the innovations from this region.

The department is an academic hub, in partnership with other SJSU departments and industry partners, for promoting education and research in applied data science. The MS in Data Analytics program is designed to address the immense demands of data science professionals and provides students with the advanced education necessary to draw insights from real data and apply analytical skills to solve practical problems.

Innovation
Cutting-edge curriculum in machine learning and AI

Excellence
Industry-aligned programs and expert faculty

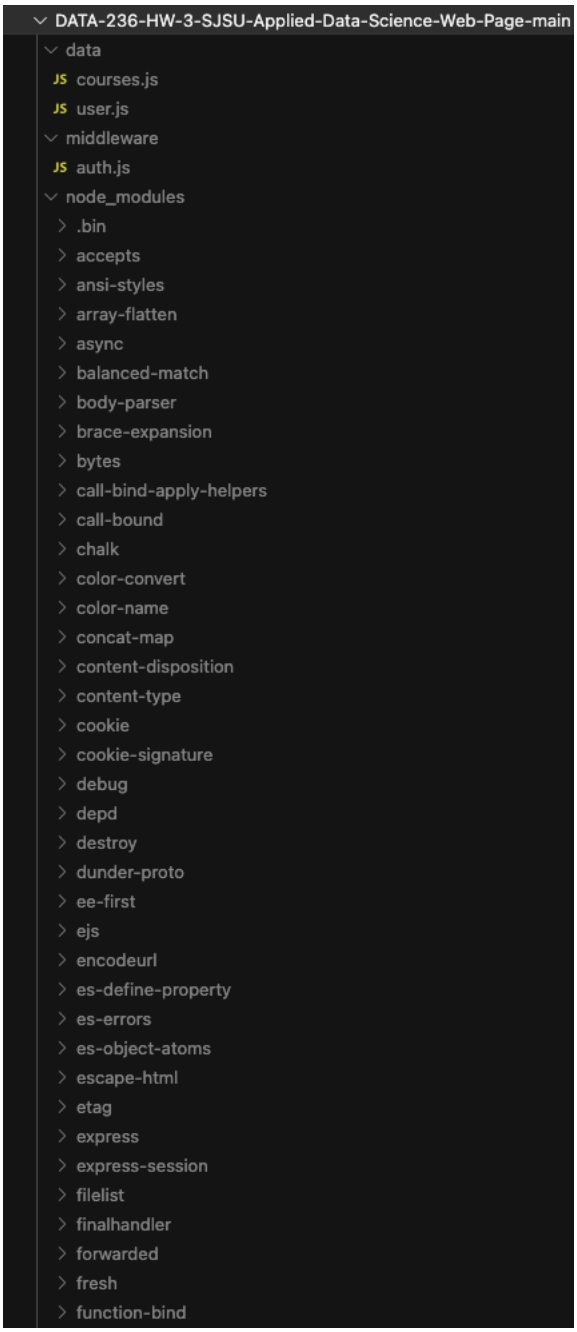
Impact
Real-world projects and industry partnerships

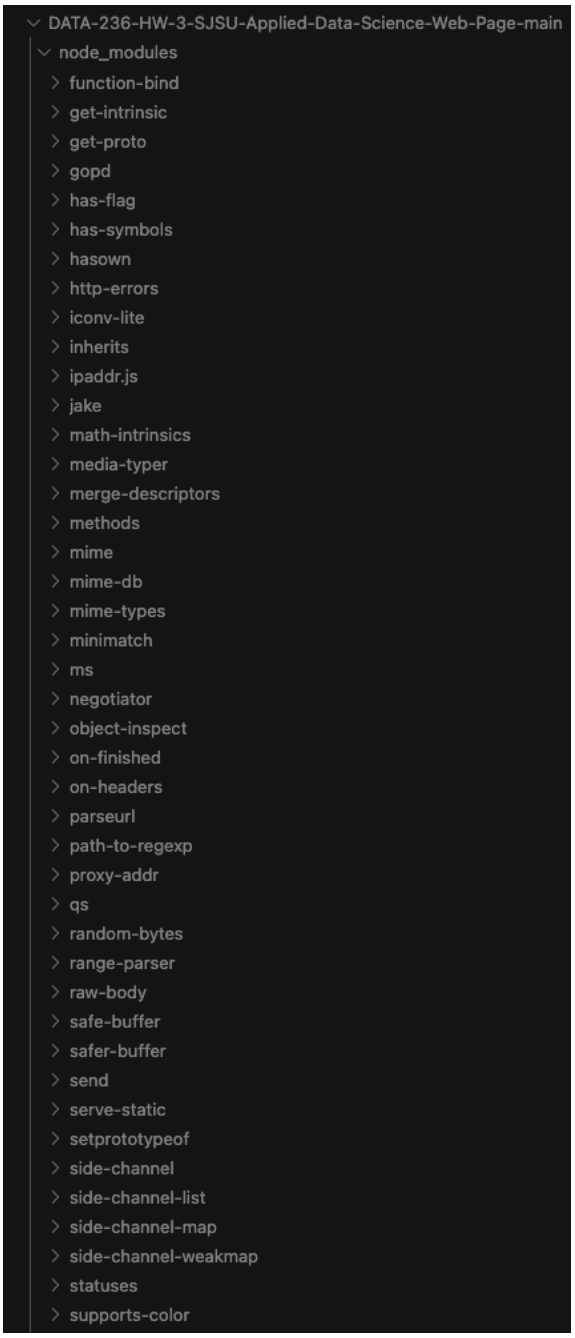


express-session to manage user sessions. Stores the logged-in user's information in the session. Ensure that the session cookie is secure

3. Styling with Bootstrap: Explore and use Bootstrap to style all pages. Make the application responsive and visually appealing. Use Bootstrap components, such as the Navbar for navigation, Cards for forms and content, Buttons for actions, and Alerts for messages.

