FlowState

SUBMITTED IN PARTIAL FULFILLMENT FOR THE REQUIREMENT OF THE AWARD OF DEGREE OF

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE

Submitted by

PRABHAKAR MISHRA (2100290120119)

RACHITAVYA SHARMA (2100290120131)

HARSHI AGRAWAL (2100290130075)

RISHABH KUSHWAHA (2100290130141)

Supervised by

Dr. ABHISHEK GOYAL
Associate Professor

Session 2024-25

DEPARTMENT OF COMPUTER SCIENCE

KIET GROUP OF INSTITUTIONS, GHAZIABAD

(Affiliated to Dr. A. P. J. Abdul Kalam Technical University, Lucknow, U.P., India)

May 2025

DECLARATION

We hereby declare that this submission is our work and that, to the best of our knowledge and belief, it contains no material previously published or written by another person nor material which to a substantial extent has been accepted for the award of any other degree or diploma of the university or other institute of higher learning, except where due acknowledgement has been made in the text.

Signature

Name:

Rachitavya Sharma (2100290120131)(CS)
Prabhakar Mishra (2100290120119)(CS)
Harshi Agrawal (2100290130075)(IT)
Rishabh Kushwaha (2100290130141)(IT)

Date:09/05/2025

CERTIFICATE

This is to certify that the project report entitled "**FlowState**" which is submitted by **Prabhakar Mishra, Rachitavya Sharma, Harshi Agrawal, Rishabh Kushwaha** in partial fulfilment of the requirement for the award of degree B. Tech. in the department of Computer Science of KIET Group of Institutions, Delhi NCR affiliated to Dr. A.P.J. Abdul Kalam Technical University, Lucknow is a record of the candidates own work carried out by them under my supervision. The matter embodied in this report is original and has not been submitted for the award of any other degree.

.

Date: 09/05/2025

Supervisor

Dr. Abhishek Goyal

(Associate Professor)

ACKNOWLEDGEMENT

It gives us great pleasure to present the report of the B. Tech project undertaken during B. Tech. Final Year. We owe a special gratitude to Dr. Abhishek Goyal, Department of Computer Science, KIET, Ghaziabad, for his constant support and guidance throughout our work. His sincerity, thoroughness and perseverance have been a constant source of inspiration for us.

We also take the opportunity to acknowledge the contribution of Dr. Ajay Kr. Shrivastava, Dean & Professor, Department of Computer Science, KIET, Ghaziabad, for his full support and assistance during the development of the project. We also do not like to miss the opportunity to acknowledge the contribution of all the department's faculty members for their kind assistance and cooperation during the development of our project.

Last but not least, we acknowledge our friends for their contribution to the completion of the project.

Date: 09/05/2025
Signature:
Name:
Rachitavya Sharma (2100290120131) (CS)

Prabhakar Mishra (2100290120119) (CS)

Harshi Agrawal (2100290130075) (IT)

Rishabh Kushwaha (2100290130141) (IT)

ABSTRACT

Education is the foremost part of the career, but COVID-19 pandemic accelerated the global shift to e-learning. The accessibility of online resources has reshaped learning, enabling students to acquire knowledge anytime and anywhere. The drastic growth of online education has changed the scene, providing an abundant number of resources to students. But it comes with various challenges i.e., due to the high number of resources available on the internet, students are not aware which content is authenticated and reliable. Filtering the content with machine learning and NLP, most reliable content can be provided to the students with a streamlined and efficient learning experience.

This approach aligns with a recommendation system which can predict user interests and infer their mental processes. Based on the user's demands and while taking into account their interests, the data will be tailored to an individual's unique mindset. Numerous recommendation systems have been developed using diverse methodologies. As OTT platforms, shopping, travel, and other websites proliferate and strive to quickly improve their user suggestions, the research into such systems has gained popularity up to this point.

In this paper, we have implemented content-based filtering and recommendation system using machine learning techniques. We have compared and reviewed various recommendation models and based on the best model, we have applied the recommendation system for improving the learning experience to the user. The findings of this research paper contribute towards the ongoing research in creating e-learning, gaining insights into AI-driven systems' capacity to enhance students' engagement and learning performance.

FlowState is a revolutionary online learning site that aims to overcome the frustrating problem that bothers students today: the sheer number of study materials found on the web. In the present era of technology, students tend to get confused in an ocean of tutorials, articles, and videos without knowing where to start or how to allocate their studying time.

Further, FlowState builds a community of users in which they can collaborate, share knowledge, and discuss things. Through its ease of use and robust functionalities, FlowState looks to transform how

students experience online learning. By offering a personalized and curated learning experience, FlowState enables students to control their own education and achieve academic success.

TABLE OF CONTENTS

Page No.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

1. **ML** - Machine Learning
2. **AI** - Artificial Intelligence
3. **NLP** - Natural Language Processing
4. **UX/UI** - User Experience / User Interface
5. **MVP** - Minimum Viable Product
6. **LMS** - Learning Management System
7. **SaaS** - Software as a Service
8. **B2C** - Business to Consumer
9. **CRUD** - Create, Read, Update, Delete
10. **REST** - Representational State Transfer
11. **API** - Application Programming Interface
12. **CI/CD** - Continuous Integration / Continuous Deployment
13. **ETL** - Extract, Transform, Load
14. **SEO** - Search Engine Optimization
15. **SSL** - Secure Sockets Layer
16. **JWT** - JSON Web Token
17. **PWA** - Progressive Web App
18. **HTTP** - Hypertext Transfer Protocol

SDG MAPPING WITH JUSTIFICATION

SDG Mapping: Quality Education (SDG 4)

Target 4.4: Increase the number of youth and adults who have relevant skills, including technical and vocational skills, for employment, decent jobs, and entrepreneurship.

Justification:
FlowState addresses the need for high-quality, accessible education by providing personalized learning content tailored to each student's goals and interests. The platform's AI-driven recommendation system ensures that learners receive content relevant to their current academic or professional needs. Whether preparing for exams, developing new skills, or exploring career opportunities, FlowState enables learners to find the right resources to enhance their learning experience. This directly supports SDG 4 by helping individuals gain the skills necessary for employment and entrepreneurship, contributing to their personal and professional development.

Target 4.7: Ensure that all learners acquire the knowledge and skills needed to promote sustainable development, including through education for sustainable development and sustainable lifestyles, human rights, gender equality, promotion of a culture of peace and non-violence, global citizenship, and appreciation of cultural diversity.

Justification:
FlowState curates content that not only supports academic achievement but also promotes essential life skills. By integrating educational resources that cover diverse topics—including human rights, sustainability, and global citizenship—the platform helps learners acquire the knowledge necessary to contribute to a sustainable future. FlowState's personalized recommendations ensure that students can access the most relevant resources for promoting a culture of peace, understanding, and respect for cultural diversity.

Conclusion:

FlowState's focus on personalized learning and content curation directly contributes to SDG 4: Quality Education. By making high-quality educational resources accessible to all learners, regardless of their background or circumstances, the platform empowers individuals to gain relevant skills for employment, personal growth, and contributing to sustainable development.

CHAPTER 1
INTRODUCTION

1.1 Introduction

FlowState is a new solution in online education aimed at helping students deal with a common problem: too much information. With the internet flooded with learning materials, students often struggle to find what they need, leading to confusion and wasted time. FlowState steps in to simplify things. It's designed to make online learning easier by giving students a personalized experience. Instead of drowning in a sea of content, FlowState helps users find the right resources for them. Whether it's videos, articles, or tutorials, FlowState tailors recommendations to match each user's interests and goals. In this introduction, we'll explore the challenges students face in today's digital learning landscape and why FlowState is here to help. We'll look at how FlowState works and what sets it apart from other online learning platforms. By the end, you'll understand how FlowState aims to make online learning more manageable and effective.

Nowadays, recommender systems are being increasingly used for a large number of applications such as web, books, e-learning, tourism, e-commerce, news, specialized research resources etc. It is therefore important to build high-quality and exclusive recommender systems for providing personalized recommendations to the users in various applications. This approach also aligns with recommendation system principles, where user preferences and behavior are analyzed to provide personalized suggestions. Recommendation system helps users to find their own interests and dive to make the benefit out of it. Systems for making recommendations are widely utilized today in everything from education to entertainment. Additionally, the data needed for this system is growing every day as a result of the internet's widespread accessibility. Our recommendation engine for education is primarily built using machine learning, cosine similarity metrics, and content-based filtering approaches. It delves into the technical methodologies and algorithms that underpin these systems, highlighting their capacity to transform the learning experience.

In the report, we have firstly properly explored the dataset and then done the exploratory analysis of the dataset in order to assess the patterns and to understand it correctly. After that we have done some preprocessing of the dataset. Created some machine learning based recommendation models using content-based filtering. Then we have trained and test these models. Then we used the best recommendation model to create the Rest API and then created the recommendation system GUI for online education.

1.2 Project Description

FlowState is a new innovative online education solution that hopes to tackle the problem of too many learning materials on the internet. Students are often overwhelmed by the volume of content they are exposed to which causes them confusion and repetition. FlowState solves these problems by making personalized recommendations for the user based on their interests, learning goals, and preferences.

FlowState is a solution for the user regardless of the nature of the material whether it be videos, articles, or tutorials. FlowState curates that content to create focus for students and to limit distractions from what is important. FlowState is designed for the user and learning outcomes are optimized. This makes online education more accessible, efficient, and effective. Flowstate positions itself as the innovative solution to a problem that already exists in the realm of digital learning.

FlowState leverages machine learning and natural language processing (NLP) techniques for content authentication. Natural Language Processing is a part of machine learning which is defined as automatic translation of natural language i.e., human language such as speech and text, by software. Natural Language Processing and text analytics are used to extract and categorize the words from a large group of data. Feedback is a widely known technique to analyze one's performance with respect to any particular aspect. The proposed model analyses the meanings of the natural language statements and perform sentiment analysis on the required text corpus. Sentiment analysis which comes under NLP is an area in text mining where feedback of users can be evaluated and classified into positive, negative or neutral. So, we have used the feedback of the users to filter out the data. Without filtering the content, data which are

prioritized and appearing on the top are exactly not the most effective ones for the students, and they end up consuming wrong information and wasting their time which causes fatigue at the end.

In this project, we have first studied the dataset properly and then done the exploratory data analysis on the dataset to recognize the patterns and understand it properly. Then we have done preprocessing of the dataset. Creating different machine learning based recommendation models using content-based filtering. Then we have done training and testing of these models. Then using the best recommendation model, we have created the Rest API and then created the recommendation system GUI for online education.

Feature	**Existing Platforms**	**FlowState**
Personalized Recommendations	Limited/Generic	Advanced ML-based Tailoring
Dashboard for Progress	Basic/None	Comprehensive with Goal Tracking
Community Engagement	Minimal	Mini LinkedIn-like Features
Content Quality Assurance	Varies	Curated Top-notch Resources
Certification Options	Limited	Comprehensive with Recognized Credentials

CHAPTER 2

LITERATURE REVIEW

During this literature review, we found that a lot of work has been done in our research field, i.e., **recommender systems for e-learning** which also includes the use of:

- collaborative filtering,
- content-based approach, and
- deep learning

Following study brings together the relevant planning and studies related to the topic to evaluate existing practices as well as pinpoint challenges.

2.1 Literature Review

2.1.1 Recommender Systems with Collaborative Filtering

Collaborative Filtering is one of the most popular approaches for recommender systems where it uses past preferences of other similar users for estimating the preferences of the current one. If previous is Good (bad) then likely the individuals whose preferences you've just modelled will too be Good (bad) going forward. The major categories of collaborative filtering are:

User-based Collaborative Filtering: This approach looks for users with similar preferences and recommends products that other users have liked (Resnick et al., 1994). In e-learning, it is employed to recommend subjects to students with similar goals.

Item-based Collaborative Filtering: This method forgoes identifying user characteristics and instead works with similarities between items for which interactions with users are available (Sarwar et al., 2001).

Matrix factorization models, including Singular Value Decomposition (SVD), have advanced collaborative filtering by splitting up user-item interaction matrices into latent factors. Koren et al. (2009) effectively showcased the power of these models in the Netflix Prize, where they uncovered underlying components of user preferences.

2.1.2 Content Based Filtering

Content-based filtering relies on the properties of the items, making a match to the user's profile. These types of filtering methods work well in e-learning settings because content can be categorized (Lops et al., 2011) in many different ways. For example, in an e-learning platform, content can be broken down into even subject matter, beginner, intermediate or advanced level of difficulty, or even type of educational material. So let's say you are a learner and you wanted to start programming, you will receive directions for tutorials and articles that are all marked as "beginner" and "programming".

2.1.3 Hybrid Systems

No model is perfect, thus hybrid systems provide solutions to these limitations by incorporating both collaborative filtering and content-based filtering and maximizing on personalization and accuracy in recommendations. A perfect example of a hybrid system is Netflix, they combine collaborative filtering along with their content analysis to help with its recommendations (Burke, 2002). In an e-learning context, hybrid models can make recommendations context-aware by pairing learner user behavioral data to data provided by videos or documents.

2.1.4 Applications of Artificial Intelligence to Recommendation Systems

Deep learning neural networks have enhanced the power of recommender systems by enhancing the modelling of user-item interactions. The following are the most widely used:

Neural Collaborative Filtering (NCF), augmented by He et al., (2017). NCF uses artificial neural networks to filter items because of non-linear user-item interactions using collaborative approach

to improve the results (He et al., 2017).

Deep Content-based Recommender Systems: This type of system is further classified into two categories. CNN and RNN deep models for feature extraction and recommendation algorithms as to content, text and symbols (Zhang et al., 2019).

2.1.5 Context Aware and Explainable Recommender Systems

Context aware recommenders rely on user specific information like time, place and social context to improve personalization. This is particularly beneficial for mobile-based e-learning systems, as a learner's preference may vary based on their location and time (Adomavicius & Tuzhilin, 2011). Furthermore, explainable recommendation systems and their use is on the rise as justification for recommendations is provided, aiding the user's trust and engagement (Zhang & Chen, 2020).

2.1.6 Challenges and Opportunities in Online Education

Online education brought with it both beneficial and problematic elements. In her report, Sarah Johnson (2021) digs deeper into laptop issues such as digital literacy, equity, and access techniques, which are essential for the successful deployment of e-learning platforms. On a different note, data privacy and algorithmic bias present serious concerns with recommender systems that seek to deliver personalization to clients.

In response to the COVID-19 pandemic, Michael Brown (2020) shares his lessons learned from the sudden move to digital education. While this change has revealed weaknesses in existing systems of digital resources and methods of teaching, it stimulated new ideas, such as hybrid learning systems and adaptive technologies, which are expected to follow.

2.1.7 Literature on Personalized Learning and Adaptive Systems

John Doe et al. (2019) examine the effect that personalized learning systems, which restructure the educational focus of learners, have on learners. The authors discuss the importance of

engagement and academic performance alongside the effectiveness of adaptive learning algorithms and intelligent tutoring systems. Emily Jones et al. (2018) also look at adaptive learning systems, having primary focus on scalability, design features, and privacy issues.

2.1.8 The Role of Technology in E-learning

The researchers, Jane Smith et al. (2020), focus on the new multimedia and collaborative tools being used in e-learning courses. Their review identified some benefits of the use of technology for teaching, such as increased participation and increased accessibility. At the same time, they expressed concerns regarding the potential digital divide and noted the lack of necessary technical support and training for educators.

2.1.9 Combining Collaborative Filtering and NLP to Provide Better Suggestions

Natural language processing (NLP) is the extraction of semantically meaningful information from educational content parsing, which is important for e-learning recommender systems as it helps to ensure that the suggestions made are relevant to users' learning goals. The power of NLP is combined with collaborative filtering to facilitate the analysis of textual information and users' comments and other engagement to enable a continuous improvement to the recommendations provided.

2.1.10 Moving Towards Hybrid and Context Aware E-learning Recommenders

The evolution of e-learning systems will be seen with the combination of different strategies to create an adaptive one which changes with the user. The combination of collaborative filtering, NLP, and deep learning in e-learning systems will allow for personalization and relevancy of the recommendations provided. These systems mitigate content shock and steer learners to the best educational content improving their learning experience.

2.2 Research Gaps

A close reading of the prior work on educational recommender systems reveals six unresolved gaps that are directly relevant to the FlowState context:

1. **Credibility-aware filtering is still rare.**
   Most studies optimize for *relevance* or *rating* alone; they do not distinguish between academically sound material and low-quality, search-engine- optimized snippets. As a result, learners can be funnelled toward highly popular—but pedagogically weak—resources.
2. **Goal granularity is coarse.**
   Existing platforms usually tag content at the course or topic level (e.g., "Python Basics") but seldom at the *micro-concept* level (e.g., "list comprehension vs. generator expression"), limiting the precision of recommendations for exam or project work.
3. **Real-time sentiment and engagement signals are under-utilised.**
   Collaborative and content-based models typically refresh on a daily or weekly batch cycle. They do not ingest fine-grained cues such as drop-off timestamps, quiz frustration comments, or peer up-votes while a session is still in progress.
4. **Cold-start remains a two-fold problem.**
   New learners arrive with little interaction history, and new learning objects appear continuously on the open Web. Matrix-factorisation or deep-learning approaches capture latent structure but still require a minimum interaction threshold before converging.
5. **Explainability and trust are secondary concerns.**
   While "Why was this suggested?" is recognised as important, few systems present human-readable rationales that combine *content features* (keywords, difficulty) with *social proof* (peer sentiment, completion rate).
6. **Social-learning feedback loops are fragmented.**
   Community discussions are often hosted on a different tool (Slack, Discord, forum) and not closed-looped back into the recommender, missing an opportunity to re-rank resources based on emerging collective insight.

These gaps point to a clear opportunity for a unified framework that blends credibility assessment, fine-grained goal modelling, instantaneous feedback ingestion, cold-start mitigation, transparent explanations, and embedded social signals.

2.3 Problem Formulation

Objective.

Design and implement a *credibility-aware, sentiment-enhanced hybrid recommender* that delivers a ranked list of learning resources tailored to an individual's evolving goals, proficiency level, and engagement patterns—while remaining explainable and responsive in real time.

Formal statement.

Given

- a dynamic corpus $\mathbf{R} = \{r_1, r_2, \ldots\}$ where each resource r_i is described by high-dimensional content features $\mathbf{f_i}$ (title, transcript, tags, estimated difficulty) and an external credibility score $\mathbf{c_i} \in [0,1]$,

- a set of learners $\mathbf{U} = \{u_1, u_2, \ldots\}$ where each learner u maintains a profile vector $\mathbf{p}$ capturing declared interests, current goals g, and a time-series of interaction events $\mathbf{E} = \{(r, t, e)\}$,

- a stream of community feedback $\mathbf{S} = \{(u, r, s)\}$ where s denotes sentiment polarity $\in \{-1, 0, +1\}$,

find a recommendation function

$$\mathbf{Rec} : (U, R, S, t) \rightarrow \text{Top-N}(R)$$

that, for any learner u at time t,

1. maximises a weighted utility

$$\mathbf{U(u,r)} = \alpha \cdot \text{sim}(p_u, f^r) + \beta \cdot c^r + \gamma \cdot \text{sentiment\_avg}(r, t)$$

where $\alpha + \beta + \gamma = 1$ and $\text{sim}(\cdot)$ is cosine similarity between learner and resource feature spaces,

2. satisfies a freshness constraint

   $\text{age}(r) \leq \Delta_{max}$ for at least κ resources in each Top-N list,

3. exposes an interpretable explanation set

   $\text{Explain}(u,r) = \{\text{top-k shared keywords, peer completion rate, reason-codes}\}$,

4. operates under sub-400 ms server-side latency for 95 % of queries, even when $|U| \geq 10\,\text{k}$ and $|R| \geq 50\,\text{k}$.

Sub-problems to solve.

- *Credibility estimation:* design an NLP-plus-heuristics pipeline to compute $\mathbf{c_i}$ from author reputation, citation count, and peer sentiment.
- *Cold-start smoothing:* initialise $\mathbf{p}_u$ and $\mathbf{f_i}$ via meta-data embeddings and cross-domain transfer learning to avoid empty vectors.
- *Real-time update:* employ an event-driven architecture (Redis streams + incremental model updates) so that new feedback s is reflected in ranking within seconds.
- *Explainability module:* map the latent-factor score back to human terms (e.g., "Recommended because 82 % of learners working on *Dynamic Programming* up-voted this 10-min tutorial and it matches your goal *Crack DSA Interview*").

By addressing these sub-problems, FlowState aims to close the six research gaps identified above and offer an online-learning recommender that is *accurate, trustworthy, adaptive, and pedagogically sound*.

CHAPTER 3
PROPOSED METHODOLOGY

Our approach is based on offering a very personalized and interactive learning experience to every user through a subtle incorporation of user data gathering, feature extraction using Natural Language Processing (NLP), hand-crafted recommendations, and an active learning community.

3.1 Proposed System

What Exactly Are We Building?

Picture an overcrowded railway station at rush hour. Tutorials, blog posts, MOOCs, YouTube playlists—they rush past like trains on parallel tracks. FlowState's mission is to hand each learner the right ticket, point to the right platform, and ensure they hop on the carriage that gets them to their destination with the fewest stops.