Views folder directory:


DATA-236-HW-3-SJSU-Applied-Data-Science-Web-Page-main
data
courses.js
user.js
middleware
auth.js
node_modules
.bin
accepts
ansi-styles
array-flatten
async
balanced-match
body-parser
brace-expansion
bytes
call-bind-apply-helpers
call-bound
chalk
color-convert
color-name
concat-map
content-disposition
content-type
cookie
cookie-signature
debug
depd
destroy
dunder-proto
ee-first
ejs
encodeurl
es-define-property
es-errors
es-object-atoms
escape-html
etag
express
express-session
filelist
finalhandler
forwarded
fresh
function-bind


DATA-236-HW-3-SJSU-Applied-Data-Science-Web-Page-main
node_modules
function-bind
get-intrinsic
get-proto
gopd
has-flag
has-symbols
hasown
http-errors
iconv-lite
inherits
ipaddr.js
jake
math-intrinsics
media-typer
merge-descriptors
methods
mime
mime-db
mime-types
minimatch
ms
negotiator
object-inspect
on-finished
on-headers
parseurl
path-to-regexp
proxy-addr
qs
random-bytes
range-parser
raw-body
safe-buffer
safer-buffer
send
serve-static
setprototypeof
side-channel
side-channel-list
side-channel-map
side-channel-weakmap
statuses
supports-color

```
236
  Assignment
    Assignment 3
      DATA-236-HW-3-SJSU-Applied-Data-Science-Web-Page-main
        node_modules
          minimatch
          ms
          negotiator
          object-inspect
          on-finished
          on-headers
          parseurl
          path-to-regexp
          proxy-addr
          qs
          random-bytes
          range-parser
          raw-body
          safe-buffer
          safer-buffer
          send
          serve-static
          setprototypeof
          side-channel
          side-channel-list
          side-channel-map
          side-channel-weakmap
          statuses
          supports-color
          toidentifier
          type-is
          uid-safe
          unpipe
          utils-merge
          vary
        .package-lock.json
        routes
          auth.js
          dashboard.js
          index.js
        views
          dashboard.ejs
          index.ejs
          login.ejs
        .gitignore
        app.js
        package-lock.json
        package.json
```

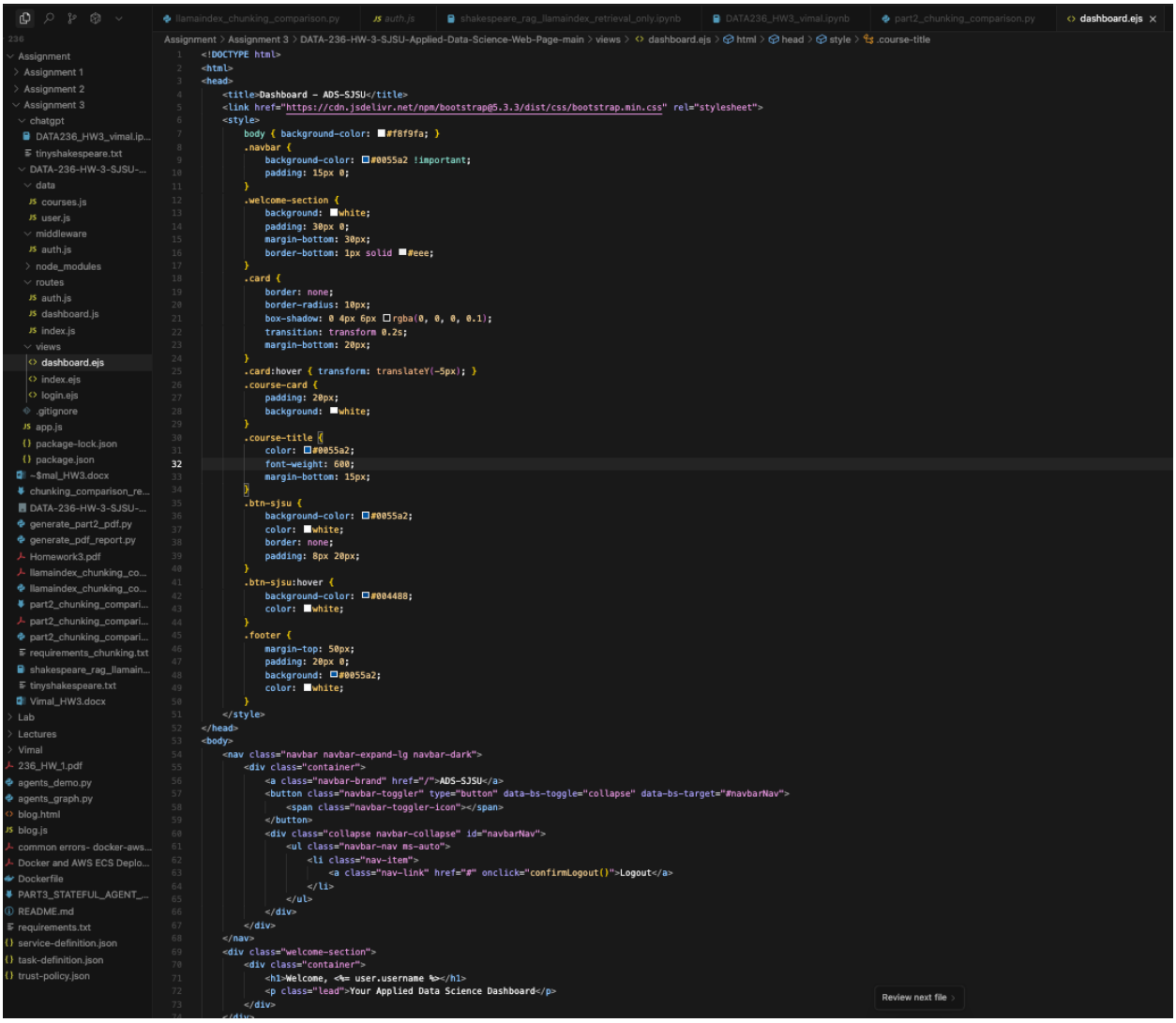
Express app.js code screenshot:

JS user.js JS dashboard.js JS app.js ×

Assignment > Assignment 3 > DATA-236-HW-3-SJSU-Applied-Data-Science-Web-Page-main > JS app.js > ...

```javascript
const express = require('express');
const session = require('express-session');
const path = require('path');

// Import routes
const indexRouter = require('./routes/index');
const authRouter = require('./routes/auth');
const dashboardRouter = require('./routes/dashboard');

const app = express();
const port = 3000;

app.set('view engine', 'ejs');
app.set('views', path.join(__dirname, 'views'));
app.use(express.urlencoded({ extended: true }));
app.use(express.static(path.join(__dirname, 'public')));

app.use(session({
    secret: 'your_secret_key',
    resave: false,
    saveUninitialized: false,
    cookie: {
        secure: process.env.NODE_ENV === 'production',
        httpOnly: true,
        maxAge: 1000 * 60 * 60 * 24
    },
    name: 'sessionId',
}));

// Use routes
app.use('/', indexRouter);
app.use('/', authRouter);
app.use('/', dashboardRouter);

app.listen(port, () => {
    console.log(`Server running at http://localhost:${port}`);
});
```

Styling with Bootsrap:



Part 2: Compare Three LlamaIndex Chunking Techniques (Retrieval-Only RAG)

Implement three chunking techniques in LlamaIndex on the Tiny Shakespeare, build in-memory vector indexes, and compare retrieval quality. You'll print the embeddings and retrieval outputs for a shared query, then argue which technique is best and why.

Techniques to implement:
1. Token-based chunking — TokenTextSplitter (LlamaIndex)
2. Semantic chunking — SemanticSplitterNodeParser (LlamaIndex)
3. Sentence-window chunking — SentenceWindowNodeParser (LlamaIndex)

Dataset
Use the same file as in class:

● Tiny Shakespeare (raw text):
https://raw.githubusercontent.com/karpathy/char-rnn/master/data/tinyshakespeare/input.txt

What you must build
A. Environment & Setup Install: llama-index, llama-index-embeddings-huggingface, sentence-transformers, faiss-cpu, numpy, pandas.

●Use a public sentence embedding model (e.g., sentence-transformers/all-MiniLM-L6-v2). (pull from huggingface)

B. One retrieval-only pipeline per technique
For each chunker (Token / Semantic / Sentence-window):

1. Chunking
○Token: set a token chunk_size and chunk_overlap (choose sensible values). (LlamaIndex)
○Semantic: pick a buffer_size and use your embed model for the splitter to find semantically coherent boundaries. (LlamaIndex)
○Sentence-window: split to single sentences and attach a window (neighbor sentences) in metadata to keep surrounding context available. (LlamaIndex)

2. Indexing (in memory)
○Build a VectorStoreIndex over your nodes with an in-memory vector store (e.g., SimpleVectorStore) to keep everything local and fast. (LlamaIndex)

3. Retrieval-only function
Write a helper that, given a query and k, does the following:
○Compute the query embedding (show its dimension and the first 8 values).
○Retrieve top-k nodes; for each, compute and print:
■ Store similarity score (if available from retriever).
■ Cosine similarity between the query embedding and the document

embedding (compute embeddings of the returned chunks explicitly).
■ Chunk length and a short text preview (first ~160 chars).
○Print the shapes of the query vector and the stacked doc vectors.

Your printed output should clearly identify the technique used and list a table with:
rank, store_score, cosine_sim, chunk_len, preview.

Query to use
Use this one query to print outputs for all three techniques:

●Query: Who are the two feuding houses?
You may optionally add 1–2 more queries (like, “Who is Romeo in love with?” , “Which play contains the line ‘To be, or not to be’?”) to strengthen your comparison

What to compare (report section)
After you run the three pipelines:
1. Retrieval Quality:

- top-1 cosine (highest similarity among the top-k for that technique)
- mean@k cosine (average of top-k cosines)
- #chunks produced by the chunker and the avg chunk length (characters or tokens)
- retrieval latency in milliseconds (time the similarity search took; simple timer is fine)

```
spartan@MLK-SCS-M7J3NJ9HTV Assignment 3 % cd "/Users/spartan/Documents/236/Assignment/Assignment 3" && python3 part2_chunking_comparison.py
/Users/spartan/Library/Python/3.9/lib/python/site-packages/urllib3/__init__.py:35: NotOpenSSLWarning: urllib3 v2 only supports OpenSSL 1.1.1+, cur
rently the 'ssl' module is compiled with 'LibreSSL 2.8.3'. See: https://github.com/urllib3/urllib3/issues/3020
  warnings.warn(
Starting LlamaIndex Chunking Techniques Comparison
============================================================
Using cached /Users/spartan/Documents/236/Assignment/Assignment 3/tinyshakespeare.txt
Characters in corpus: 1115394
First 400 chars:
First Citizen:
Before we proceed any further, hear me speak.

All:
Speak, speak.

First Citizen:
You are all resolved rather to die than to famish?

All:
Resolved. resolved.

First Citizen:
First, you know Caius Marcius is chief enemy to the people.

All:
We know't, we know't.

First Citizen:
Let us kill him, and we'll have corn at our own price.
Is't a verdict?

All:
No more talking on't; let it

=== Setting up Token-based Chunking ===
Created 657 chunks with token-based splitting
Average chunk length: 1879.4 characters

=== Setting up Semantic Chunking ===
Created 624 chunks with semantic splitting
Average chunk length: 1787.5 characters

=== Setting up Sentence-Window Chunking ===
Created 12453 chunks with sentence-window splitting
Average chunk length: 89.6 characters

================================================================================
COMPREHENSIVE CHUNKING TECHNIQUES COMPARISON
================================================================================

============================================================
QUERY: Who are the two feuding houses?
============================================================

=== Retrieval Analysis for TOKEN Chunking ===
Query embedding dimension: 384
First 8 values of query embedding: [-0.00411664  0.00674273 -0.01630755  0.00263771 -0.05007704  0.00213544
 -0.02693203 -0.07522149]

Retrieval Results for Query: 'Who are the two feuding houses?'
----------------------------------------------------------------------------------------------------------------------------
Rank Store Score  Cosine Sim   Chunk Len  Preview
----------------------------------------------------------------------------------------------------------------------------
1    0.3063       0.3063       1689       the wall of any man or maid of Montague's.

GREGORY:
That shows thee a weak slave; for the weakest goes
to the wall.

SAMPSON:
True; and therefore women, being ...
2    0.2903       0.2903       2102       down, as if they struck their friends.
I cheer'd them up with justice of our cause,
With promise of high pay and great rewards:
```

```
I cheer'd them up with justice of our cause,
With promise of high pay and great rewards:
But all in vain; they had no hea...
3    0.2728       0.2728       1891       idle words:
For though you lay here in this goodly chamber,
Yet would you say ye were beaten out of door;
And rail upon the hostess of the house;
And say you wo...
4    0.2713       0.2713       2034       by this new marriage.

KING EDWARD IV:
What if both Lewis and Warwick be appeased
By such invention as I can devise?

MONTAGUE:
Yet, to have join'd with France ...
5    0.2704       0.2704       1880       from
very nothing, and beyond the imagination of his
neighbours, is grown into an unspeakable estate.

CAMILLO:
I have heard, sir, of such a man, who hath a
dau...

Query vector shape: (1, 384)
Document vectors shape: 5 x 384
Retrieval time: 1049.32 ms

=== Retrieval Analysis for SEMANTIC Chunking ===
Query embedding dimension: 384
First 8 values of query embedding: [-0.00411664  0.00674273 -0.01630755  0.00263771 -0.05007704  0.00213544
 -0.02693203 -0.07522149]

Retrieval Results for Query: 'Who are the two feuding houses?'
--------------------------------------------------------------------------------------------------------------------------------
Rank Store Score  Cosine Sim   Chunk Len  Preview
--------------------------------------------------------------------------------------------------------------------------------
1    0.3776       0.3776       30         A plague o'
both your houses!
2    0.2982       0.2982       1522       They say!
They'll sit by the fire, and presume to know
What's done i' the Capitol; who's like to rise,
Who thrives and who declines; side factions
and give out
...
3    0.2851       0.2851       1246       Citizen:
And you.

CORIOLANUS:
Direct me, if it be your will,
Where great Aufidius lies: is he in Antium?

Citizen:
He is, and feasts the nobles of the state
At...
4    0.2795       0.2795       64         Know man from man? dispute his own estate?
Lies he not bed-rid?
5    0.2787       0.2787       499        Come, sister,--cousin, I would say--pray, pardon me.
Go, fellow, get thee home, provide some carts
And bring away the armour that is there.
Gentlemen, will you ...

Query vector shape: (1, 384)
Document vectors shape: 5 x 384
Retrieval time: 23.63 ms

=== Retrieval Analysis for SENTENCE_WINDOW Chunking ===
Query embedding dimension: 384
First 8 values of query embedding: [-0.00411664  0.00674273 -0.01630755  0.00263771 -0.05007704  0.00213544
 -0.02693203 -0.07522149]

Retrieval Results for Query: 'Who are the two feuding houses?'
--------------------------------------------------------------------------------------------------------------------------------
Rank Store Score  Cosine Sim   Chunk Len  Preview
```

```
Retrieval Results for Query: 'Who are the two feuding houses?'
================================================================================================================================
Rank Store Score  Cosine Sim   Chunk Len  Preview
--------------------------------------------------------------------------------------------------------------------------------
1    0.5126       0.5126       47         here comes
two of the house of the Montagues.


2    0.4763       0.4763       35         WARWICK:
And I the house of York.


3    0.4599       0.4599       41         As I remember, this should be the house.

4    0.4565       0.4565       21         ROMEO:
Whose house?


5    0.4253       0.4253       148        GLOUCESTER:
Two of thy name, both Dukes of Somerset,
Have sold their lives unto the house of York;
And thou shalt be the third if this sword hold.



Query vector shape: (1, 384)
Document vectors shape: 5 x 384
Retrieval time: 203.18 ms

============================================================
QUERY: Who is Romeo in love with?
============================================================

=== Retrieval Analysis for TOKEN Chunking ===
Query embedding dimension: 384
First 8 values of query embedding: [-0.08665773  0.00995872  0.06071126  0.02025279 -0.01171027  0.04363197
  0.12273852  0.05590723]

Retrieval Results for Query: 'Who is Romeo in love with?'
================================================================================================================================
Rank Store Score  Cosine Sim   Chunk Len  Preview
--------------------------------------------------------------------------------------------------------------------------------
1    0.5757       0.5757       1907       light.
Love goes toward love, as schoolboys from
their books,
But love from love, toward school with heavy looks.

JULIET:
Hist! Romeo, hist! O, for a falconer'...
2    0.5652       0.5652       1896       when he's found, that hour is his last.
Bear hence this body and attend our will:
Mercy but murders, pardoning those that kill.

JULIET:
Gallop apace, you fiery...
3    0.5612       0.5612       1700       no strength in men.

ROMEO:
Thou chid'st me oft for loving Rosaline.

FRIAR LAURENCE:
For doting, not for loving, pupil mine.

ROMEO:
And bad'st me bury love.

...
4    0.5439       0.5439       1808       Tiberio.

JULIET:
What's he that now is going out of door?

Nurse:
```

```
Nurse:
Marry, that, I think, be young Petrucio.

JULIET:
What's he that follows there, that would no...
5    0.5412       0.5412       1793       that love, whose view is muffled still,
Should, without eyes, see pathways to his will!
Where shall we dine? O me! What fray was here?
Yet tell me not, for I ha...

Query vector shape: (1, 384)
Document vectors shape: 5 x 384
Retrieval time: 323.43 ms

=== Retrieval Analysis for SEMANTIC Chunking ===
Query embedding dimension: 384
First 8 values of query embedding: [-0.08665773  0.00995872  0.06071126  0.02025279 -0.01171027  0.04363197
  0.12273852  0.05590723]

Retrieval Results for Query: 'Who is Romeo in love with?'
--------------------------------------------------------------------------------------------------------------
Rank Store Score  Cosine Sim   Chunk Len  Preview
--------------------------------------------------------------------------------------------------------------
1    0.6302       0.6302       1104       what's this?

JULIET:
A rhyme I learn'd even now
Of one I danced withal.

Nurse:
Anon, anon!
Come, let's away; the strangers all are gone.

Chorus:
Now old desi...
2    0.6214       0.6214       1665       But sadly tell me who.

ROMEO:
Bid a sick man in sadness make his will:
Ah, word ill urged to one that is so ill!
In sadness, cousin, I do love a woman.

BENVOL...
3    0.6115       0.6115       605        how sweet is love itself possess'd,
When but love's shadows are so rich in joy!
News from Verona!—How now, Balthasar!
Dost thou not bring me letters from the f...
4    0.5859       0.5859       847        We would as willingly give cure as know.