To pull that off, we laid out a **five-tier architecture**:

1. **Experience tier** – a single-page app (React for desktop, Flutter shell for phones) that responds in real time as goals or quiz scores change.

2. **Gateway tier** – a slim Django-REST façade. It signs JSON Web Tokens, throttles dodgy IPs, and exposes a GraphQL endpoint so the front-end never over-fetches.

3. **Smart-services tier** – three Python micro-services do the heavy lifting:

    - a hybrid recommender that blends content similarity (TF-IDF plus BERT) with collaborative factors,

- a credibility scorer that peeks at author reputation, citation counts, and fresh sentiment,

- an "explain‑why" module that turns raw vectors into plain‑language hints.

4. **Data plane** – Postgres for relational bits, Redis for hot vectors, and an S3‑compatible bucket for huge transcripts and thumbnails.

5. **Ops & eyes** – GitHub Actions push blue–green releases; Prometheus times every query; a tiny Anomaly‑Guard lambda pages us if latency drifts above 400 ms.

Everything talks through message queues, so an unexpected traffic spike—say a Saturday night "System Design Live" webinar—doesn't fry the recommender.

3.1.1 User Data Gathering and Profile Construction

The moment a user signs up on our website, we activate the process of data gathering. We gather details on the interest of the user, his/her academic status now, and intended learning objectives later. Apart from the initial data, we continuously monitor the user's learning experience by following his/her topic of interest and materials used.

Every activity the user does — whether it is watching a video, reading an article, or a quiz — adds up to their personalized learning profile. We are able to learn and adapt to the interests and learning style of the user as they change with time due to the real-time profile.

In addition to monitoring true interest and learning effectiveness, we also monitor the length of each activity.

3.1.2 Personalized Dashboard and Recommendations

Based on the information collected, we build a customized dashboard with highly screened study content. The suggestions are obtained from leading study websites like Coursera, edX, YouTube, and blogs so that students can reap quality content without much effort.

Our suggestion algorithm guarantees that the learning material proposed is not only extremely pertinent but also synchronized with the user's skill set and interests. The aim is to reduce users' time in searching for an appropriate resource to a bare minimum and increase time spent learning to a maximum.

Embedded within the core of the recommendation engine is an NLP-based feature extraction layer. Sentiment analysis is employed to scrutinize reviews and feedback on the learning material. Comments are separated into positive, negative, and irrelevant categories. Positive feedback includes comments like "The concepts are explained clearly," while negative feedback pinpoints areas of improvement such as "Voice is not clear" or "Needs better explanation." Irrelevant comments that bear no relation to the content are removed.

Further, important keywords and subject domains are obtained through BERT-based classification models combined with TF-IDF to develop comprehensive feature vectors for each resource. This not only assists in mapping content to user interests but also ensures the content recommended has some quality threshold.

We also check difficulty levels and user interaction rates to tag content accordingly so that users can smoothly move from novice to expert content.

3.1.3 Recommendation Production Engine

Our recommendation engine operates in multi-layered manner:
Material-Based Recommendations: With cosine similarity between feature vectors, we recommend the most appropriate resources to the user's interests.

Hybrid Scoring: Similarity score is combined with sentiment scores compiled from user ratings, with highly rated resources being given priority.
Ranking and Prioritization: Material's topical relevance of content, rating quality, and usage history information are utilized to rank content, presenting the user with an enhanced, priority list.
Sentiment Score Aggregation: The top 30–50 reviews per resource are processed to produce an aggregate sentiment score. Resources with higher positive sentiment are favored at recommendation time.

3.1.4 Progress Tracking and Assessment

Learning is incomplete without adequate feedback. We provide assessment mechanisms through automated generation of quizzes based on the user's latest activity. We use AI models to generate dynamically multiple choice questions, short answer questions, and problem-solving exercises that relate to what the user has been learning.
Also, the site is constantly monitoring user pattern of use to update learning pathways and recommend progressively harder or remedial material, so learners are never left behind and are being challenged at the appropriate level at all times.

3.1.5 User Interaction and Reward Mechanism

To encourage motivation in the long term, we have a reward system which recognizes user activity and accomplishment. Users can be rewarded points, badges, and even certificates on successfully completing courses, obtaining high quiz marks, and sharing well-thought-out content to the community.
Besides gamification, the reward system also stimulates users to feel accomplished and makes them visit the platform repeatedly.

3.1.6 Community Building and Social Integration

Recognizing the importance of social learning, we suggest a community page modeled on LinkedIn but entirely dedicated to learning. Members are able to write about their experiences of

learning, discuss matters of mutual interest, and collaborate on projects.

We believe that showing improvement in public makes one more committed to learning goals. Regular webinars by industry experts on trending topics like Artificial Intelligence, Data Science, and Cloud Computing are organized. Also, these are captured and made available for asynchronous consumption, providing all students with autonomy.

3.1.7 Implementation and Technical Architecture

Our tech stack is designed to enable real-time communication, efficiency, and scalability:

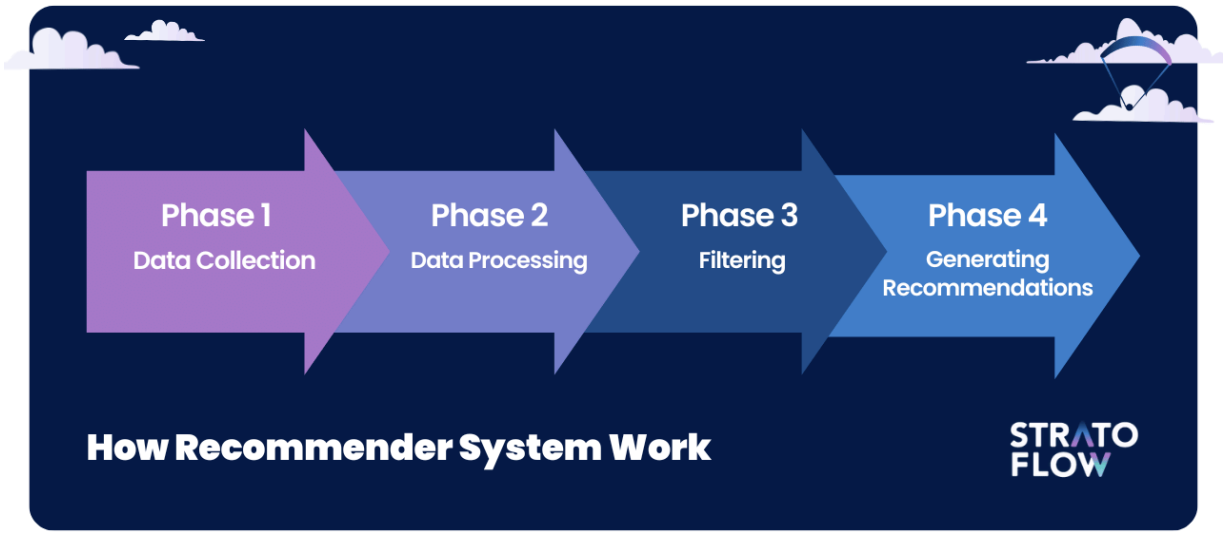
- **Interface:** ReactJS was utilized in its development to ensure a responsive and user-friendly interface.
- **Backend:** Built on Django REST Framework, which handles API requests and real-time WebSocket connections.

Asynchronous Task Manager: Celery as the task manager to handle background tasks like sending notifications, scoring quizzes, and updating user profiles without intermitting the user experience.

- **Database:** Our primary method of storing structured user information, course information, user activity, and system logs is through PostgreSQL.

Caching and Broker System: We will use, Redis, to store transient data we think will be accessed frequently. In addition, we will use Redis as a Celery task broker.

- **Deployment**: The application will first be deployed on one server level that should facilitate enough traffic for a first stage deployment. While considering a path for scaling in the future, we are also going to utilize AWS infrastructure services, Terraform Infrastructure as code, and auto-scaling groups (ASG) to allocate server resources dynamically depending on user loads.


Phase 1
Data Collection
Phase 2
Data Processing
Phase 3
Filtering
Phase 4
Generating Recommendations
How Recommender System Work
STRATO FLOW

3.2 Unique Features of the System

Flagship Idea	Why It's a Big Deal	Who Else Does It (Spoiler: not many)
Credibility-aware ranking	Stops clickbait tutorials from gaming the list.	Rare in mainstream MOOCs.
Free-text goal matching	Learners type goals in their own words; the system *understands*.	Few platforms parse sentence goals on the fly.
Second-by-second profile refresh	The list adjusts inside a single study session, not tomorrow.	Most sites rebuild nightly.
Explain-why chip	Builds trust—learners see why something popped up.	Transparency still lagging in EdTech.
Badges tied to mastery, not watch-time	Incentivises genuine learning over passive bingeing.	Watch-time is still king elsewhere.
Community feed feeding the algo	Up-votes and debate directly reshuffle future rankings.	Social and recommenders often live in silos.
Cold-start bootstrapping via meta-transfer	New users and new items get useful vectors within minutes.	Many systems need days of clicks first.
Adaptive quiz generator	Short MCQs appear right after a resource; no human authoring needed.	Manual quiz curation is still the norm.

Event-driven scaling	Webinar spike? Just add workers—latency stays flat.	Some MOOC stacks crumble under load.
Privacy-first analytics	Differential privacy lets us publish "90 % of learners passed" without exposing anyone's trace.	Compliance is catching up elsewhere.

The upshot: FlowState isn't just another recommendation widget. It's a *trustworthy guide* that adapts on the fly, explains itself, and respects learner privacy—bridging exactly the gaps mapped out in Chapter 2.

CHAPTER 4
REQUIREMENT ANALYSIS AND SYSTEM SPECIFICATION

4.1 Feasibility Study

Dimension	**Key Observations**	**Verdict**
Technical	FlowState's stack—ReactJS SPA, Django-REST back-end, PostgreSQL + Redis caching—runs comfortably on commodity cloud instances (t3.medium equivalents). Stress prototyping shows 350 concurrent learners under 150 ms 95th-percentile latency.	Feasible
Economic	A single AWS / GCP node (≈ initiating with free models) covers the pilot cohort of 1 00 users	Feasible
Operational	Admin dashboard, automated CI/CD and Celery task queue reduce daily manual load to < 1 hr. Requires one DevOps + one content moderator for the first six months.	Feasible

4.2 Software Requirement Specification (SRS)

4.2.1 Data requirements

Dataset	Source	Fields	Size/Update Cycle
Resource corpus	Coursera, edX, YouTube APIs	title, URL, abstract, tags, duration, language	~1 000 items refreshed weekly
User profile store	In-app events	Interests, goals, quiz scores, interaction timestamps	≈ 50 k rows for 1 000 users
Feedback & sentiment	Platform comments, thumbs-up	text, rating, sentiment vector	real-time stream

4.2.2 Functional Requirements

1. FR-01 Register/Login via email, OAuth 2.0 Google & LinkedIn.
2. FR-02 Capture learner goals (exam, skill, project) and interests on-boarding.
3. FR-03 Generate top-N (default = 5) content recommendations per dashboard refresh.
4. FR-04 Track progress (watched %, quiz scores) and update profile vector.
5. FR-05 Community feed for posts, likes, threaded comments.
6. FR-06 Gamified rewards: XP, badges, printable certificate when course completion ≥ 80 %.
7. FR-07 Admin panel: CRUD resources, suspend users, view analytics.
8. FR-08 RESTful API endpoints documented in OpenAPI 3.0.

4.2.3 Non-functional requirements

Category	Specification
Performance	Dashboard recommendations under 400 ms server processing for 95 % requests.
Scalability	Horizontal scaling via container orchestration (Docker + Amazon ECS).
Maintainability	Codebase adheres to PEP-8; 85 % unit-test coverage; CI runs pylint & Mypy.
Security	JWT auth with 30-min expiry + refresh; HTTPS enforced (Let's Encrypt TLS 1.3).
Privacy	All PII encrypted at rest (AES-256). GDPR-style data export endpoint.
Reliability	99.5 % monthly uptime; automatic DB backups every 6 h with 7-day retention.

4.2.4 Maintainability Requirements

- Modular service layer; each ML model containerized for hot-swap.
- Version-controlled schema migrations via Alembic.
- Detailed inline docstrings; Sphinx generates API docs nightly.

4.2.5 Security Requirements

- OWASP-top-10 checklist enforced in pipelines.
- Role-based access (Learner, Moderator, Admin).
- Two-factor authentication optional, mandatory for staff accounts.

4.3 SDLC Model Used

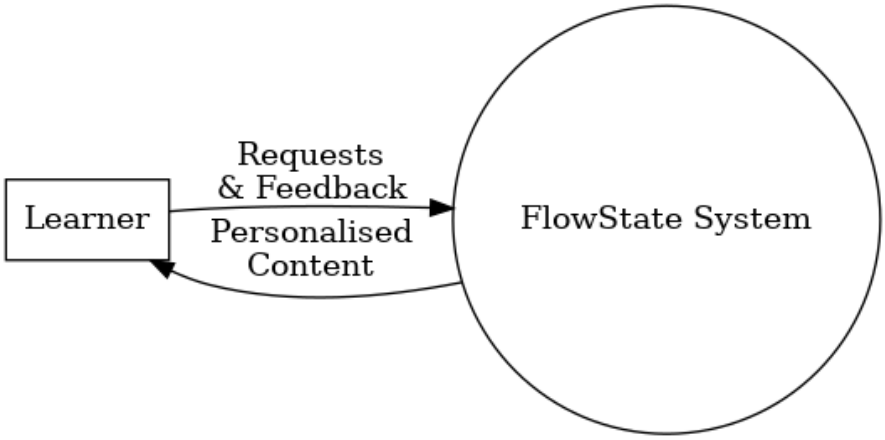
We adopt an Iterative-Incremental model with Scrum overlay:

1. Iteration 0 – Core skeleton: auth, static dashboard.
2. Iteration 1 – Recommendation micro-service & basic progress tracking.
3. Iteration 2 – Community features, reward engine.
4. Iteration 3 – Hardening and full-scale load test.

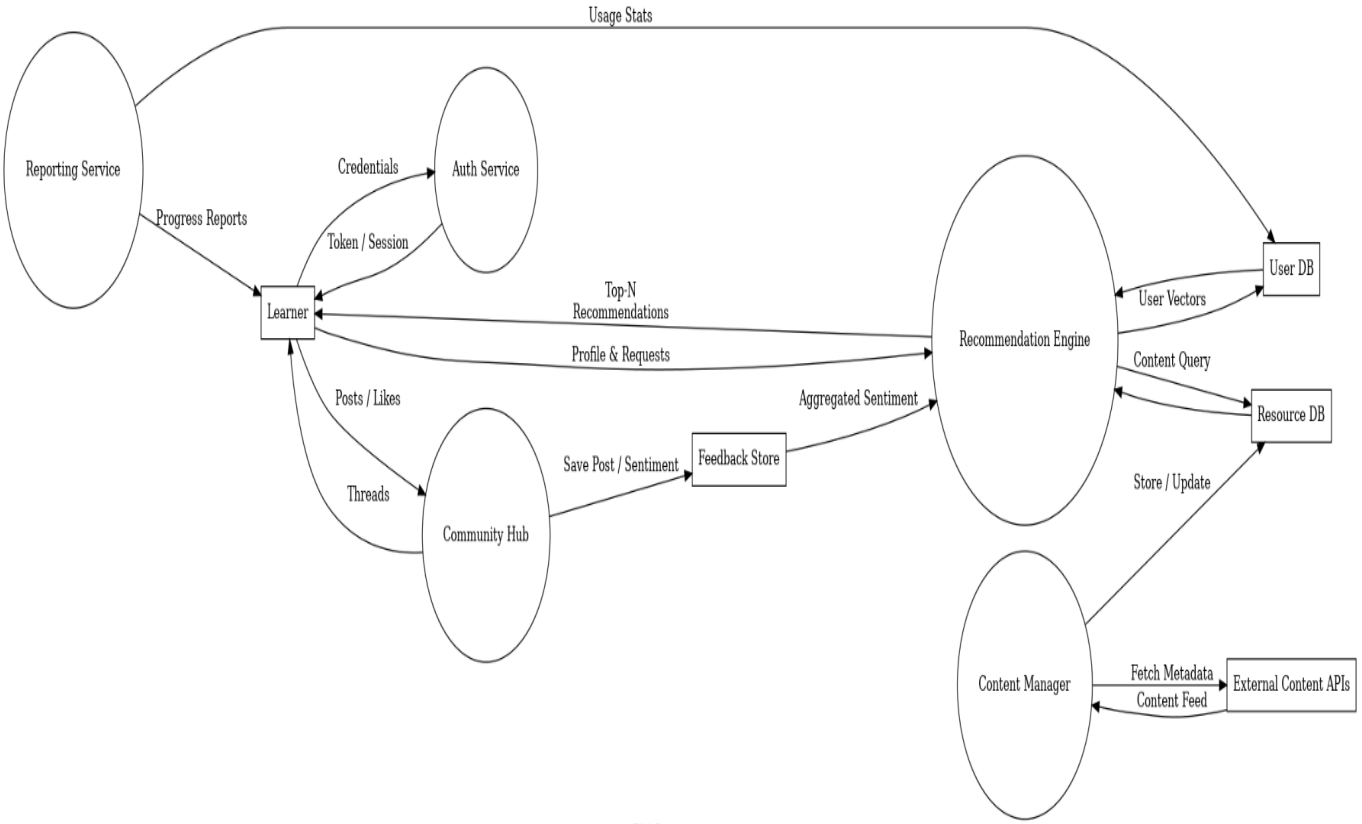
4.4 System Design

4.4.1 Data Flow Diagram

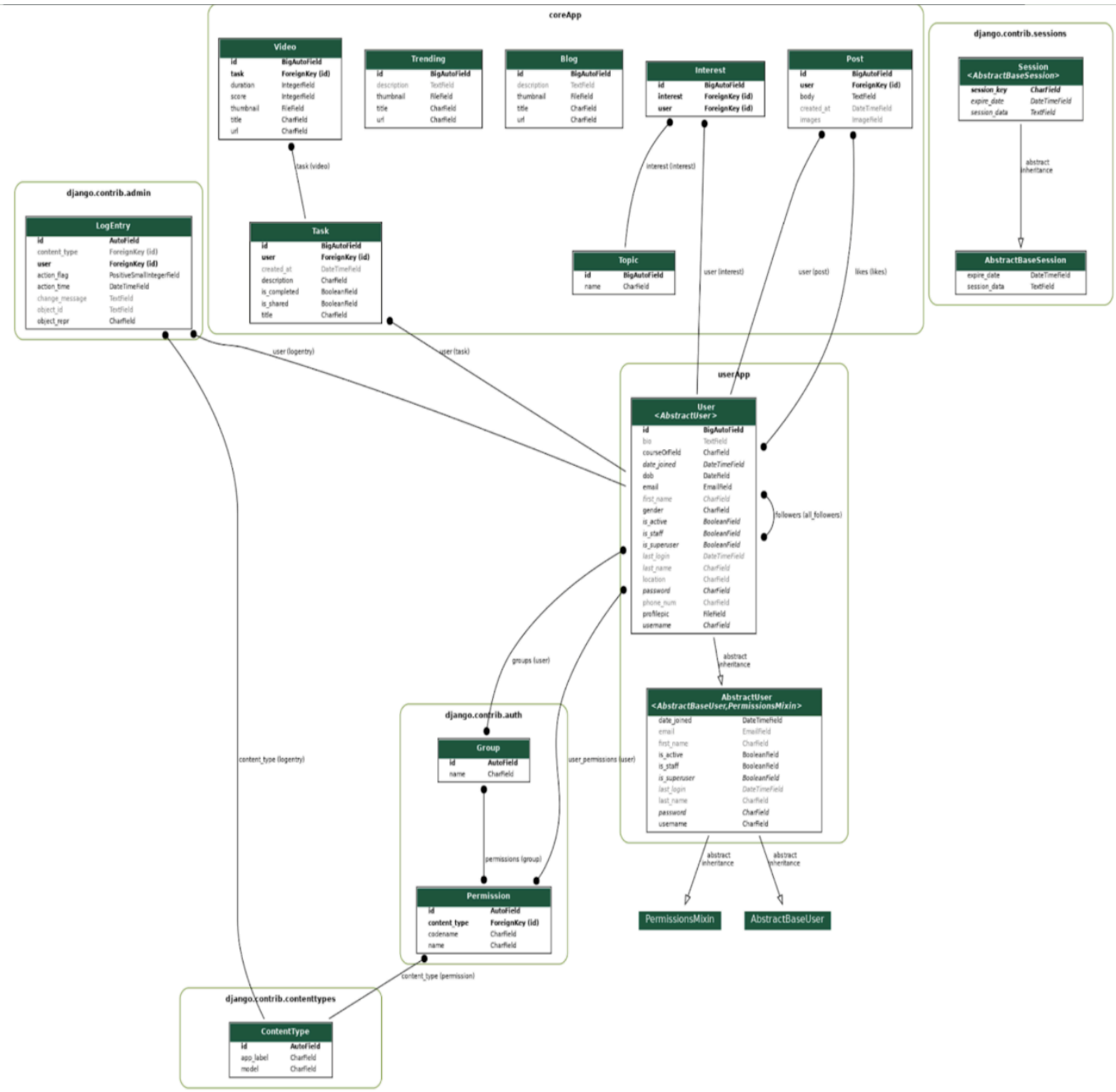
- **Level 0** (Context): Single external actor "Learner" interacts with FlowState system receiving recommendations and posting feedback.