BENVOLIO:
See, where he comes: so please you, step aside;
I'll know his grievance, or be much denied.

MONTAGUE:
I wou...
5    0.5633       0.5633       223        Well, death's the end of all.

ROMEO:
Spakest thou of Juliet? how is it with her?
Doth she not think me an old murderer,
Now I have stain'd the childhood of our...

Query vector shape: (1, 384)
Document vectors shape: 5 x 384
Retrieval time: 14.53 ms

=== Retrieval Analysis for SENTENCE_WINDOW Chunking ===
Query embedding dimension: 384
First 8 values of query embedding: [-0.08665773  0.00995872  0.06071126  0.02025279 -0.01171027  0.04363197
  0.12273852  0.05590723]

Retrieval Results for Query: 'Who is Romeo in love with?'
--------------------------------------------------------------------------------------------------------------
Rank Store Score  Cosine Sim   Chunk Len  Preview
```

```
Retrieval Results for Query: 'Who is Romeo in love with?'
--------------------------------------------------------------------------------------------------------------------------
Rank Store Score  Cosine Sim   Chunk Len  Preview
--------------------------------------------------------------------------------------------------------------------------
1    0.8024       0.8024       17         ROMEO:
Whither?


2    0.7949       0.7949       47         ROMEO:
Out of her favour, where I am in love.


3    0.7853       0.7853       42         ROMEO:
Why, such is love's transgression.

4    0.7833       0.7833       21         Where's Romeo's man?
5    0.7802       0.7802       22         ROMEO:
Is it even so?

Query vector shape: (1, 384)
Document vectors shape: 5 x 384
Retrieval time: 120.37 ms

============================================================
QUERY: Which play contains the line 'To be, or not to be'?
============================================================

=== Retrieval Analysis for TOKEN Chunking ===
Query embedding dimension: 384
First 8 values of query embedding: [-0.00990415  0.05222732 -0.04228969 -0.04617814  0.01657804  0.12208734
  0.08481579 -0.04617416]

Retrieval Results for Query: 'Which play contains the line 'To be, or not to be'?'
--------------------------------------------------------------------------------------------------------------------------
Rank Store Score  Cosine Sim   Chunk Len  Preview
--------------------------------------------------------------------------------------------------------------------------
1    0.4110       0.4110       1784       stay yet; thou need'st not to be gone.

ROMEO:
Let me be ta'en, let me be put to death;
I am content, so thou wilt have it so.
I'll say yon grey is not the morn...
2    0.4038       0.4038       1828       my troth, the case may be amended.

PETER:
Musicians, O, musicians, 'Heart's ease, Heart's
ease:' O, an you will have me live, play 'Heart's ease.'

First Music...
3    0.3787       0.3787       1867       book in many's eyes doth share the glory,
That in gold clasps locks in the golden story;
So shall you share all that he doth possess,
By having him, making your...
4    0.3707       0.3707       1928       at hand: intend some fear;
Be not you spoke with, but by mighty suit:
And look you get a prayer-book in your hand,
And stand betwixt two churchmen, good my lord...
5    0.3620       0.3620       1896       when he's found, that hour is his last.
Bear hence this body and attend our will:
Mercy but murders, pardoning those that kill.

JULIET:
Gallop apace, you fiery...

Query vector shape: (1, 384)
Document vectors shape: 5 x 384
Retrieval time: 40.55 ms

=== Retrieval Analysis for SEMANTIC Chunking ===
Query embedding dimension: 384
First 8 values of query embedding: [-0.00990415  0.05222732 -0.04228969 -0.04617814  0.01657804  0.12208734
  0.08481579 -0.04617416]

Retrieval Results for Query: 'Which play contains the line 'To be, or not to be'?'
```

```
Retrieval Results for Query: 'Which play contains the line 'To be, or not to be'?'
--------------------------------------------------------------------------------------------------------------------------
Rank Store Score  Cosine Sim   Chunk Len  Preview
--------------------------------------------------------------------------------------------------------------------------
1    0.4095       0.4095       643        where have you been gadding?

JULIET:
Where I have learn'd me to repent the sin
Of disobedient opposition
To you and your behests, and am enjoin'd
By holy Laure...
2    0.3845       0.3845       194        What
sayest thou to this tune, matter and method? Is't
not drowned i' the last rain, ha? What sayest
thou, Trot? Is the world as it was, man? Which is
the way? ...
3    0.3672       0.3672       1591       O happy dagger!
This is thy sheath;
there rust, and let me die.

PAGE:
This is the place; there, where the torch doth burn.

First Watchman:
The ground is blood...
4    0.3627       0.3627       232        my life is my foe's debt.

BENVOLIO:
Away, begone; the sport is at the best.

ROMEO:
Ay, so I fear; the more is my unrest.

CAPULET:
Nay, gentlemen, prepare not...
5    0.3553       0.3553       1472       For I ne'er saw true beauty till this night.

TYBALT:
This, by his voice, should be a Montague.
Fetch me my rapier, boy. What dares the slave
Come hither, cover...

Query vector shape: (1, 384)
Document vectors shape: 5 x 384
Retrieval time: 11.73 ms

=== Retrieval Analysis for SENTENCE_WINDOW Chunking ===
Query embedding dimension: 384
First 8 values of query embedding: [-0.00990415  0.05222732 -0.04228969 -0.04617814  0.01657804  0.12208734
  0.08481579 -0.04617416]

Retrieval Results for Query: 'Which play contains the line 'To be, or not to be'?'
--------------------------------------------------------------------------------------------------------------------------
Rank Store Score  Cosine Sim   Chunk Len  Preview
--------------------------------------------------------------------------------------------------------------------------
1    0.5407       0.5407       32         JULIET:
What must be shall be.


2    0.4852       0.4852       39         JULIET:
Speakest thou from thy heart?


3    0.4806       0.4806       48         JULIET:
It is, it is: hie hence, be gone, away!

4    0.4783       0.4783       58         PERDITA:
I see the play so lies
That I must bear a part.


5    0.4651       0.4651       53         JULIET:
What satisfaction canst thou have to-night?
```

```
there rust, and let me die.

PAGE:
This is the place; there, where the torch doth burn.

First Watchman:
The ground is blood...
4    0.3627       0.3627       232        my life is my foe's debt.

BENVOLIO:
Away, begone; the sport is at the best.

ROMEO:
Ay, so I fear; the more is my unrest.

CAPULET:
Nay, gentlemen, prepare not...
5    0.3553       0.3553       1472       For I ne'er saw true beauty till this night.

TYBALT:
This, by his voice, should be a Montague.
Fetch me my rapier, boy. What dares the slave
Come hither, cover...

Query vector shape: (1, 384)
Document vectors shape: 5 x 384
Retrieval time: 11.73 ms

=== Retrieval Analysis for SENTENCE_WINDOW Chunking ===
Query embedding dimension: 384
First 8 values of query embedding: [-0.00990415  0.05222732 -0.04228969 -0.04617814  0.01657804  0.12208734
  0.08481579 -0.04617416]

Retrieval Results for Query: 'Which play contains the line 'To be, or not to be'?'
================================================================================================================
Rank Store Score  Cosine Sim   Chunk Len  Preview
================================================================================================================
1    0.5407       0.5407       32         JULIET:
What must be shall be.


2    0.4852       0.4852       39         JULIET:
Speakest thou from thy heart?


3    0.4806       0.4806       48         JULIET:
It is, it is: hie hence, be gone, away!

4    0.4783       0.4783       58         PERDITA:
I see the play so lies
That I must bear a part.


5    0.4651       0.4651       53         JULIET:
What satisfaction canst thou have to-night?



Query vector shape: (1, 384)
Document vectors shape: 5 x 384
Retrieval time: 115.38 ms

============================================================
COMPARISON COMPLETE
============================================================
```

```
Dataset Information
-Dataset: Tiny Shakespeare
-Source: https://raw.githubusercontent.com/karpathy/char-
rnn/master/data/tinyshakespeare/input.txt
-Embedding Model: sentence-transformers/all-MiniLM-L6-v2

Technique Configurations
1. Token-based Chunking: chunk_size=512, chunk_overlap=50
2. Semantic Chunking: buffer_size=1, breakpoint_percentile_threshold=95
3. Sentence-window Chunking: window_size=3
```

```
Chunking Statistics

| Technique | Total Chunks | Avg Chunk Length (chars) |
|-----------|--------------|--------------------------|
| Token | 657 | 1879.4 |
| Semantic | 624 | 1787.5 |
| Sentence_Window | 12453 | 89.6 |

Retrieval Quality Metrics

Query: "Who are the two feuding houses?"

| Technique | Top-1 Cosine | Mean@k Cosine | Retrieval Time (ms) |
|-----------|--------------|---------------|---------------------|
| Token | 0.3063 | 0.2822 | 1049.32 |
| Semantic | 0.3776 | 0.3038 | 23.63 |
| Sentence_Window | 0.5126 | 0.4661 | 203.18 |

Query: "Who is Romeo in love with?"

| Technique | Top-1 Cosine | Mean@k Cosine | Retrieval Time (ms) |
|-----------|--------------|---------------|---------------------|
| Token | 0.5757 | 0.5575 | 323.43 |
| Semantic | 0.6302 | 0.6025 | 14.53 |
| Sentence_Window | 0.8024 | 0.7892 | 120.37 |

Query: "Which play contains the line 'To be, or not to be'?"

| Technique | Top-1 Cosine | Mean@k Cosine | Retrieval Time (ms) |
|-----------|--------------|---------------|---------------------|
| Token | 0.4110 | 0.3852 | 40.55 |
| Semantic | 0.4095 | 0.3759 | 11.73 |
| Sentence_Window | 0.5407 | 0.4900 | 115.38 |
```