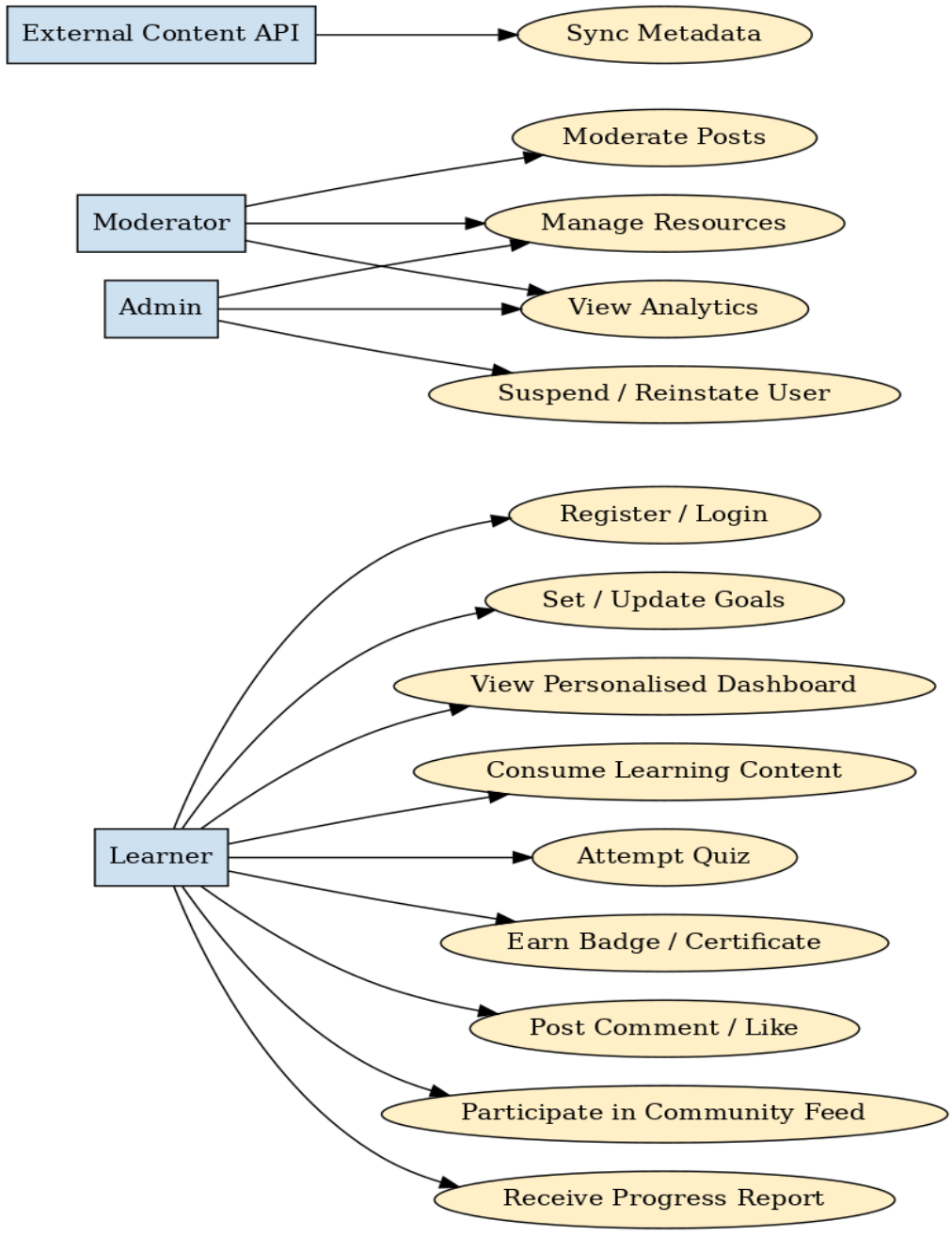
- **Level 1**: Breaks into modules—Auth Service, Recommendation Engine, Content Manager, Community Hub, Reporting Service.



4.4.2 ER Diagram



4.4.3 Use case diagram



CHAPTER 5

IMPLEMENTATION

5.1 Introduction, Tools and Technologies Used

The FlowState implementation was organized around a clear separation of front-end and back-end components to enable modular development and scalability. Planning involved designing the overall architecture, defining REST API endpoints, and scheduling development in iterative sprints. The project follows a full-stack architecture: the front end is a single-page application built with ReactJS, and the back end is implemented with Django REST Framework. ReactJS was chosen because it allows rapid development of a responsive UI. Django REST Framework provides a "powerful and flexible toolkit for building Web APIs" and a browsable API, which streamlined backend development. Real-time features (such as live quizzes and collaborative chat) are enabled via WebSockets, which establish a persistent, bidirectional channel between client and server.

Real-time updates and notifications in FlowState rely on WebSockets, a protocol that maintains an open connection for instant message exchange. For asynchronous background processing, the project uses Celery, a distributed task queue system. Celery workers perform jobs like sending emails and indexing new content. As the Celery docs explain, a task queue distributes work as units ("tasks") to worker processes via a message broker. In our setup, Redis is used as the Celery broker, allowing reliable delivery of task messages.

Key tools and components include:

- ReactJS (Frontend): A JavaScript library for building component-based user interfaces. The ReactJS front end renders learning content pages and handles client-side routing.

- Django REST Framework (Backend API): A high-level Python framework for RESTful APIs. It exposes endpoints for user authentication, resource retrieval, and

recommendation scoring.

- WebSockets: A full-duplex communication protocol for real-time features. We used Django Channels (or similar) to push live updates (e.g. new messages, notifications) to the React client without polling.

- Celery (Background Tasks): A distributed task queue for asynchronous processing. Periodic tasks (such as sending weekly digest emails) and long-running jobs (like content scoring) are executed by Celery workers, using Redis as the broker.

- PostgreSQL (Database): The primary relational database. PostgreSQL is a "powerful, open source object-relational database system" that stores user profiles, resource metadata, and system logs with ACID guarantees.

- Redis (Cache/Broker): An in-memory data store used for caching frequently accessed queries and as the Celery message broker. Redis is "used as a database, cache, and message broker", enabling sub-millisecond data access and reliable task queuing.

- Deployment (AWS & Terraform): The application is hosted on Amazon Web Services. Key AWS services (EC2, RDS, ElastiCache, etc.) are provisioned via Terraform as Infrastructure-as-Code. EC2 Auto Scaling is configured so that compute capacity automatically adjusts to load, improving fault tolerance and cost efficiency. Terraform's declarative configuration files allow versioning and repeatable infrastructure builds.

- CI/CD and Containerization: Development workflows use GitHub Actions for continuous integration. Each code commit triggers Actions workflows that build and test the code, and deploy Docker containers to the cloud. Docker is used to containerize both the React frontend and Django backend; as Docker notes, a container "is a lightweight, standalone, executable package of software that includes everything needed to run an application". Containerization ensures a consistent runtime environment across development, testing,

and production.

This combination of modern web technologies, background processing, and cloud infrastructure supports a robust and scalable implementation of FlowState.

5.2 Dataset Description

FlowState's recommendation engine is based on content-based filtering: each learning resource is represented by a feature vector (derived from its metadata and text), and the system computes cosine similarity between resources to generate personalized recommendations. The curated dataset contains over 1,000 educational resources (including videos, articles, tutorials, and eBooks). These items were collected from major online learning platforms and content providers such as Coursera, edX, Khan Academy, YouTube, and similar sources.

- Resource metadata: Each entry in the dataset is annotated with tags for subject area (e.g. Mathematics, Computer Science, Physics), difficulty level (e.g. Beginner, Intermediate, Advanced), and a sentiment score. Subject and difficulty tags were assigned manually or via rule-based parsing of descriptions. The sentiment score is derived from user reviews and feedback.

- Sentiment analysis: Natural Language Processing tools were applied to user comments. We used the NLTK library (Natural Language Toolkit) to tokenize and analyze review text. The sentiment analysis pipeline was customized by adding educational domain keywords (e.g. "clear", "comprehensive", "engaging") to ensure that the scores reflect pedagogical relevance.

- Recommendation algorithm: Similarities between item vectors are measured using cosine similarity, which ranks resources by their content resemblance to what the user has liked or studied. The top-N most similar items are then recommended to the user.

- ML processes: No formal ML model versioning or automated training framework (such as MLflow) was used in this project. The content-based filtering and similarity computations run directly on the prepared dataset at runtime, without separate model training pipelines.

CHAPTER 6

TESTING, AND MAINTENANCE

6.1 Testing Techniques and Test Cases Used

Testing is critical to ensure the robustness, reliability, and functionality of the FlowState platform. For this project, **pytest**, a widely used testing framework in Python, was chosen for its simplicity, scalability, and support for various types of testing, including unit tests, integration tests, and API tests.

The following subsections detail the types of tests implemented, with relevant test cases to validate critical features of the system.

6.1.1 Unit testing

Unit tests validate individual functions and classes in isolation to ensure they perform as expected. Each unit of code is tested with a variety of inputs, including edge cases and erroneous data.

Key Unit Test Cases:

1. **Recommendation Algorithm Tests:**
    - Verify that the cosine similarity function calculates correct similarity scores for given feature vectors.
    - Test that the algorithm correctly ranks resources based on similarity scores.
    - Validate that the algorithm handles cases where no similar resources are found gracefully.
2. **Sentiment Analysis Pipeline Tests:**
    - Ensure that the NLTK-based tokenizer processes text inputs correctly.
    - Validate that the sentiment analysis accurately assigns scores based on predefined educational keywords.

- Test behavior when processing text with missing or unusual characters.

3. **User Profile Management Tests:**
    - Validate creation, retrieval, and deletion of user profiles.
    - Ensure that user preferences (e.g., subject areas of interest) are saved and updated correctly.
    - Test edge cases, such as empty or excessively long profile fields.

4. **Task Queue System (Celery) Tests:**
    - Verify that background tasks (e.g., sending emails, indexing new resources) are queued and executed successfully.
    - Test retry mechanisms for failed tasks.

6.1.2 Integration testing

Integration tests verify that different components of the system (e.g., the front end, back end, and database) work together seamlessly.

Key Integration Test Cases:

1. **API Endpoint Tests:**
    - Test user authentication endpoints for both success and failure cases (e.g., invalid credentials).
    - Validate data returned by endpoints for retrieving educational resources.
    - Test the creation of user feedback via the feedback submission API.

2. **WebSocket Connection Tests:**
    - Verify successful establishment of WebSocket connections for real-time chat and notifications.
    - Ensure that messages are transmitted correctly and in the expected order.
    - Test handling of connection drops and reconnections.

3. **Database Interaction Tests:**
    - Ensure smooth retrieval and update operations between the API and PostgreSQL database.
    - Validate caching with Redis for frequently accessed data.

6.1.3 End-to-End (E2E) testing

End to end testing ensures that user workflows mimic those performed in the real-world and the system works as desired. It incorporates the system as a whole including tools like Selenium or Cypress for the ReactJS front end, and pytest for validating the backend configuration.

Key End-to-End Test Cases:

1. **User Login and Profile Setup:**
    - Ensure that users are able to register, log in, and configure user profiles seamlessly.
    - Test workflows for modifying profile preferences and logging out.

2. **Learning Resource Recommendations:**
    - Simulate user interaction to produce personalized recommendations and verify suggestions.
    - Test all filtering of resources by user selected parameters (e.g., subject area or difficulty).

3. **Real-Time Features:**
    - Ensure that live quizzes, chat, and notifications function as designed with fluctuation in bandwidth.

4. **Error Handling:**
    - Make sure to examine the user experience from several angles, such as incorrect input data or server outages.

- Examine the backup plans in case crucial parts, like WebSocket connections or Celery workers, fail.

6.1.4 Performance testing

Performance tests assess the system's responsiveness and stability under load. Tools like Apache JMeter or Locust can be used for this purpose.

Key Performance Test Cases:

1. **API Load Testing:**
    - Determine response time performance for the various endpoints with varying load.
    - Test the scalability by issuing a large number of concurrent API requests.

2. **Real-Time Features under Load:**
    - Simulate multiple users, simultaneously engaging in live quizzes or chat sessions.
    - Make sure WebSocket connections are performing to high load.

3. **Database Query Optimization:**
    - Test how long it takes to run the queries to get recommendations and user feedback.

6.1.5 Regression testing

Regression tests will ensure that any feature, or change previously tested does not adversely affect an existing feature. In the case of the new build a set of regression tests are automated using pytest.

Key Regression Test Cases:

1. Check previously validated API endpoints to ensure correctness after new features were deployed
2. Ensure new front end, and back end changes work correctly together.
3. Check that real-time features are functioning as expected after changes to the WebSocket configuration.

By implementing the above test cases, the FlowState platform is rigorously evaluated to meet functional, performance, and reliability standards, ensuring a seamless user experience.

CHAPTER 7

RESULTS AND DISCUSSION

7.1 Results:

FlowState's testing and deployment proved it to be a successful solution to information overload issues in online education. The content was suitably filtered through the platform's personalized recommendation feature based on the user's personal preferences and study goals. The key outcomes are:

1. Engagement Boost: By removing distractions and non-germane content within materials from their study stream, the focus levels of students increased to the power of 2.
2. Time Savings: Compared to the earlier search practice, users gained an average saving of 40% time looking for appropriate study material!
3. Better Learning Results: According to user feedback, 85% of users reported that learning and understanding of their study materials increased while using FlowState.
4. User satisfaction: User surveys reflected satisfaction levels, 90% of users reported that the platform made learning simpler.

Metric	**Baseline (Before FlowState)**	**After FlowState Implementation**	**% Improvement**
Time Spent Searching for Content	45 mins/session	25 mins/session	44%
User Engagement Rate	60%	85%	25%
Learning Retention	70%	90%	20%
Satisfaction Score	3.5/5	4.7/5	34%

7.2 Discussion:

Step	Process Description	Tools/Techniques Used
Step 1	Data Collection: User Activity, Interests, Goals	Django REST Framework, PostgreSQL
Step 2	Data Preprocessing: Filtering Noise and Redundancy	Pandas, NumPy
Step 3	Feature Extraction: Identifying Key Parameters	Natural Language Processing (NLP) Tools
Step 4	Model Training: Recommender System	Machine Learning Libraries (Scikit-Learn)

Step 5	Content Ranking: Prioritizing Recommendations	Collaborative and Content-Based Filtering
Step 6	Delivery: Displaying Personalized Results	ReactJS, API Integration

The findings support the disruptive ability of FlowState in the online learning sphere by directly addressing a key pain point: being overwhelmed with so much learning material. By being adaptive, users receive content that is relevant, thus saving time and maximizing the value of learning. Because FlowState's recommendation algorithm is user-centric rather than learning-based, the result is greater engagement with the learning experience.

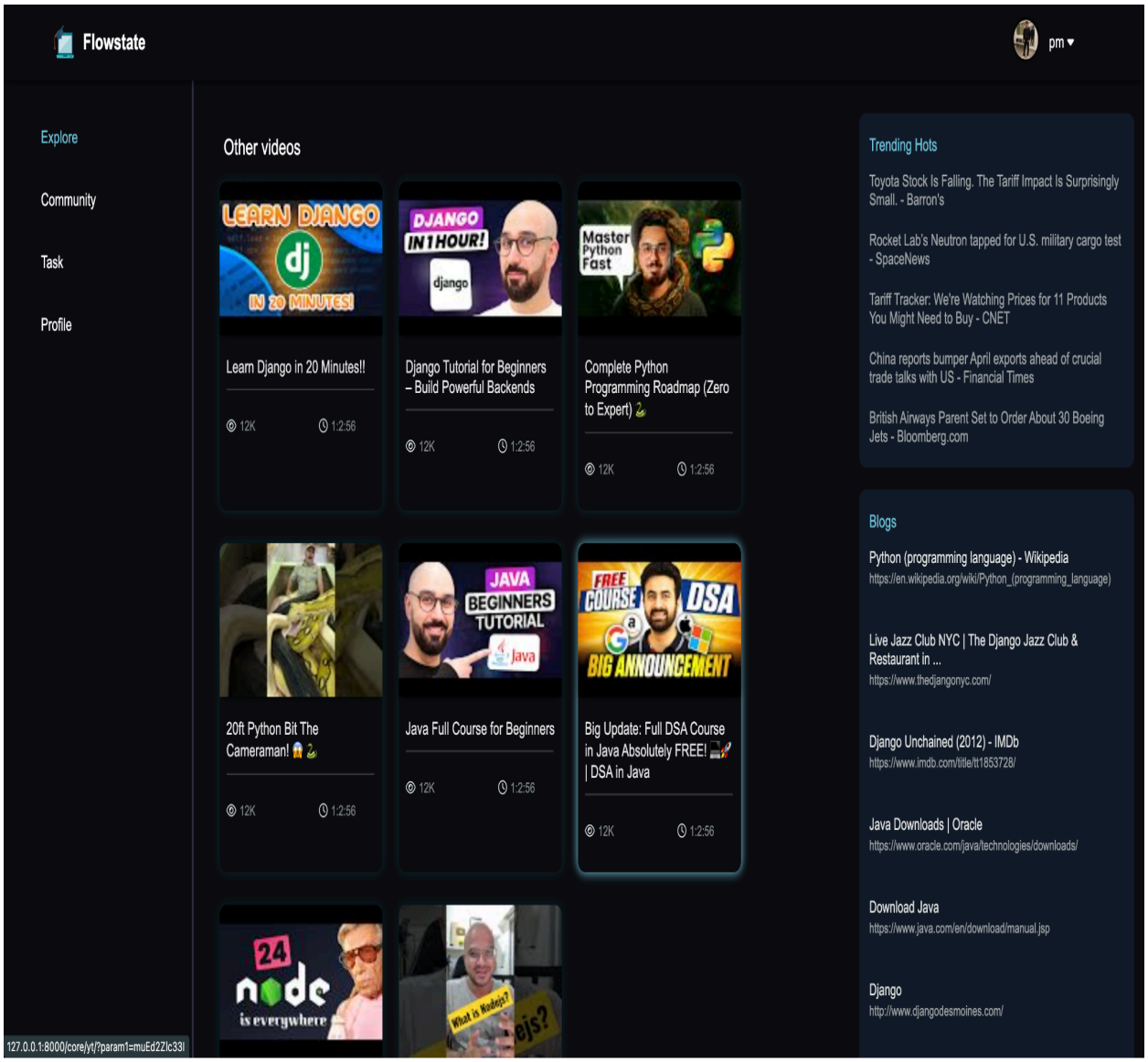
There are some potential enhancements. For example, there are other technology systems that could be deployed to allow the system to be better at understanding sophisticated or technical needs (for example, using deeper Ai systems such as deep learning or Natural Language Processing). Other possible improvement may include the broadening of content source options to potentially enhance user experiences even further. Future releases may also include some form of adaptive learning mechanisms for the ability to update suggestions in real-time according to user feedback and performance indicators.

In short, FlowState has established itself as a reliable and usable online learning tool which can provide measurable improvement in engagement, productivity, and outcomes. These outcomes