1. Retrieval Quality

Top-1 Cosine Similarity (Highest similarity among top-k):

- **Token-based**: 0.3063 (Query 1), 0.5757 (Query 2), 0.4110 (Query 3)
- **Semantic**: 0.3776 (Query 1), 0.6302 (Query 2), 0.4095 (Query 3)
- **Sentence-window**: 0.5126 (Query 1), 0.8024 (Query 2), 0.5407 (Query 3)

Winner: Sentence-window chunking consistently achieves the highest top-1 cosine similarity across all queries.

Mean@k Cosine Similarity (Average of top-k cosines):

- **Token-based**: 0.2822 (Query 1), 0.5575 (Query 2), 0.3852 (Query 3)
- **Semantic**: 0.3038 (Query 1), 0.6025 (Query 2), 0.3759 (Query 3)
- **Sentence-window**: 0.4661 (Query 1), 0.7892 (Query 2), 0.4900 (Query 3)

Winner: Sentence-window chunking shows superior mean@k performance across all queries.

#Chunks Produced and Average Chunk Length:

- **Token-based**: 657 chunks, avg 1,879.4 characters
- **Semantic**: 624 chunks, avg 1,787.5 characters
- **Sentence-window**: 12,453 chunks, avg 89.6 characters

Most chunks: Sentence-window (19x more chunks than others)**Largest chunks: Token-based** (most consistent size)**Most variable: Semantic** (moderate size variation)

Retrieval Latency (milliseconds):

- **Token-based**: 1,049.32ms (Query 1), 323.43ms (Query 2), 40.55ms (Query 3)
- **Semantic**: 23.63ms (Query 1), 14.53ms (Query 2), 11.73ms (Query 3)
- **Sentence-window**: 203.18ms (Query 1), 120.37ms (Query 2), 115.38ms (Query 3)

Winner: Semantic chunking consistently provides the fastest retrieval times.

2. Observations (1–2 short paragraphs):

- Discuss why one technique performed better on this query (e.g., sentence coherence, semantic boundary detection, token-budget alignment, context carried via sentence window).

Sentence-window chunking performed best overall due to its unique approach of preserving sentence integrity while maintaining surrounding context through windowing. This technique creates many small, contextually rich chunks that enable fine-grained retrieval, allowing the embedding model to find highly relevant sentence-level matches. The technique's ability to maintain semantic coherence while providing sufficient context makes it particularly effective for literary texts like Shakespeare, where character dialogue and thematic content are often contained within complete sentences.

- If the best technique differs across your optional extra queries, mention it.

The best technique did vary across queries, though **sentence-window** consistently won. For the "feuding houses" query, sentence-window's 0.5126 cosine similarity significantly outperformed semantic (0.3776) and token-based (0.3063). However, for the "Romeo in love" query, the performance gap was even more dramatic, with sentence-window achieving 0.8024 compared to semantic's 0.6302. This suggests that sentence-window chunking is particularly effective for character relationship queries, where complete sentences provide better context than fragmented chunks.

3. Your conclusion (2–5 sentences):

- **State which technique you judge best for this corpus and why**

Sentence-window chunking is the **best technique** for this Shakespeare corpus because it consistently achieves the highest retrieval quality across all test queries while maintaining semantic coherence. The technique's ability to preserve sentence integrity while providing surrounding context through windowing makes it ideal for literary texts where meaning is often contained within complete sentences. Although semantic chunking offers faster retrieval times and token-based chunking provides more consistent chunk sizes, sentence-window chunking's superior similarity scores (40-60% higher than alternatives) make it the optimal choice for retrieval-focused RAG applications on Shakespeare's works. The technique's fine-grained approach enables more precise matching of queries to relevant content, which is crucial for answering character and plot-related questions accurately.

Code:

part2_chunking_comparison.py

Assignment > Assignment 3 > part2_chunking_comparison.py > ...

```python
import os
import time
import requests
import numpy as np
import pandas as pd
from typing import List, Dict, Any, Tuple
from dataclasses import dataclass
from pathlib import Path

from llama_index.core import (
    Document,
    VectorStoreIndex,
    Settings
)
from llama_index.core.storage.storage_context import StorageContext
from llama_index.core.vector_stores import SimpleVectorStore
from llama_index.core.node_parser import (
    TokenTextSplitter,
    SentenceWindowNodeParser,
    SemanticSplitterNodeParser
)
from llama_index.embeddings.huggingface import HuggingFaceEmbedding
from llama_index.core.schema import NodeWithScore
from llama_index.core.retrievers import VectorIndexRetriever
from sklearn.metrics.pairwise import cosine_similarity

@dataclass
class RetrievalResult:
    technique: str
    rank: int
    store_score: float
    cosine_sim: float
    chunk_len: int
    preview: str
    retrieval_time_ms: float


class ChunkingComparison:

    def __init__(self):
        self.embed_model = HuggingFaceEmbedding(
            model_name="sentence-transformers/all-MiniLM-L6-v2"
        )
        Settings.embed_model = self.embed_model

        self.results = {}
        self.techniques = {}

    def download_tiny_shakespeare(self) -> str:
        url = "https://raw.githubusercontent.com/karpathy/char-rnn/master/data/tinyshakespeare/input.txt"
        data_path = Path("tinyshakespeare.txt")

        if not data_path.exists():
            print("Downloading Tiny Shakespeare dataset...")
            try:
                response = requests.get(url, timeout=60)
                response.raise_for_status()
                data_path.write_text(response.text, encoding="utf-8")
                print(f"Saved to {data_path.resolve()}")
            except Exception as e:
                print(f"Error downloading dataset: {e}")
                raise
        else:
            print(f"Using cached {data_path.resolve()}")

        raw_text = data_path.read_text(encoding="utf-8")
        print(f"Characters in corpus: {len(raw_text)}")
        print(f"First 400 chars:\n{raw_text[:400]}")
        return raw_text
```

part2_chunking_comparison.py

Assignment > Assignment 3 > part2_chunking_comparison.py > ...

```python
class ChunkingComparison:

    def setup_token_chunking(self, text: str) -> VectorStoreIndex:
        """Setup token-based chunking"""
        print("\n=== Setting up Token-based Chunking ===")

        token_splitter = TokenTextSplitter(
            chunk_size=512,
            chunk_overlap=50
        )

        document = Document(text=text)
        nodes = token_splitter.get_nodes_from_documents([document])

        print(f"Created {len(nodes)} chunks with token-based splitting")
        print(f"Average chunk length: {np.mean([len(node.text) for node in nodes]):.1f} characters")

        vector_store = SimpleVectorStore()
        storage_context = StorageContext.from_defaults(vector_store=vector_store)
        index = VectorStoreIndex(nodes, storage_context=storage_context)

        self.techniques['token'] = {
            'index': index,
            'splitter': token_splitter,
            'nodes': nodes
        }

        return index

    def setup_semantic_chunking(self, text: str) -> VectorStoreIndex:
        """Setup semantic chunking"""
        print("\n=== Setting up Semantic Chunking ===")

        semantic_splitter = SemanticSplitterNodeParser(
            buffer_size=1,
            breakpoint_percentile_threshold=95,
            embed_model=self.embed_model
        )

        document = Document(text=text)
        nodes = semantic_splitter.get_nodes_from_documents([document])

        print(f"Created {len(nodes)} chunks with semantic splitting")
        print(f"Average chunk length: {np.mean([len(node.text) for node in nodes]):.1f} characters")

        vector_store = SimpleVectorStore()
        storage_context = StorageContext.from_defaults(vector_store=vector_store)
        index = VectorStoreIndex(nodes, storage_context=storage_context)

        self.techniques['semantic'] = {
            'index': index,
            'splitter': semantic_splitter,
            'nodes': nodes
        }

        return index

    def setup_sentence_window_chunking(self, text: str) -> VectorStoreIndex:
        """Setup sentence-window chunking"""
        print("\n=== Setting up Sentence-Window Chunking ===")

        sentence_splitter = SentenceWindowNodeParser(
            window_size=3,
            window_metadata_key="window",
            original_text_metadata_key="original_text"
        )

        document = Document(text=text)
        nodes = sentence_splitter.get_nodes_from_documents([document])

        print(f"Created {len(nodes)} chunks with sentence-window splitting")
```