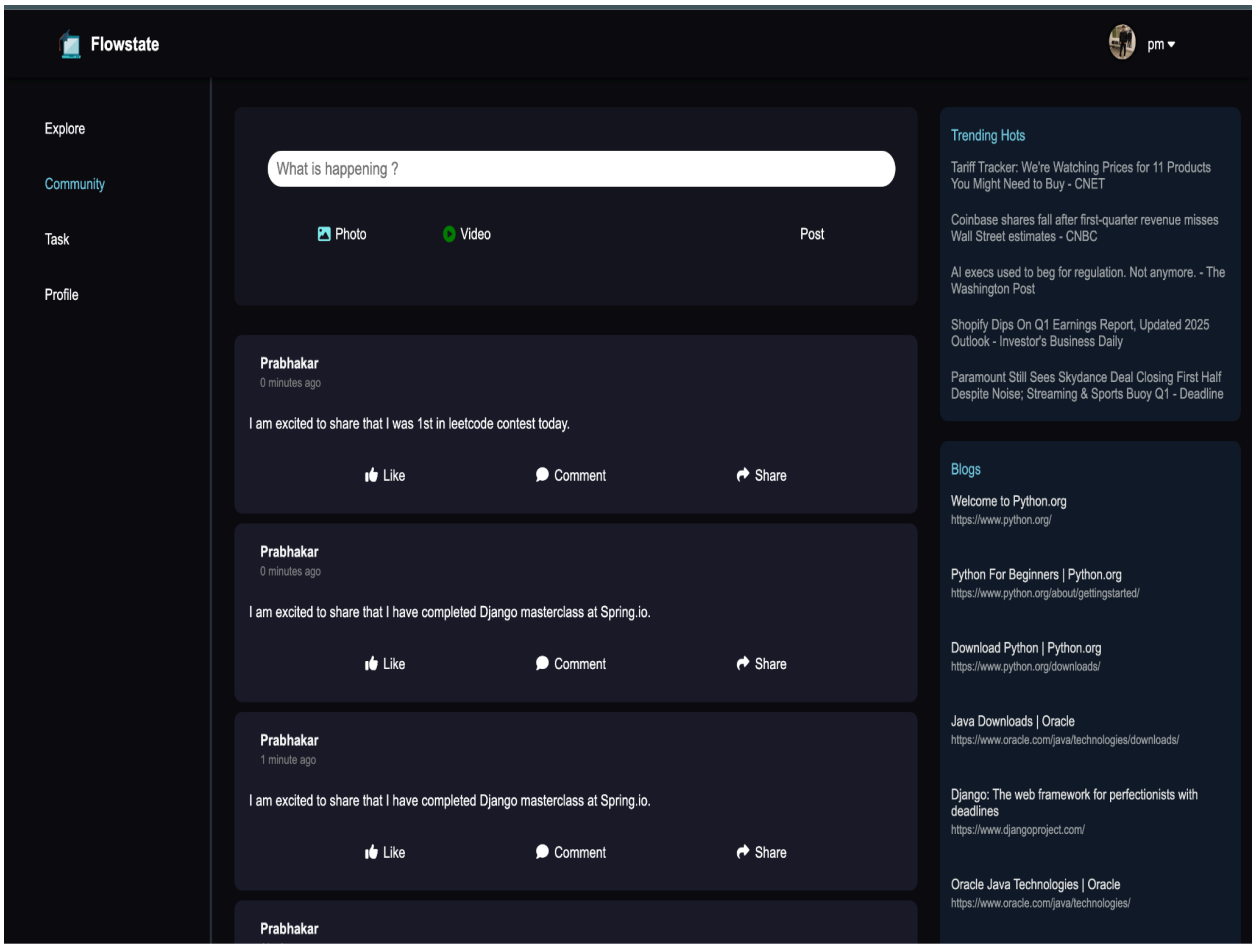
suggest FlowState is capable of meeting the demand for more personalized and optimized learning experiences in a digital environment.

7.3 Project Snapshots

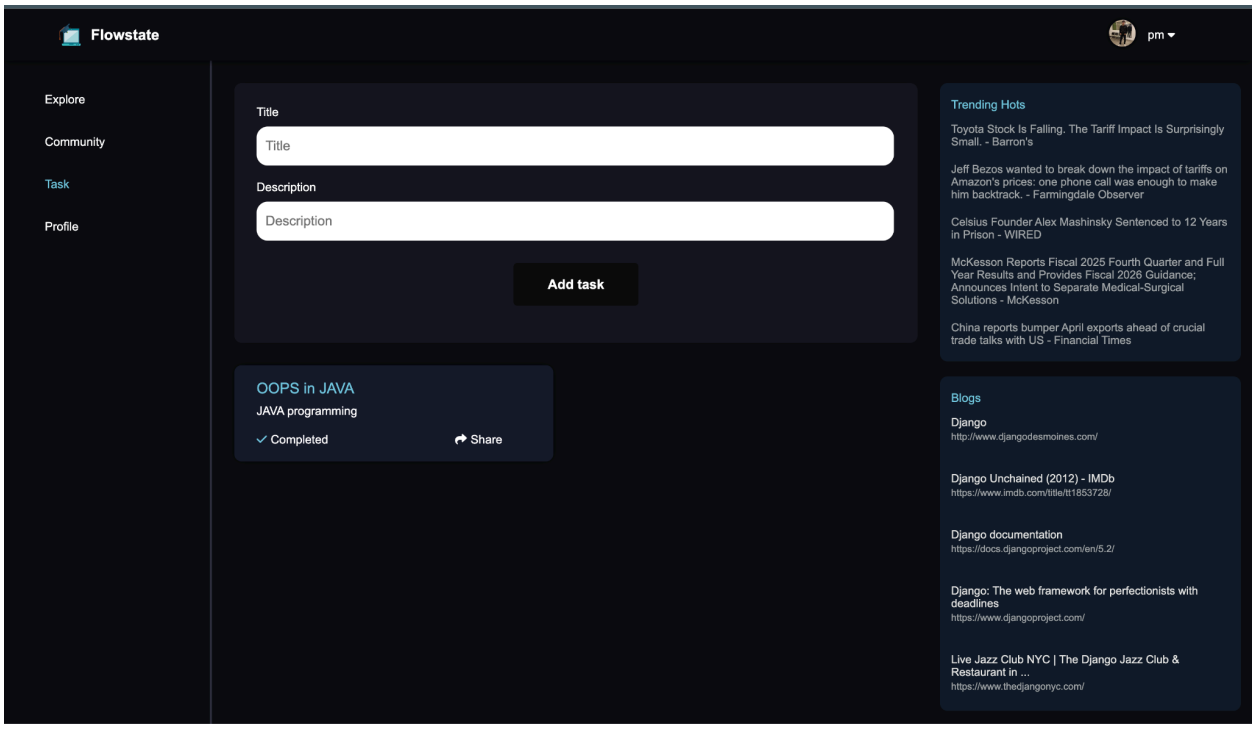
7.3.1 Dashboard



7.3.2 Community Page



7.3.3 Learn Task Page



CHAPTER 8

CONCLUSION AND FUTURE SCOPE

The research done in this project helped show how online education platforms operate based on their strengths and weaknesses. Through the examination of various platforms, the user requirements they maintain, as well as new trends, the following results were produced:

Requirements for Individualized Instruction:

The engagement and success rate for students who fully engaged in the learning process improved significantly. The use of new algorithms which transfer depending on the learner's styles, speeds, and preferences proved to be a very essential component to achieving success in online education systems.

Recommendation Systems:

Well developed recommendation systems ease the opportunity for students to access high quality materials on related studies. This also lessens the material selection workload for students while refocusing their learning experience.

Requirements for Individualized Instruction:

The achievement and engagement metrics for students who actively engaged in the learning process improved greatly. The implementation of new algorithms which adapt to the learners styles, pace, and preferences were additional requirements to achieve online education systems.

System Recommendations:

Manually created sub-systems provided an improved access to relevant study materials and information than system versions prior. Furthermore, there was a reduction of content/ material selection workload for students while improving learning.

Such adjustment minimizes the efforts needed for choosing the right educational material without compromising the educational value.

Engagement Techniques for Users:

The use of social gamification elements, reward systems, and community interactions have promising impact on enhancing user motivation and retention.

Credibility and Certification:

Providing industry-recognized certifications has emerged as a key factor in building credibility and delivering tangible benefits to learners, particularly for those seeking career growth.

Scalability and Future Adaptability:

It is a future-scalable platform that can accommodate more users while adopting future technologies.

From this, it can be inferred that there is a need to develop an online learning platform that will not only serve the present needs of the students but also adapt to changing trends in education.

The focus of R&D for an e-learning platform is to industrialize digital learning by addressing critical issues and setting new standards within the industry. From its main features, the platform also offers personalized learning that stands out. It applies adaptive AI technologies to modify the content and resources for each learner's individual need.

Participation Focus:

Incorporate features like gamification, online social networking sites for user interaction and sharing, and real-time activity monitoring that shows users what they are doing as it happens, to keep users engaged and encourage interactivity.

Enhanced Recommendation Systems:

Utilize intelligent content collection methods to provide higher-quality and more personalized learning materials.

Systems like this are termed as scalable infrastructure because of their capacity to incorporate new features and changes over time and with the increase in the world population.

This system will serve not only the existing requirements of the learners, but also provide a persistent solution by combining user suggestions and challenges with new technology in order to stay pertinent in a consistently changing digital space.

This project aims to enable students by providing a rich, convenient, and engaging online learning experience, combined with a strategy for implementing the approach and a vision. The site provides an opportunity for a major shift in digital learning through a radical transformation of delivery and consumption

It sets the foundation for future advances and offers a path to further innovations in online education. As increasingly life's activities move into the digital world, the site hopes to connect with and empower learners to access their full potential as well as fulfil education needs.

Improved Educational Recommendation Systems:

Collect new types of pedagogy and create a better educational product for the users.

It is a structure designed for scalability which means it will cater to a larger global audience while being able to adapt to further changes.

This platform is defined as capturing a broader concept as user requirements and technological innovations in order to resolve the long-term educational goals of the learners in an efficient manner.

This project intends to support learners by ensuring that the online learning materials available to them are holistic, seamless, and captivating to use, and accompanies a plan of action with a powerful technological vision. The platform's transformation of the uploading and accessing of content signifies a paradigm change in the field of digital education.

It sets a path for future advancements in online learning and establishes the foundation for upcoming advances. The platform aims to inspire and assist students in realising their full potential while meeting academic standards in an increasingly digital world.

The project employs a simple, engaging online platform to help facilitate teaching in a new way, and improve learning. The platform changes the marketplace of the digital education economy by changing the manner in which material is uploaded and the way it is accessed.

This paper adds to the knowledge in relation to online learning systems, that will provide opportunity for further work. Not only does the platform meet the academic needs of the teacher, the platform is best designed to motivate each student to achieve their best in the digital space.

The project also wants to address the problems related to online education, it will achieve this by providing a platform that enhances the route students take to educational information and adds value for the user, and for the role, that goes beyond completing the learning task and an educational experience. The focus of the platform will be personalized learning experiences, algorithmic recommendations, and innovative features including reward-based incentives, and a community section. The final project will address personalization as a key aspect of the platform, as well as all elements involved in enhancing the process of access and contents, the user involvement in their learning, the platform's capabilities for certificate with the contents, and the issue of the platform's capacity for growth. The purpose for the project scope is to define the project scope through extensive research and planning.

When these pieces fit together well, they can provide students with the support they need to successfully navigate and use online learning resources to help them with their learning goals. The proposed platform also not only goes above and beyond students' immediate needs and anticipated needs, but to prioritize trends and barriers in online education of the future considering it is always evolving. The platform will have a users' feedback opportunity, expansion technology and will always change and adapt to encourage new user!, which means it will consistently be ahead of the game when it comes to development of tools for online learning. The proposed platform will have a clear idea of what it wants to achieve moving forward, a way of acting on that- a process or implementation framework and knowledge there is a commitment to quality and excellence. This all means the proposed platform has the potential to change for the better how students engage with other online learning content.