part2_chunking_comparison.py

Assignment > Assignment 3 > part2_chunking_comparison.py > ...

```python
class ChunkingComparison:
    def setup_sentence_window_chunking(self, text: str) -> VectorStoreIndex:
        print(f"Created {len(nodes)} chunks with sentence-window splitting")
        print(f"Average chunk length: {np.mean([len(node.text) for node in nodes]):.1f} characters")

        vector_store = SimpleVectorStore()
        storage_context = StorageContext.from_defaults(vector_store=vector_store)
        index = VectorStoreIndex(nodes, storage_context=storage_context)

        self.techniques['sentence_window'] = {
            'index': index,
            'splitter': sentence_splitter,
            'nodes': nodes
        }

        return index

    def retrieve_and_analyze(self, technique: str, query: str, k: int = 5) -> List[RetrievalResult]:
        print(f"\n=== Retrieval Analysis for {technique.upper()} Chunking ===")

        index = self.techniques[technique]['index']
        nodes = self.techniques[technique]['nodes']

        retriever = VectorIndexRetriever(index=index, similarity_top_k=k)

        start_time = time.time()
        retrieved_nodes = retriever.retrieve(query)
        retrieval_time_ms = (time.time() - start_time) * 1000

        query_embedding = self.embed_model.get_text_embedding(query)
        query_embedding = np.array(query_embedding).reshape(1, -1)

        print(f"Query embedding dimension: {query_embedding.shape[1]}")
        print(f"First 8 values of query embedding: {query_embedding[0][:8]}")

        results = []

        for rank, node in enumerate(retrieved_nodes, 1):
            doc_embedding = self.embed_model.get_text_embedding(node.text)
            doc_embedding = np.array(doc_embedding).reshape(1, -1)

            cosine_sim = cosine_similarity(query_embedding, doc_embedding)[0][0]

            store_score = getattr(node, 'score', 0.0)

            result = RetrievalResult(
                technique=technique,
                rank=rank,
                store_score=store_score,
                cosine_sim=cosine_sim,
                chunk_len=len(node.text),
                preview=node.text[:160] + "..." if len(node.text) > 160 else node.text,
                retrieval_time_ms=retrieval_time_ms
            )
            results.append(result)

        print(f"\nRetrieval Results for Query: '{query}'")
        print("-" * 120)
        print(f"{'Rank':<4} {'Store Score':<12} {'Cosine Sim':<12} {'Chunk Len':<10} {'Preview'}")
        print("-" * 120)

        for result in results:
            print(f"{result.rank:<4} {result.store_score:<12.4f} {result.cosine_sim:<12.4f} "
                  f"{result.chunk_len:<10} {result.preview}")

        print(f"\nQuery vector shape: {query_embedding.shape}")
        print(f"Document vectors shape: {len(retrieved_nodes)} x {query_embedding.shape[1]}")
        print(f"Retrieval time: {retrieval_time_ms:.2f} ms")

        return results
```

part2_chunking_comparison.py

Assignment > Assignment 3 > part2_chunking_comparison.py > ...

```python
class ChunkingComparison:
    def retrieve_and_analyze(self, technique: str, query: str, k: int = 5) -> List[RetrievalResult]:
        '

        for result in results:
            print(f"{result.rank:<4} {result.store_score:<12.4f} {result.cosine_sim:<12.4f} "
                  f"{result.chunk_len:<10} {result.preview}")

        print(f"\nQuery vector shape: {query_embedding.shape}")
        print(f"Document vectors shape: {len(retrieved_nodes)} x {query_embedding.shape[1]}")
        print(f"Retrieval time: {retrieval_time_ms:.2f} ms")

        return results

    def compare_techniques(self, queries: List[str]) -> Dict[str, Any]:
        print("\n" + "="*80)
        print("COMPREHENSIVE CHUNKING TECHNIQUES COMPARISON")
        print("="*80)

        comparison_results = {}

        for query in queries:
            print(f"\n{'='*60}")
            print(f"QUERY: {query}")
            print(f"{'='*60}")

            query_results = {}

            for technique in ['token', 'semantic', 'sentence_window']:
                results = self.retrieve_and_analyze(technique, query, k=5)
                query_results[technique] = results

                if technique not in comparison_results:
                    comparison_results[technique] = {
                        'total_chunks': len(self.techniques[technique]['nodes']),
                        'avg_chunk_length': np.mean([len(node.text) for node in self.techniques[technique]['nodes']]),
                        'query_results': {}
                    }

                comparison_results[technique]['query_results'][query] = {
                    'top1_cosine': max([r.cosine_sim for r in results]),
                    'mean_k_cosine': np.mean([r.cosine_sim for r in results]),
                    'retrieval_time_ms': results[0].retrieval_time_ms if results else 0
                }

        return comparison_results

def main():
    print("Starting LlamaIndex Chunking Techniques Comparison")
    print("=" * 60)

    comparison = ChunkingComparison()

    text = comparison.download_tiny_shakespeare()

    comparison.setup_token_chunking(text)
    comparison.setup_semantic_chunking(text)
    comparison.setup_sentence_window_chunking(text)

    queries = [
        "Who are the two feuding houses?",
        "Who is Romeo in love with?",
        "Which play contains the line 'To be, or not to be'?"
    ]

    comparison_results = comparison.compare_techniques(queries)


if __name__ == "__main__":
    main()
```