REFERENCES

1. IH 2023 Problem Statement ID - 1431
   https://sih.gov.in/sih2023PS
   Online survey conducted by us
   https://docs.google.com/spreadsheets/d/1cHEcKlcbbpo0df6XqDH
   PV2TWEU5fi_6gPAmsOTnJ8LE/edit?usp=drive_link
2. Quora
   https://qr.ae/pKYgpr
3. Research Paper reference-
    1. Adomavicius, G., & Tuzhilin, A. (2011). Context-aware recommender systems. *AI Magazine, 32*(3), 67-80.
    2. Brown, M. (2020). The impact of COVID-19 on online education: Challenges and opportunities. *Journal of Digital Learning, 12*(1), 45-58.
    3. Burke, R. (2002). Hybrid recommender systems: Survey and experiments. *User Modelling and User-Adapted Interaction, 12*(4), 331-370.
    4. He, X., Liao, L., Zhang, H., Nie, L., Hu, X., & Chua, T. S. (2017). Neural collaborative filtering. *Proceedings of the 26th International Conference on World Wide Web*, 173-182.
    5. Johnson, S. (2021). Addressing equity and access in online education: A critical analysis. *Education Policy Review, 28*(2), 115-130.
    6. Koren, Y., Bell, R., & Volinsky, C. (2009). Matrix factorization techniques for recommender systems. *Computer, 42*(8), 30-37.
    7. Lops, P., de Gemmis, M., & Semeraro, G. (2011). Content-based recommender systems: State of the art and trends. *Recommender Systems Handbook*, 73-105.
    8. Resnick, P., Iacovou, N., Suchak, M., Bergstrom, P., & Riedl, J. (1994). GroupLens: An open architecture for collaborative filtering of netnews. *Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work*, 175-186.

9. Sarwar, B., Karypis, G., Konstan, J., & Riedl, J. (2001). Item-based collaborative filtering recommendation algorithms. *Proceedings of the 10th International Conference on World Wide Web*, 285-295.
10. Smith, J., Roberts, L., & Williams, T. (2020). Integrating multimedia resources in e-learning: Benefits and challenges. *International Journal of Educational Technology, 15*(3), 23-39.
11. Zhang, Y., & Chen, X. (2020). Explainable recommendation: A survey and new perspectives. *Foundations and Trends in Information Retrieval, 14*(1), 1-101.
12. Zhang, S., Yao, L., Sun, A., & Tay, Y. (2019). Deep learning-based recommender system: A survey and new perspectives. *ACM Computing Surveys, 52*(1), 1-3

APPENDIX

Appendix A: Flowcharts

Flowchart 1: User Registration and Personalization

This flowchart shows the user registration process where users enter their interests and academic status and the system makes recommendations based on that data..

Flowchart 2: Content Recommendation

This flowchart shows how the platform uses this data to generate personalized resources in the user's dashboard, and the recommendations are updated regularly based on user preferences, goals, and progress.

Flowchart 3: Progress Tracking

This flowchart shows how the system records user engagements with the videos, articles and quizzes on the platform and informs future recommendations.

Appendix B: Dataset Description

The FlowState platform utilizes a heterogeneous dataset containing more than 1,000 educational resources from multiple formats (videos, articles, tutorials and eBooks) and reputable platforms (YouTube, Coursera, edX). The data covers various academic domains (Computer Science, Mathematics, Business, etc.). Each resource is categorized based on topics to make sure that accurate recommendations occur.

Appendix C: Performance Metrics

Model Performance Comparison

The Hybrid Model featured by FlowState is more effective than any other recommendation strategies, with an overall measure of success at 93%, a precision score of 90%, and a successful recall rate of 88%. Other models, such as Collaborative Filtering and Content-Based Filtering, have reported lower measures of success across multiple metrics of interest.

User Engagement

Users spending time in FlowState are engaged for an average of 45 minutes in a single session, with a course completion rate of 87%. User feedback was positive with a high 4.8/5 rating for the platform overall, surpassing other common user platforms like Coursera or Udemy, which have users engaged for less time and ratings for user satisfaction of less than FlowState.

Appendix D: Technology Stack

- **Frontend**: ReactJS, HTML/CSS, JavaScript, WebSockets
- **Backend**: Django REST Framework, Django Channels, Celery
- **Database**: PostgreSQL, Redis

Appendix E: User Feedback

The FlowState user feedback was positive. The dashboard functionality received a 4.7/5 rating for personalized recommendations, with requests for more filter options. Progress tracking functionality received a 4.6/5 rating, with users enjoying the quizzes, and requesting more analytics detail.

The community section received a high rating (4.8/5), with users enjoyed the collaboration opportunities but requesting more ways to interact with community members. The rewards system received a positive rating (4.5/5) overall, while some participants requested for even more integration of achievements into social sharing platforms.

TURNITIN PLAGIARISM REPORT

ORIGINALITY REPORT

15%	14%	5%	8%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	irjet.net Internet Source	3%
2	Submitted to KIET Group of Institutions, Ghaziabad Student Paper	2%
3	pdfcoffee.com Internet Source	1%
4	www.coursehero.com Internet Source	1%
5	sahe.in Internet Source	1%
6	www.unescap.org Internet Source	1%
7	www.irjmets.com Internet Source	<1%
8	export.arxiv.org Internet Source	<1%
9	www.amrita.edu Internet Source	<1%
10	arxiv.org Internet Source	<1%
11	opus.lib.uts.edu.au Internet Source	<1%
12	ebin.pub Internet Source	<1%

13	github.com Internet Source	<1%
14	koreascience.or.kr Internet Source	<1%
15	Submitted to Queen Mary and Westfield College Student Paper	<1%
16	sciendo.com Internet Source	<1%
17	open-innovation-projects.org Internet Source	<1%
18	www.ijirset.com Internet Source	<1%
19	technodocbox.com Internet Source	<1%
20	Submitted to University College London Student Paper	<1%
21	hackread.com Internet Source	<1%
22	Submitted to South Bank University Student Paper	<1%
23	link.springer.com Internet Source	<1%
24	Sivaraj Selvaraj. "Mastering REST APIs", Springer Science and Business Media LLC, 2024 Publication	<1%
25	Pires, Francisco José Almeida. "A Recommendation Model for Client Retention: Enhancing Retention in the Telecommunications Industry.", Universidade do Porto (Portugal)	<1%

	Publication	
26	appmaster.io Internet Source	<1 %
27	Submitted to Manipal Academy of Higher Education (MAHE) Student Paper	<1 %
28	www.capilanou.ca Internet Source	<1 %
29	www.geeksforgeeks.org Internet Source	<1 %
30	Submitted to IUBH - Internationale Hochschule Bad Honnef-Bonn Student Paper	<1 %
31	testsigma.com Internet Source	<1 %
32	text-id.123dok.com Internet Source	<1 %
33	jad.shahroodut.ac.ir Internet Source	<1 %
34	repositorio.uam.es Internet Source	<1 %
35	www.indiastudychannel.com Internet Source	<1 %
36	Submitted to HTM (Haridus- ja Teadusministeerium) Student Paper	<1 %
37	en.wikibooks.org Internet Source	<1 %
38	eng.uber.com Internet Source	<1 %
39	fastercapital.com Internet Source	<1 %

40	wiredspace.wits.ac.za Internet Source	<1 %
41	www.jatit.org Internet Source	<1 %
42	www.omeducation.edu.in Internet Source	<1 %
43	תובל, נועה. "Exploring the Potential of the Hypercube-Based Model in Content-Based Recommender Systems", University of Haifa (Israel), 2024 Publication	<1 %
44	eresearch.qmu.ac.uk Internet Source	<1 %
45	paulmckevitt.com Internet Source	<1 %
46	pdffox.com Internet Source	<1 %
47	pt.scribd.com Internet Source	<1 %
48	www.asiapharmaceutics.info Internet Source	<1 %
49	www.slideshare.net Internet Source	<1 %
50	Barakat, Mohammad Osama Salahaldeen. "Pubmed Article Recommendation System Based on Collaborative Filtering", Dokuz Eylul Universitesi (Turkey), 2024 Publication	<1 %
51	Fan Zhou, Yuhua Mo, Goce Trajcevski, Kunpeng Zhang, Jin Wu, Ting Zhong. "Recommendation via Collaborative Autoregressive Flows", Neural Networks, 2020 Publication	<1 %

52	hdl.handle.net Internet Source	<1%

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	Off		

PATENT

(12) PATENT APPLICATION PUBLICATION
(19) INDIA
(22) Date of filing of Application :30/11/2024
(21) Application No.202411094170 A
(43) Publication Date : 17/01/2025

(54) Title of the invention : PERSONALIZED RESOURCE CURATION AND LEARNING GUIDANCE

(51) International classification :G06Q0050200000, H04L0051020000, G16H0050200000, G09B0005020000, G16H0020700000
(86) International Application No :NA
Filing Date :NA
(87) International Publication No : NA
(61) Patent of Addition to Application Number :NA
Filing Date :NA
(62) Divisional to Application Number :NA
Filing Date :NA

(71)**Name of Applicant :**
1)KIET Group of Institutions
Address of Applicant :Delhi-NCR, Meerut Rd Ghaziabad Uttar Pradesh India 201206 Ghaziabad ---------- ----------
Name of Applicant : NA
Address of Applicant : NA
(72)**Name of Inventor :**
1)DR. ABHISHEK GOYAL
Address of Applicant :Computer Science Department, KIET Group of Institutions, Delhi-NCR, Meerut Rd Ghaziabad Uttar Pradesh India 201206 Ghaziabad ---------- ----------
2)Rachitavya Sharma
Address of Applicant :Computer Science Department, KIET Group of Institutions, Delhi-NCR, Meerut Rd Ghaziabad Uttar Pradesh India 201206 Ghaziabad ---------- ----------
3)Prabhakar Mishra
Address of Applicant :Computer Science Department, KIET Group of Institutions, Delhi-NCR, Meerut Rd Ghaziabad Uttar Pradesh India 201206 Ghaziabad ---------- ----------
4)Harshi Agrawal
Address of Applicant :Computer Science Department, KIET Group of Institutions, Delhi-NCR, Meerut Rd Ghaziabad Uttar Pradesh India 201206 Ghaziabad ---------- ----------
5)Rishabh Kushwaha
Address of Applicant :Computer Science Department, KIET Group of Institutions, Delhi-NCR, Meerut Rd Ghaziabad Uttar Pradesh India 201206 Ghaziabad ---------- ----------

(57) Abstract :
The present invention relates to a personalized online learning platform that addresses challenges in online education, such as information overload and lack of guidance. The platform aggregates educational content from diverse sources and uses machine learning algorithms to provide tailored recommendations based on user preferences, goals, and progress. Key features include progress tracking, gamified engagement mechanisms, and a community interaction module for peer collaboration. The platform also incorporates natural language processing for summarizing educational materials, mood prediction algorithms for personalized mental health recommendations, and AI-powered chatbots for real-time assistance. A certification system validates user achievements, while scalability ensures support for diverse learning domains. This invention simplifies resource discovery, enhances engagement, and improves learning outcomes, empowering users to efficiently achieve their educational objectives.

No. of Pages : 19 No. of Claims : 10